

# 1. Grundlagen Digitaler Bildverarbeitung

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

14./21. Oktober 2024

# Kurze Vorstellung meiner Person



**Informatik-Doktorand (2006-09)**

MPI Informatik, Saarbrücken

**Postdoctoral Fellow (2009-11)**

University of Chicago, IL, USA



**Wissenschaftl. MA (2011-14)**

MPI Intelligente Systeme, Tübingen

**Juniorprofessor (2013-2017)**

Informatik, Universität Bonn

**Professor (seit 2017)**  
B-IT und Informatik, Universität Bonn

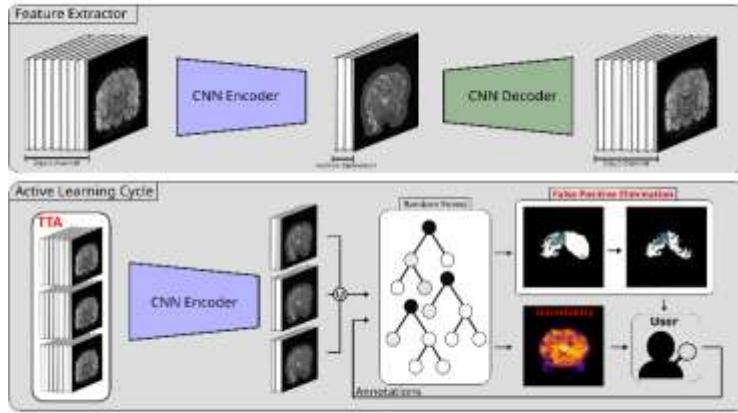


# Kurze Vorstellung unserer Gruppe

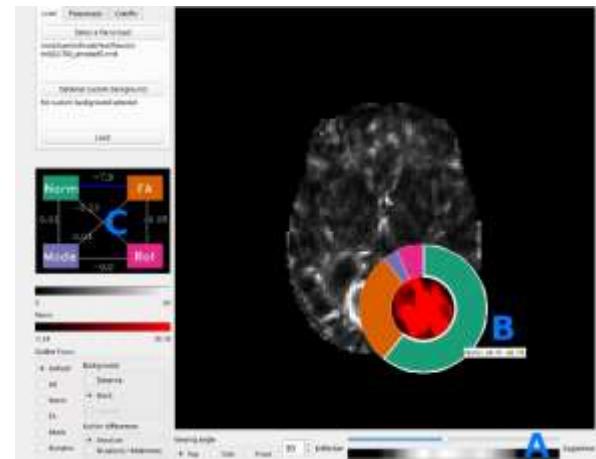
- **Arbeitsgebiet: Visualisierung und Medizinische Bildanalyse**
  - Aktuell 1 Postdoktorand, 3 aktive Doktorand\*innen, 3 HiWis, 6 aktive Abschlussarbeiten (MSc/BSc)



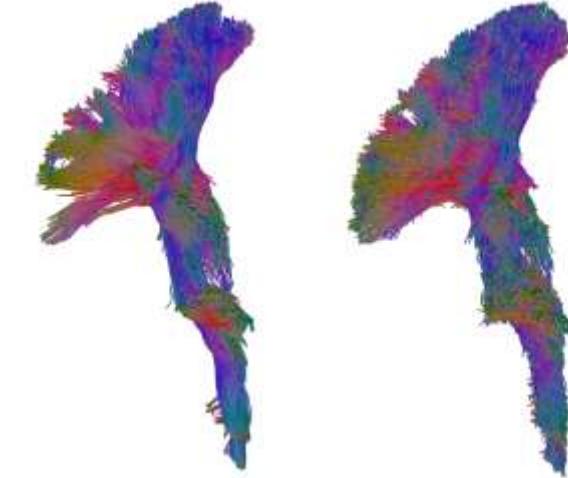
# Publikationen aus Abschlussarbeiten heraus



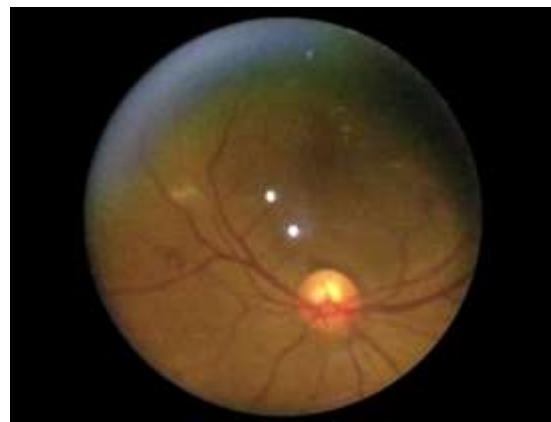
[Lennartz et al., UNSURE 2024]



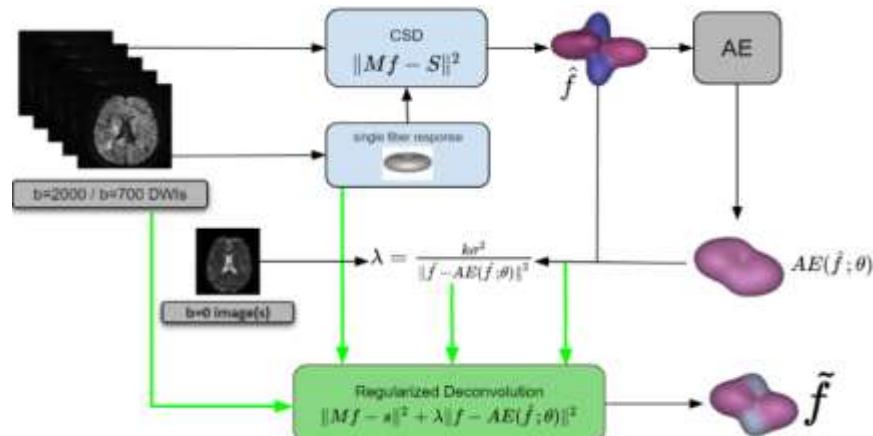
[Bareth et al., EuroVis SP 2023]



[Grün et al., VCBM 2021]



[Mueller et al., OMIA 2020]



[Patel et al., MICCAI 2018]

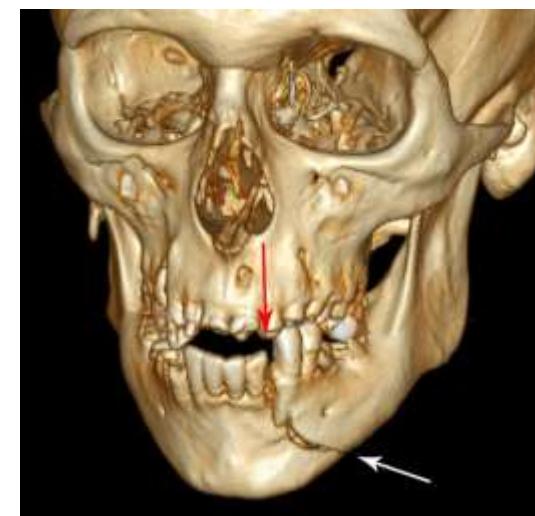


[Soundararajan et al., EuroVis 2015] 4

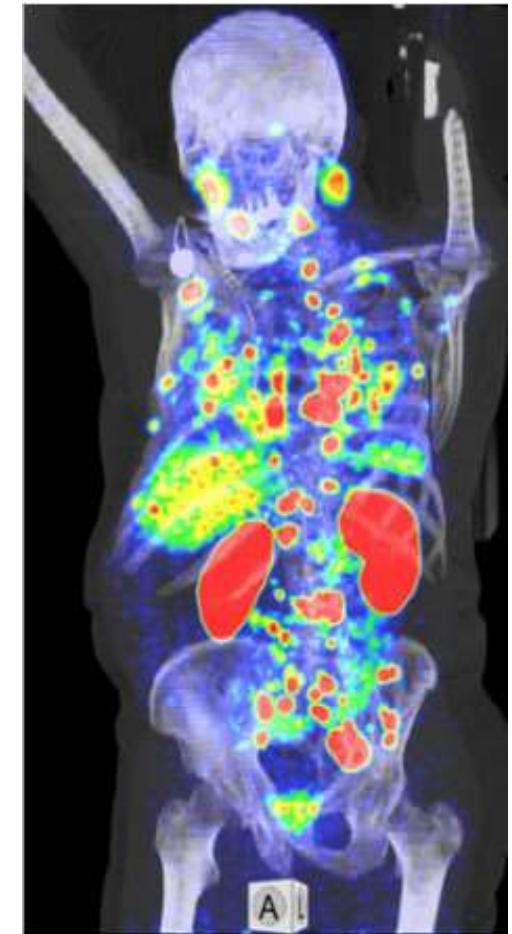
# **1.1 Geplante Inhalte dieser Vorlesung**

# Motivation: Bildgebung in der Medizin

- **Bildgebende Verfahren** liefern der Medizin ansonsten nicht zugängliche Informationen zu *Struktur* und *Funktion*, insbesondere im Körperinneren
- **Algorithmen** sind für die Rekonstruktion, (Vor-)verarbeitung und Visualisierung unerlässlich
  - *Beispiel:* Computertomographie



3D-Rekonstruktion  
aus CT-Scan



Fusioniertes  
PET/CT

# Motivation: Bildgestützte Diagnostik

In vielen Bereichen der **medizinischen Diagnostik** ist die Bildgebung Teil der alltäglichen Routine.



Pädiatrische Echokardiografie



Zahnmedizinische Panoramäröntgenaufnahme

# Motivation: Bildgestützte Intervention

Bildgebung unterstützt häufig die **Planung** oder sogar **intraoperativ** die **Durchführung** von Interventionen wie chirurgischen Eingriffen oder Strahlentherapien



OP-Planung mittels Transkranieller Magnetstimulation



Intraoperatives CT

# Motivation: Autonome Bildgestützte Diagnostik

2018 wurde mit IDx-DR das erste autonome diagnostische System in den USA zugelassen (Erkennung diabetischer Retinopathie)

- 2023: Mehr als 200 KI-basierte Produkte in der Radiologie



<https://www.eyediagnosis.co/>

Autonomous AI  
that instantly  
detects disease

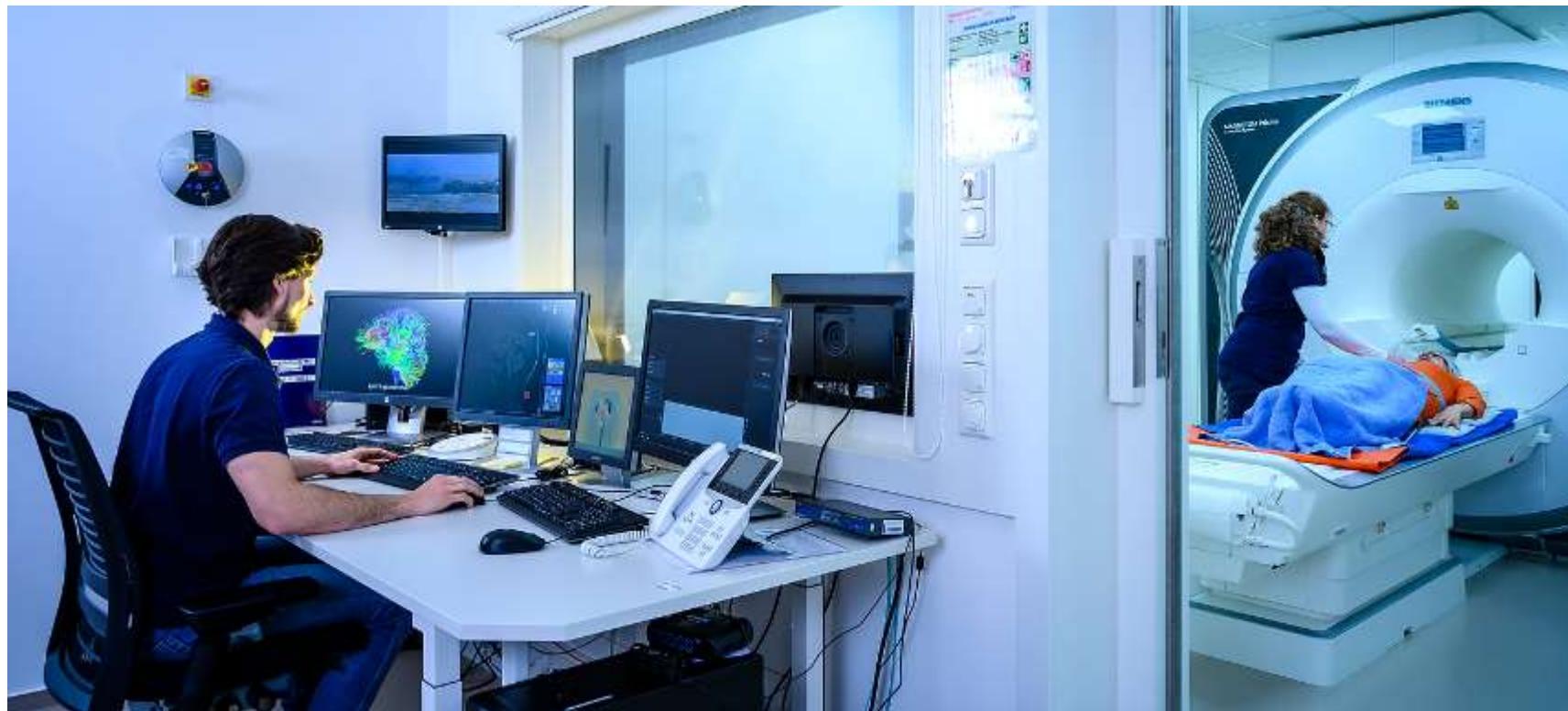
Increase patient access to early  
disease detection.

Learn More

# Motivation: Bildgestützte Gesundheitsforschung

Viele bildgebende Verfahren sind sicher genug um damit freiwillige Teilnehmer von **Gesundheitsstudien** zu untersuchen

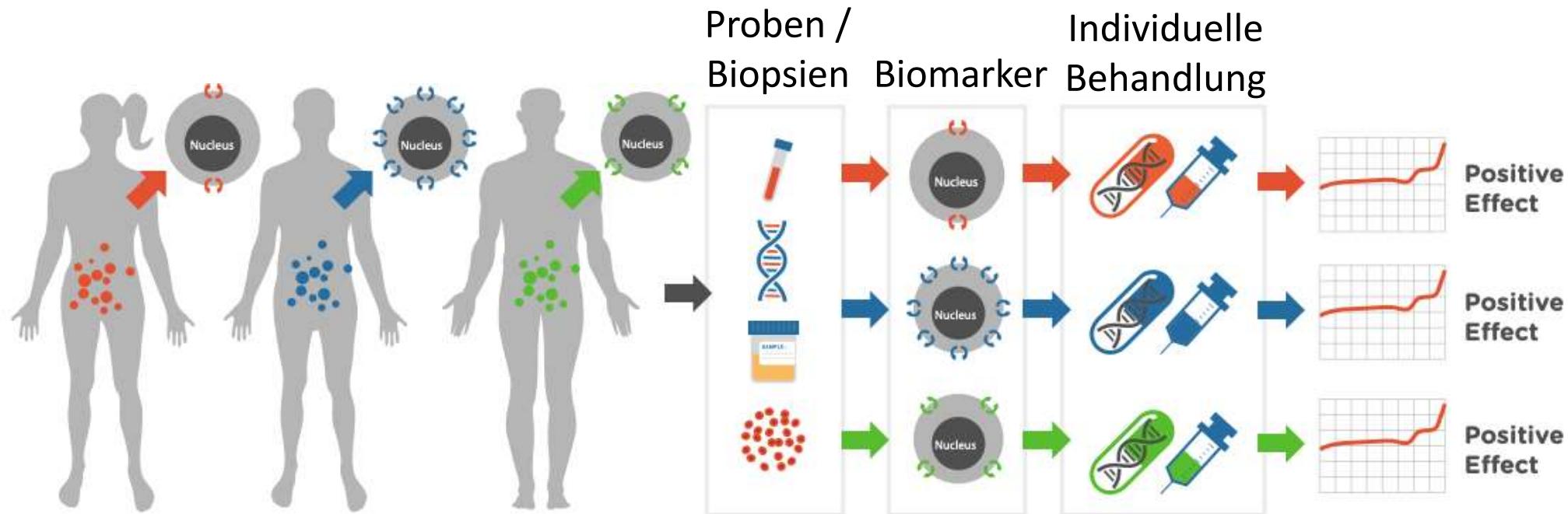
- „**nichtinvasive**“ Bildgebung
- *Beispiel:* Rheinland-Studie zu gesundem Altern am DZNE in Bonn



# Ausblick: Präzisionsmedizin

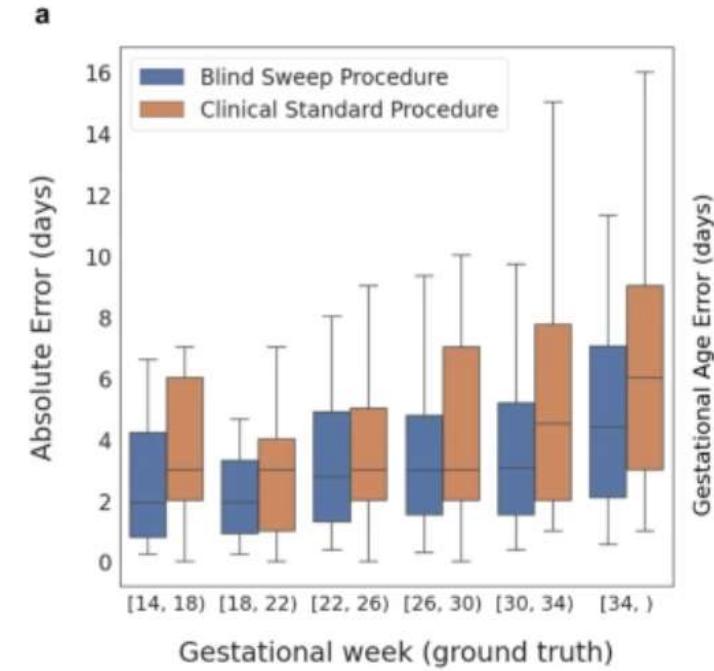
Als **Präzisionsmedizin** bezeichnet man eine stärkere Anpassung medizinischer Behandlungen an den individuellen Patienten

- *Beispiel:* Therapieentscheidungen in der Tumormedizin
- Aktuell primär auf Grundlage von Genanalysen
- *Hoffnung:* Durch Bildgebung Phänotypen stärker berücksichtigen



# Ausblick: Medizinische Versorgung im Globalen Süden

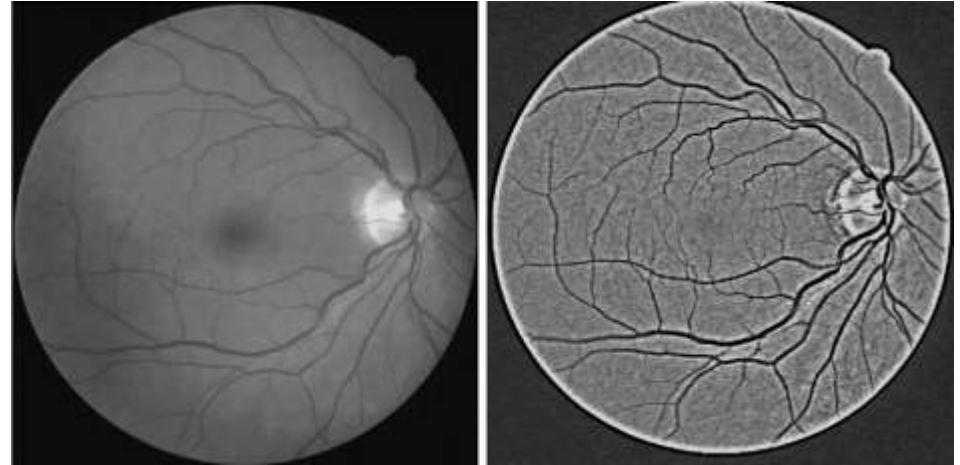
Eine automatische Befundung mittels tragbaren Sensoren und Computern könnte die Lage in medizinisch unversorgten Gegenden verbessern



# 1. Grundlagen Digitaler Bildverarbeitung

## Vorverarbeitung von Bildern zur

- Kontrastverbesserung
- Rauschunterdrückung
- Schärfung
- Kantenerkennung



Kontrastverstärkung in Retina-Aufnahme

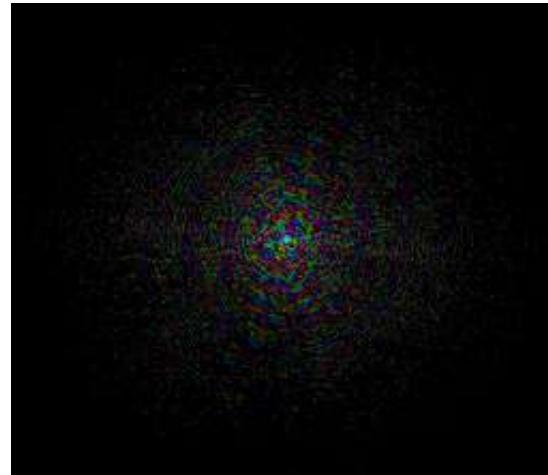


Zunehmende Glättung eines MRT-Scans

## 2. Grundlagen der Signaltheorie

Verständnis von **Theorie und praktischen Konsequenzen** von

- Abtasttheorem und Aliasing
- Fourier-Transformation
- Vergrößerung / Verkleinerung der Bildauflösung



MRT-Bild und seine Fourier-Transformation



Fehlerhaft / sinnvoll reduzierte Bildauflösung<sup>14</sup>

# 3. Bildgebende Verfahren

## Funktionsweise und Bildcharakteristika von

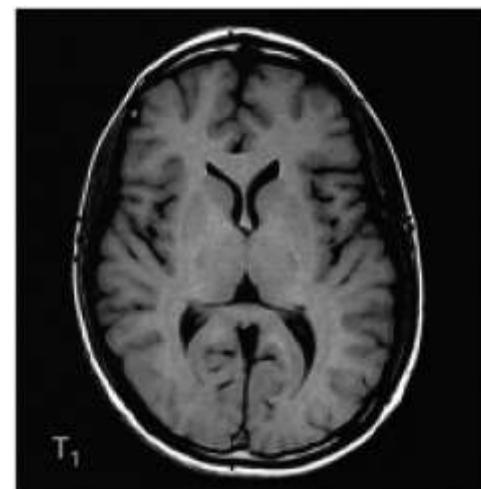
- Röntgenbildgebung
- Computertomographie
- Emissionstomographie
- Magnetresonanztomographie
- Ultraschall
- (Optischer Kohärenztomographie)



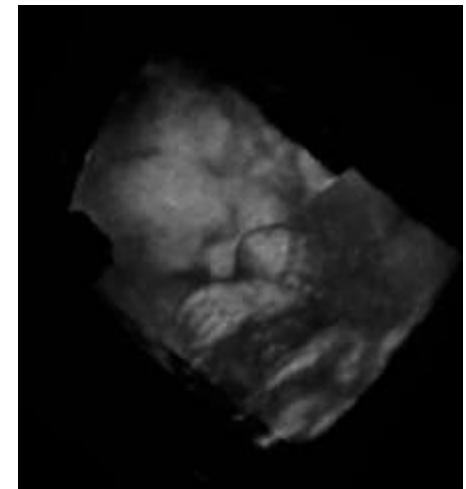
Röntgenbild



CT-Scanner



MRT-Scan

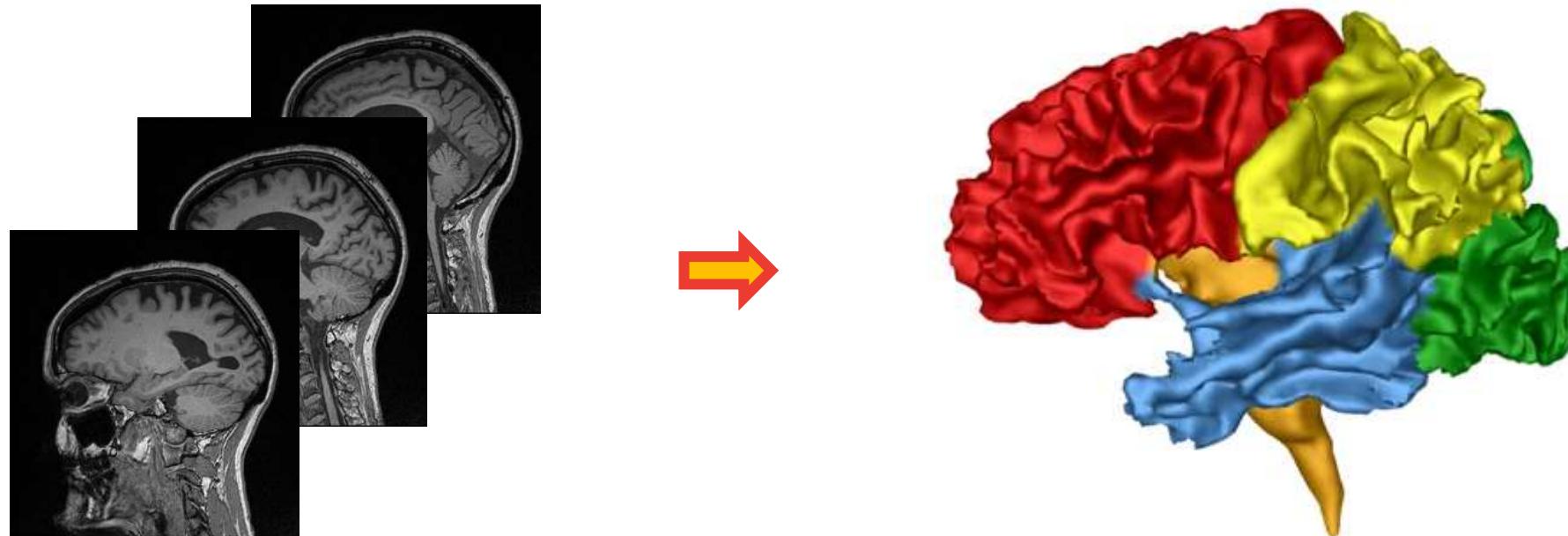


3D-Ultraschall

# 4. Bildsegmentierung

**Abgrenzung von Bildobjekten (Organe, Tumore, etc.)**

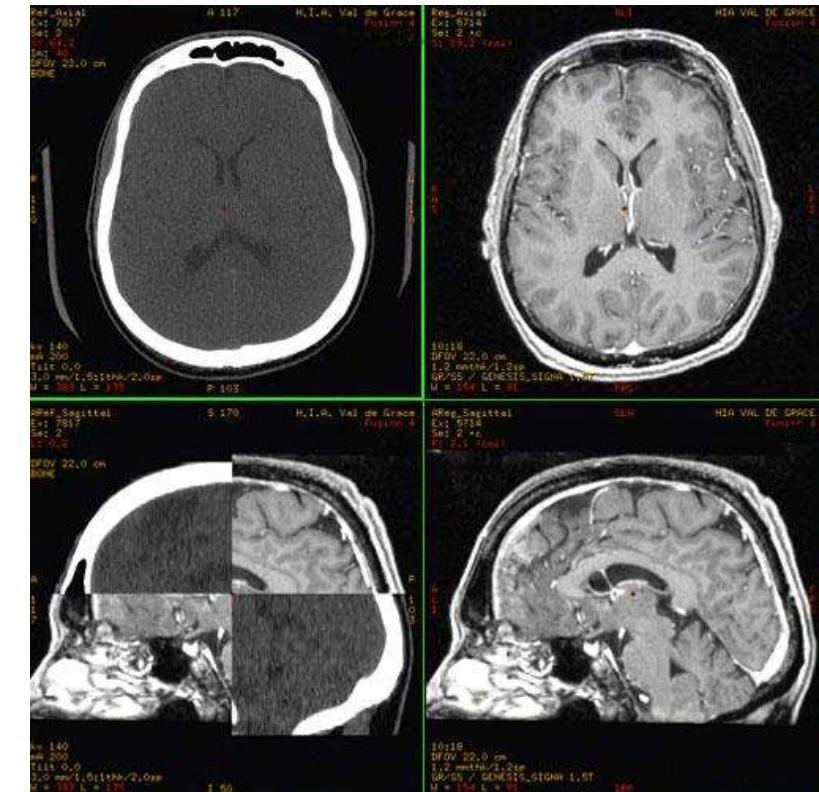
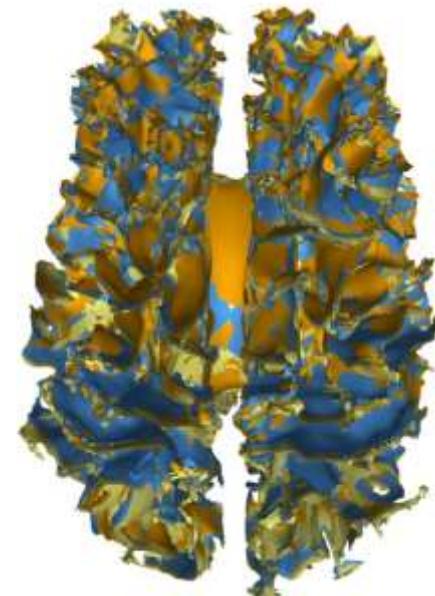
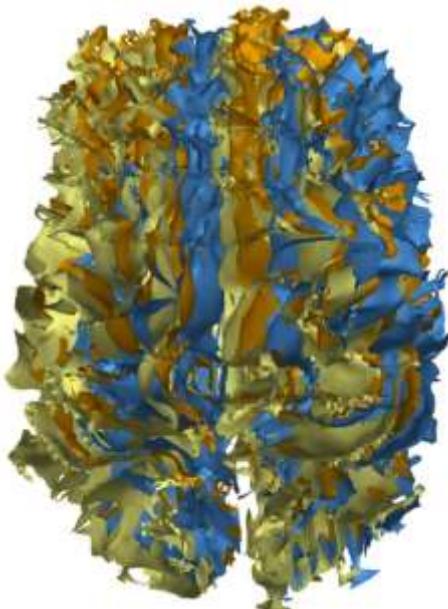
- Schwellenwert-Verfahren
- Aktive Konturen / Deformierbare Modelle
- Formmodelle



# 5. Bildregistrierung

## Bilder in Korrespondenz bringen

- Bildtransformationen
- Kostenfunktionen
- Optimierungsalgorithmen



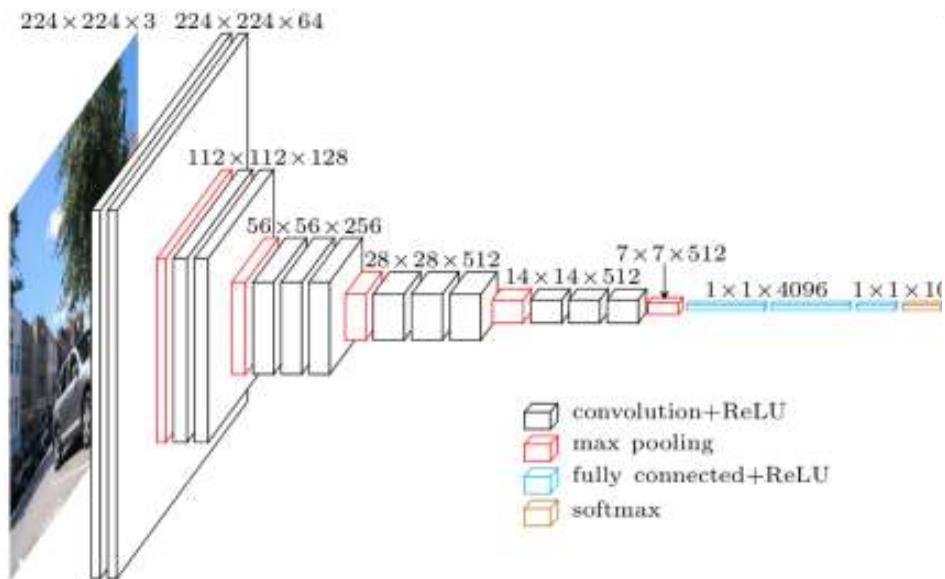
Registrierung von CT/MRT

Registrierung wiederholter Scans

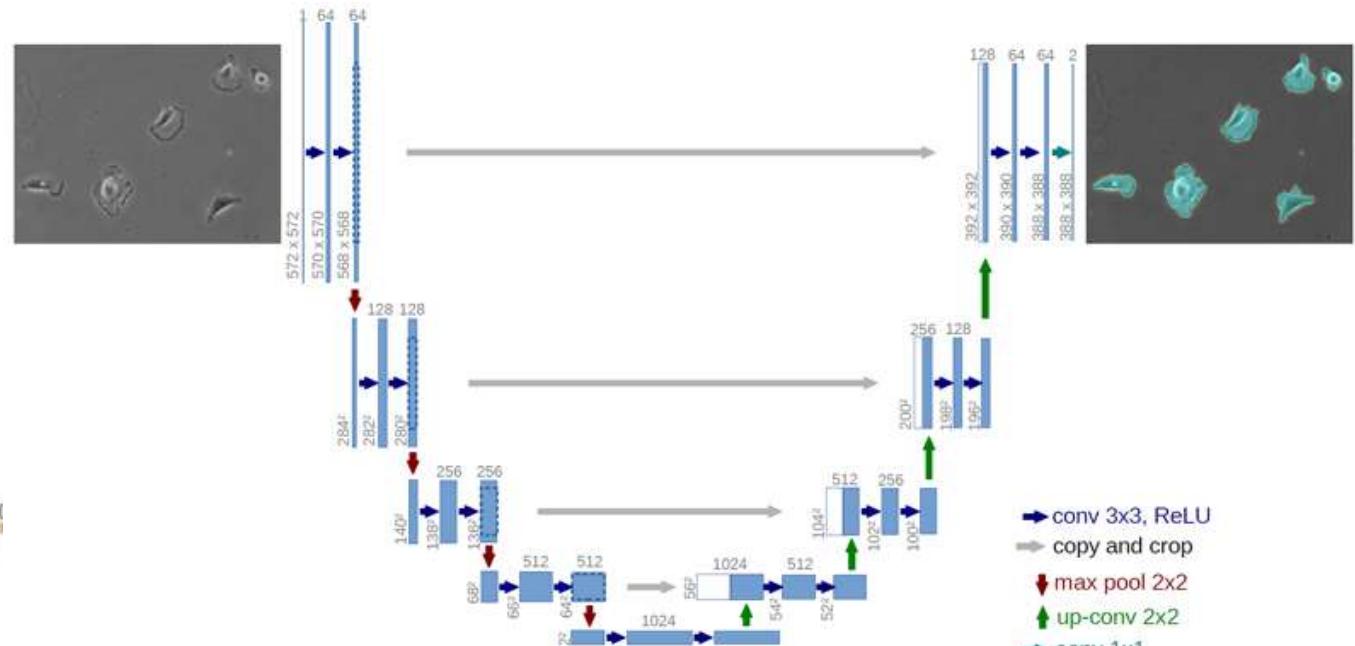
# 6. Bildanalyse mittels Deep Learning

## Funktionsweise und Anwendungsbeispiele neuronaler Netze

- Faltungsnetzwerke (CNNs)
- Bildklassifizierung
- Bildsegmentierung



Bildklassifizierung [Simonyan/Zisserman, 2014]



Bildsegmentierung  
[Ronneberger et al., MICCAI 2015]

## **1.2 Organisatorisches**

# **Vorlesungen und eCampus**

- **Vorlesungen** finden jeden Montag um 14 c.t. im HSZ HS3 statt
- Wir nutzen **eCampus**
  - um Ihnen Folien und ggf. weitere Materialien zur Verfügung zu stellen
  - für die Übungsabgaben
  - als öffentliches Forum für Fragen und Diskussionen
  - falls nötig, um ZOOM-Links / Vorlesungsvideos zu verbreiten
    - Ich versuche die Vorlesung wann immer möglich in Präsenz zu halten!
  - Bitte tragen Sie sich als Kursmitglied ein um an den Übungen teilzunehmen und ggf. wichtige Ankündigungen zu erhalten

# Übungsbetrieb

- **Übungen**
  - Voraussetzung um zur Klausur zugelassen zu werden
    - *Kriterium:*  $\geq 50\%$  der Punkte insgesamt, mindestens eine Präsentation
    - Bilden Sie bitte möglichst **Gruppen von drei Studierenden**, jedoch nicht mehr.
    - Gruppieren Sie sich im Laufe des Semesters falls nötig gern um.
  - Es wird 11 reguläre Übungsblätter geben
    - Plus Probeklausur am Ende, die zur Zulassung *nicht* mitzählt
  - Übungen werden jeden Montag veröffentlicht, eine Woche später eingereicht, am Donnerstag besprochen
  - Globalübung **Donnerstags um 14 c.t.** im HSZ, HS 4
    - Diese Woche: Sprechstunde für eventuelle Rückfragen

# Hinweise zu den Übungen

- **Praktische Erfahrung mit Implementierung und Anwendung relevanter Algorithmen** zählt zu unseren zentralen Lernzielen
  - Sie profitieren davon, obwohl die Klausur keine Programmieraufgaben im engeren Sinne enthält
  - Diese Fähigkeiten benötigen Sie spätestens für eine Projektgruppe oder BSc-Arbeit in verwandten Bereichen
- **(Online-)Recherchen sind Teil des Lernprozesses**
  - und kein Anzeichen dafür, dass die Aufgabe falsch gestellt wäre
- **Wir tolerieren keine Plagiate!**
  - Für Lösungen, die in sehr ähnlicher Form von mehreren Gruppen eingereicht wurden oder offensichtlich kopiert sind, können wir keine Punkte vergeben.

# Kreditpunkte

- Am Semesterende wird es eine **Klausur** geben
  - Vorläufige Planung: 10. Februar und 2. April
  - Genaue Daten und Zeiten gebe ich bekannt, sobald sie feststehen
- Die bestandene Klausur ist im Wahlpflichtbereich des **BSc Informatik** 6 ECTS wert
  - Haben wir Teilnehmer\*innen aus anderen Studiengängen?

## **1.3 Punktweise Bildtransformationen und Histogramme**

# Wie können wir Bilder (mathematisch) beschreiben?

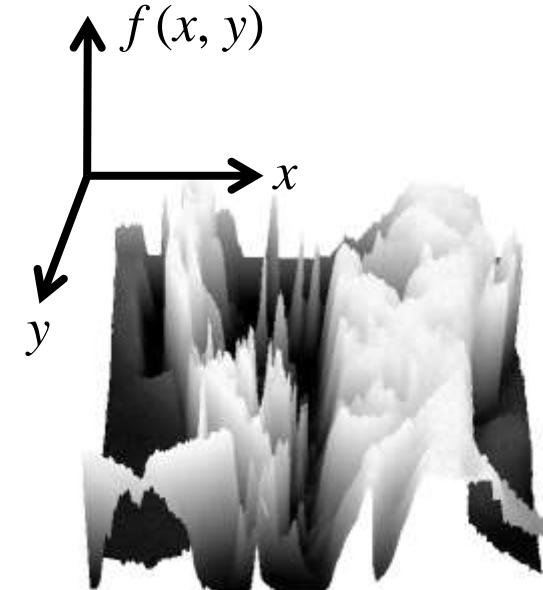


# Bilder als Funktionen

- Bilder können als **Funktion** (2D-Signal) beschrieben werden:

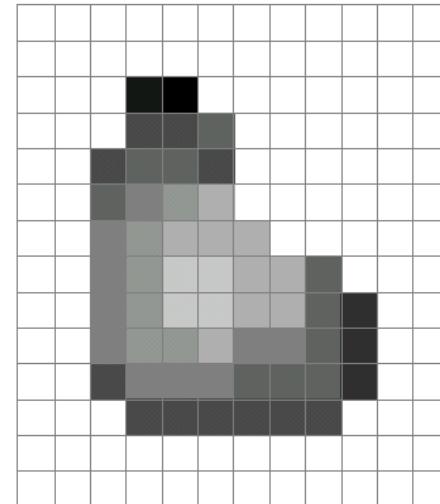
$$f: \mathbb{R}^2 \rightarrow \mathbb{R}$$

- $f(x,y)$  ist die **Intensität** des Bildes an Position  $(x,y)$
- $\mathbf{f}(x,y)$  kann ein Vektor sein, z.B. im Fall von Farbbildern
  - *Beispiel:* RGB = Intensitäten der rot/grün/blau-Kanäle



# Bilder als Matrizen

- **Digitale Bilder** sind eine Diskretisierung der Intensitäts-Funktion
  - **Abtastung (Sampling)** = räumliche Diskretisierung in **Pixel**
  - **Quantisierung** = Diskretisierung der Werte (z.B. Rundung auf ganze Zahlen)
    - *Beispiel:* 8bit “Tiefe”, 0 = schwarz, 255 = weiß
    - Farbbilder haben mehrere **Kanäle**



=

255	255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255	255	255
255	255	127	145	200	200	175	175	175	95	255	255	255	255
255	255	127	145	200	200	175	175	175	95	47	255	255	255
255	255	127	145	145	175	127	127	95	95	47	255	255	255
255	255	74	127	127	127	95	95	95	95	47	255	255	255
255	255	255	74	74	74	74	74	74	74	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255	255

# Filterung von Bildern

- Ein **Bildfilter**  $T: f \mapsto g$  ist ein Operator, der ein Eingabebild  $f$  auf ein Ausgabebild  $g$  abbildet
- *Beispiele:*



$$g(x, y) = f(x, y) + 20$$



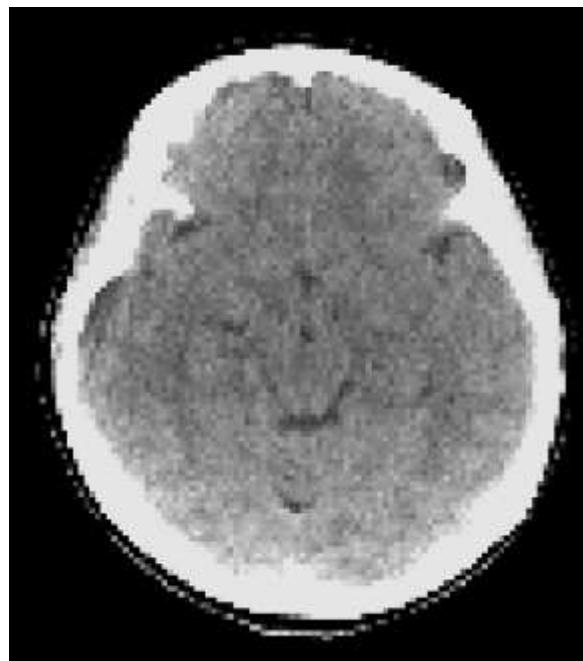
$$g(x, y) = f(-x, y)$$

- Ein **punkt- oder pixelweiser Operator** lässt sich schreiben als  
$$g(x, y) = t(f(x, y))$$

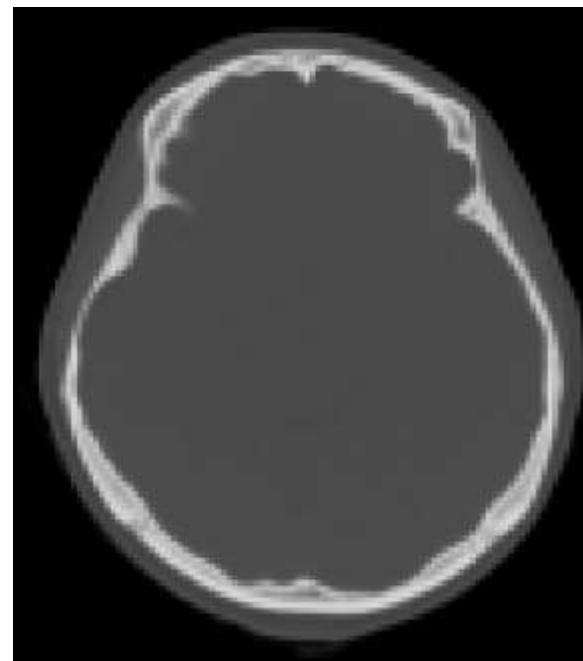
# Fensterung

Als **Fensterung** bezeichnet man eine Einschränkung des Wertebereichs medizinischer Bilder auf ein “Fenster”

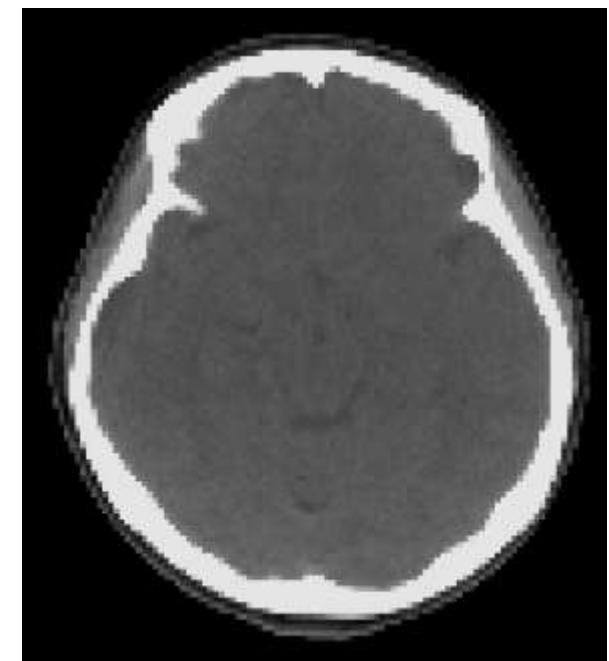
- Größere / kleinere Werte werden weiß / schwarz dargestellt
- Optimiert den Kontrast für Strukturen innerhalb des Fensters



Gehirnfenster



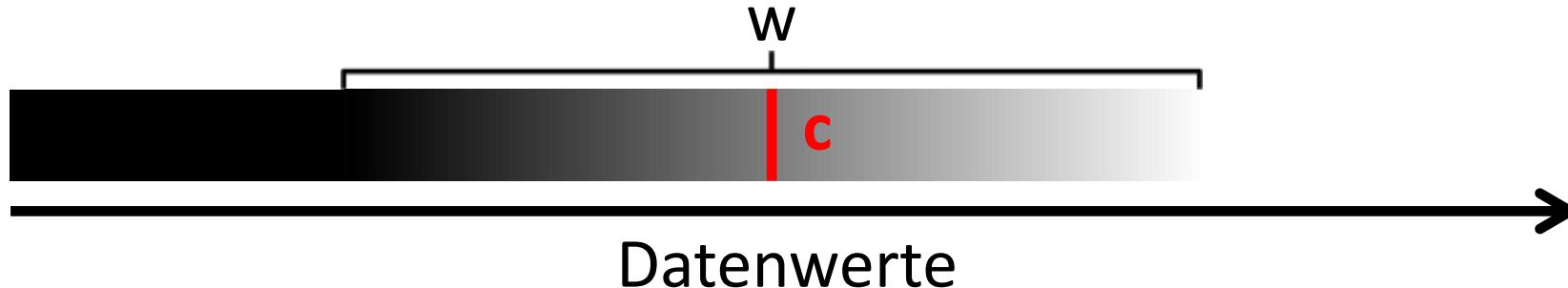
Knochenfenster



Blutungs-Fenster

# Spezifikation von Fenstern

Fenster beschreibt man durch ein Zentrum  $c$  und eine Breite  $w \geq 1$ .



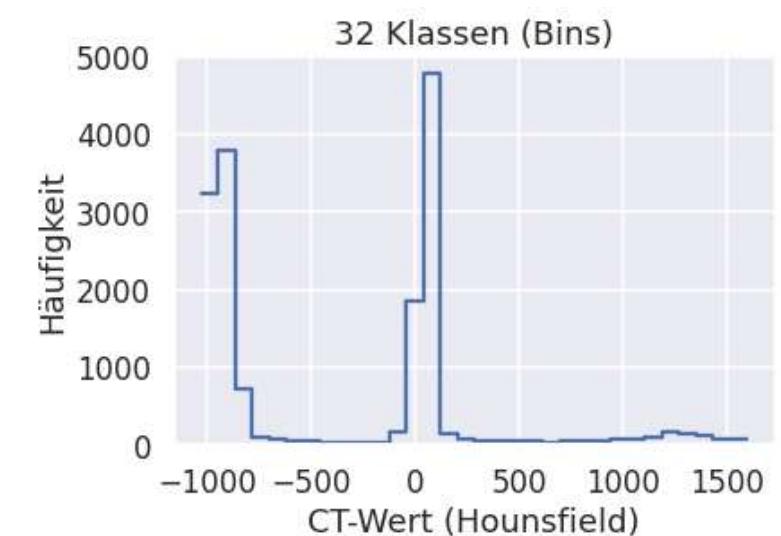
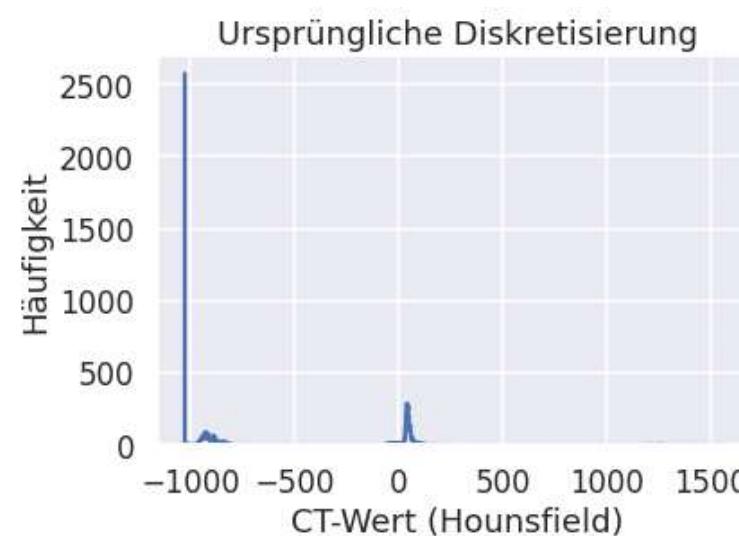
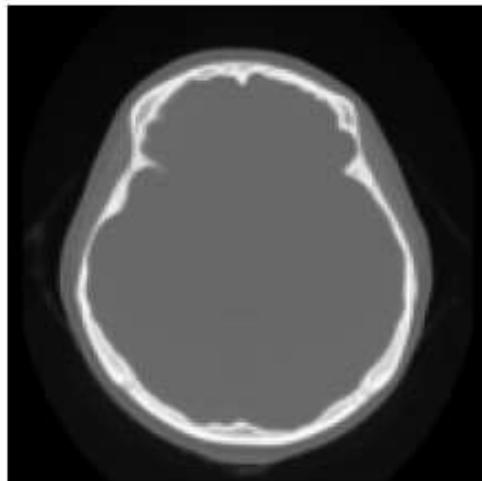
Für Datenwert  $x$  und auf  $C \in [0,1]$  normierte Graustufen beschreibt der folgende Pseudocode die Fensterung:

```
if (x <= c - 0.5 - (w-1)/2) then C := 0  
else if (x > c - 0.5 + (w-1)/2) then C := 1  
else C := (x - (c - 0.5)) / (w-1) + 0.5
```

- Ausführung in Fließkomma-Arithmetik, Skalierung/Rundung am Ende
- $w=1$  führt zu einer Binarisierung (Schwellenwertbildung)

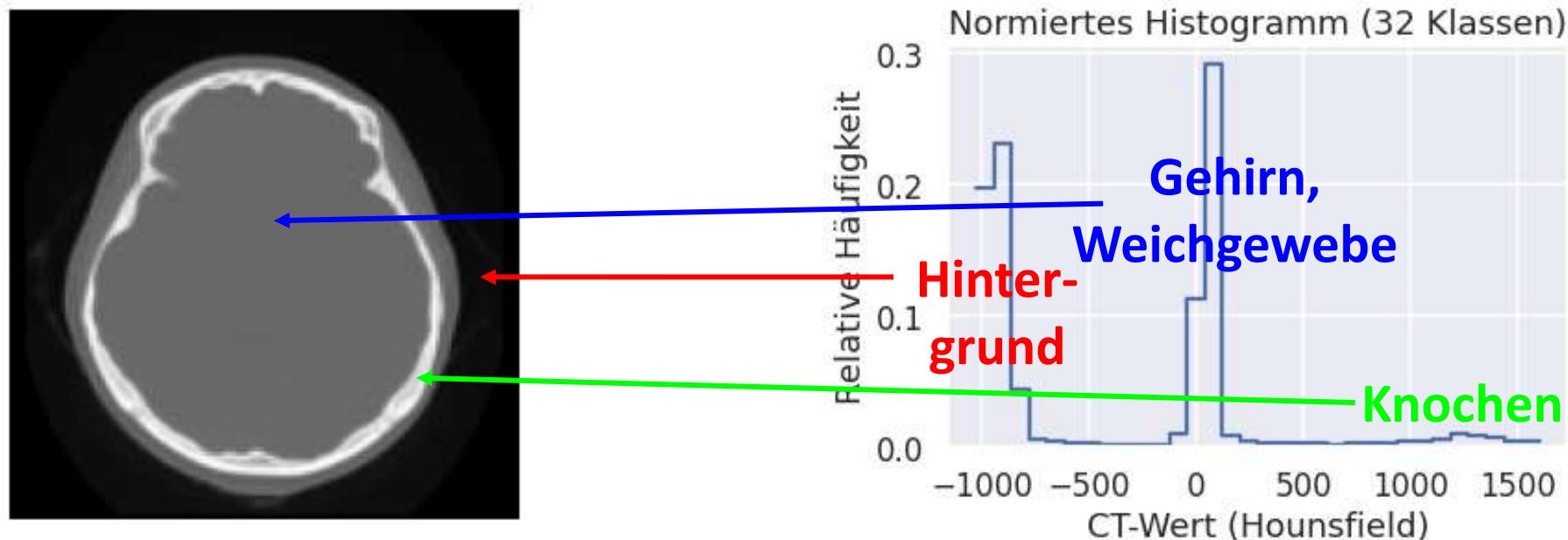
# Histogramme

- **Histogramme** beschreiben die Häufigkeitsverteilung von Intensitätswerten in einem Bild
  - $H(I)$  nutzt die gegebene (ganzzahlige) Diskretisierung der Werte und zählt wie viele Pixel den Wert  $I$  haben
    - *Alternativ:* Grobere Diskretisierung („Binning“) fasst ähnliche Werte zusammen
  - Das mit der Zahl  $N$  der Pixel normierte Histogramm  $H_n(I) := \frac{1}{N} H(I)$  kann man als *Wahrscheinlichkeitsverteilung* interpretieren



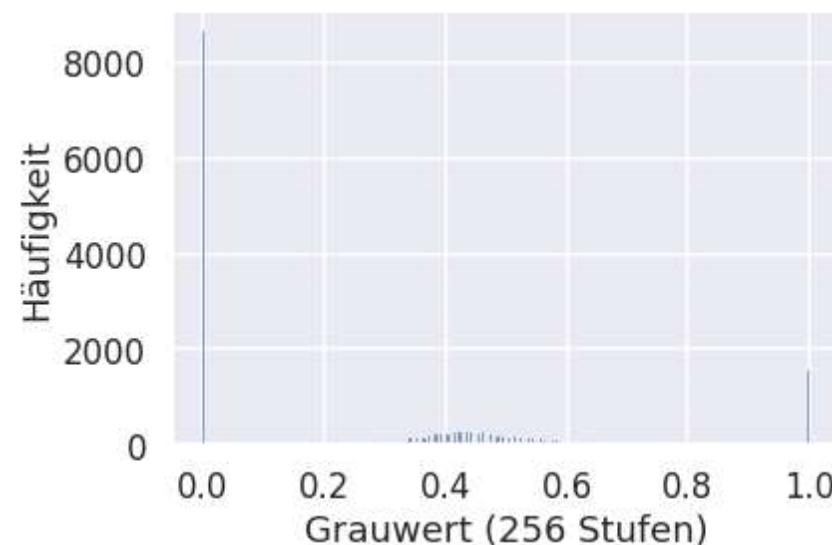
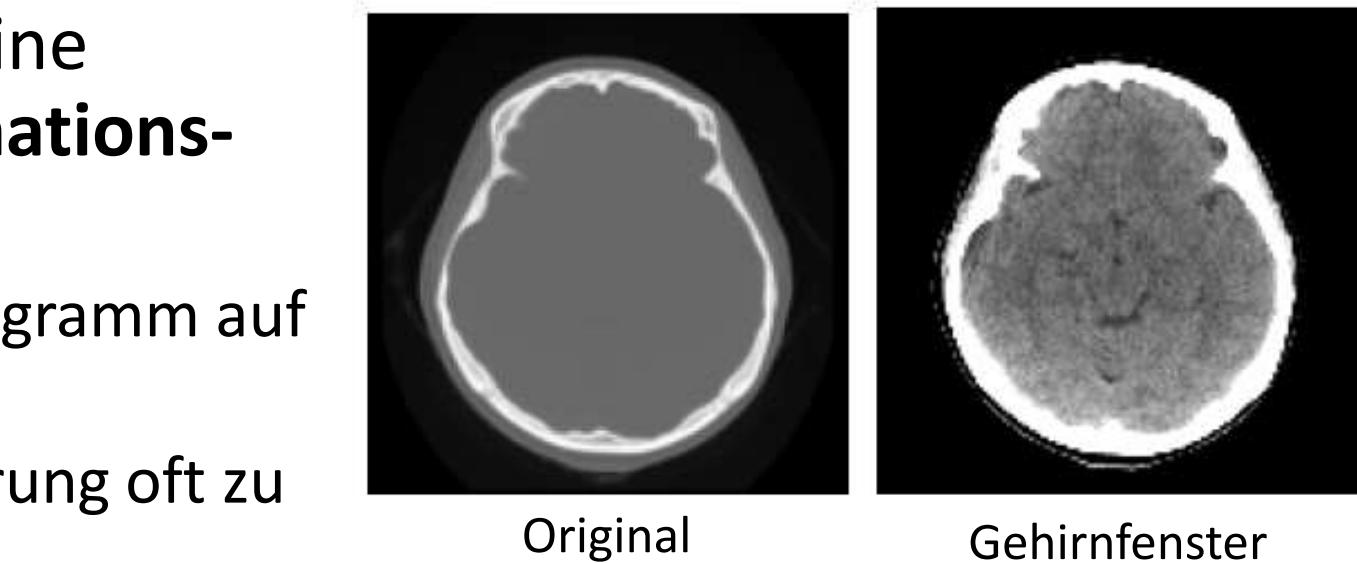
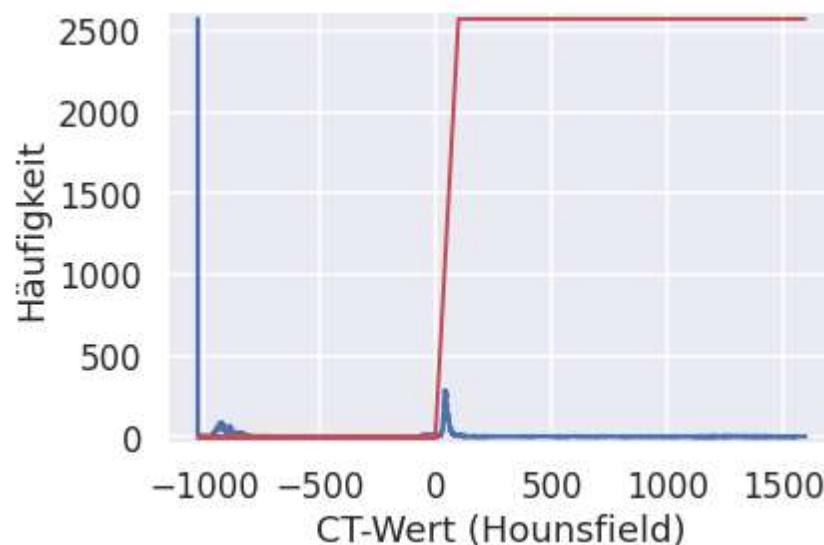
# Histogramm-Analysen

- Häufig haben verschiedene Materialien oder Gewebetypen charakteristische Intensitäten, die als **Gipfel im Histogramm** erkennbar sind
  - **Histogramm-Analysen** bieten Anhaltspunkte für geeignete Fenster
  - **Täler im Histogramm** bieten sich als Schwellenwerte an



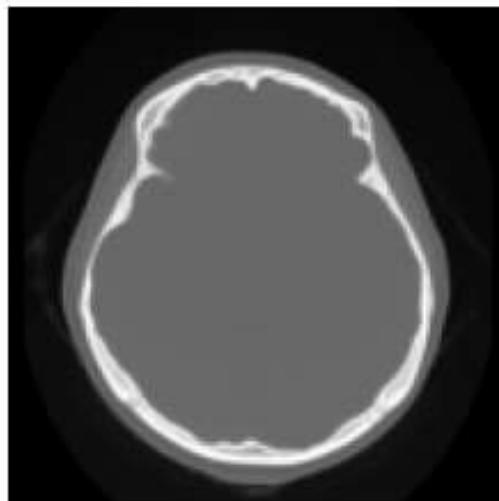
# Fensterung als Histogrammtransformation

- Fensterung lässt sich durch eine stückweise lineare **Transformationskennlinie** beschreiben
  - Begrenzt und spreizt das Histogramm auf das Fenster
  - Führt aufgrund der Diskretisierung oft zu **Lücken im Histogramm**

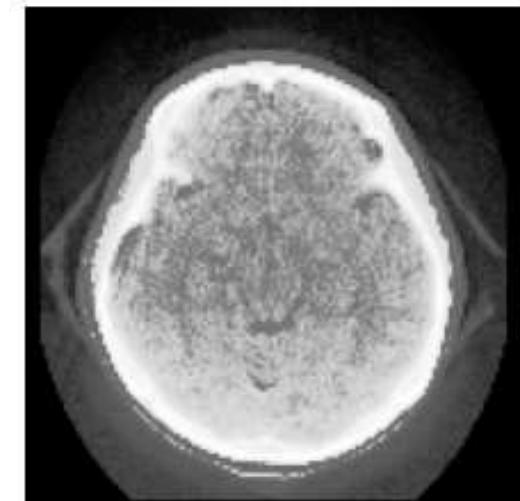
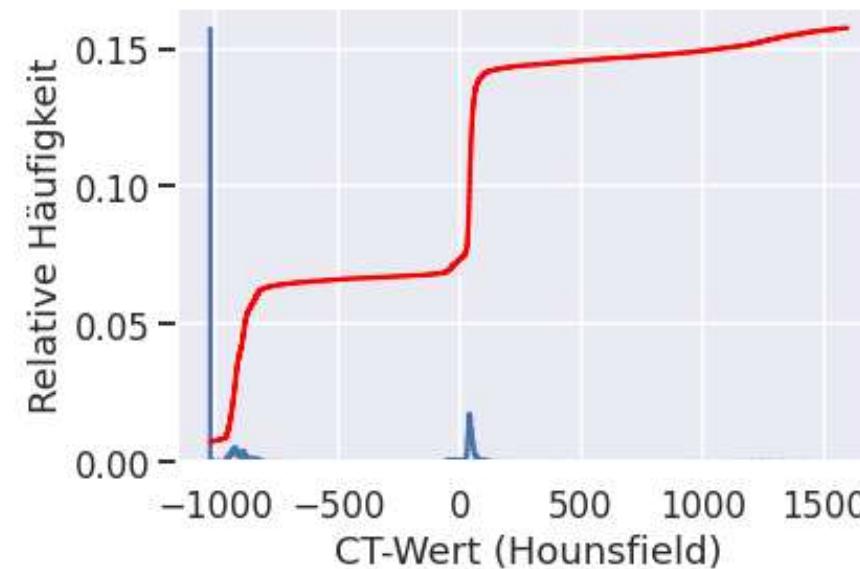


# Histogrammausgleich: Definition

- Ein **Histogrammausgleich**  $t_{HA}(I)$  (engl. *histogram equalization*) strebt eine Kontrastoptimierung durch Gleichverteilung aller Intensitäten an
  - Die Transformationskennlinie ergibt sich aus dem **kumulativen Histogramm**  $H_k(I) := \sum_{i=0}^I H_n(i) = P(i \leq I)$
  - Bei Skalierung der Ausgabe auf  $[0, I_{\max}]$ :  $t_{HA}(I) := I_{\max} \times H_k(I)$



Original



Histogrammausgleich

# Histogrammausgleich: Theoretische Begründung

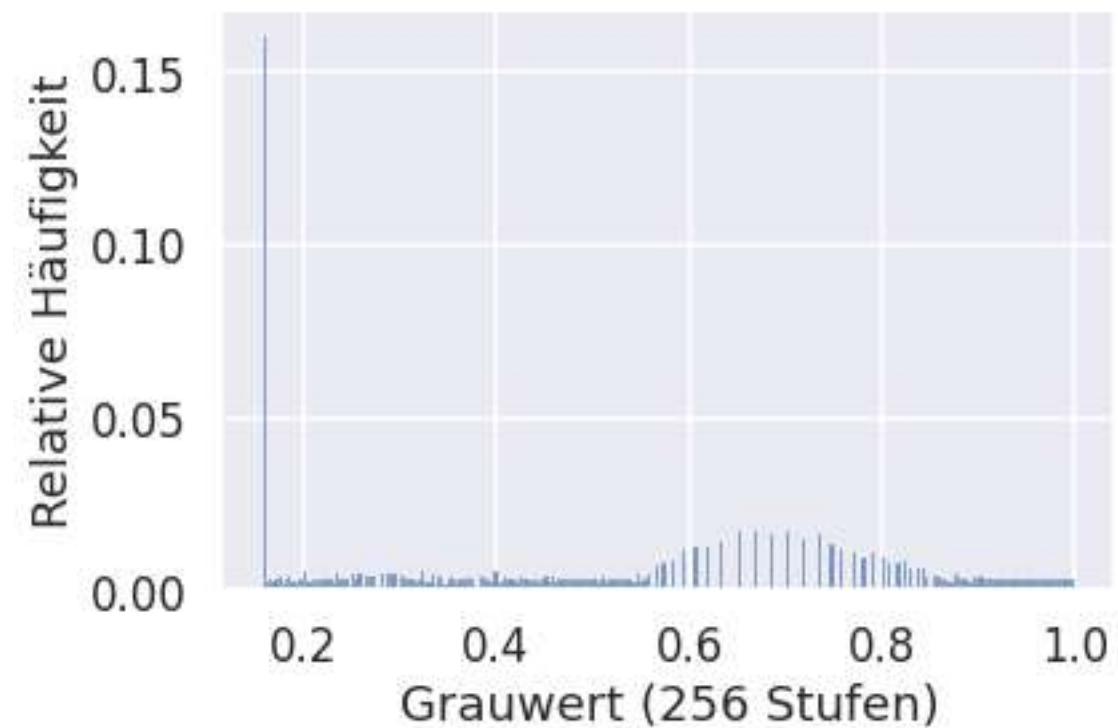
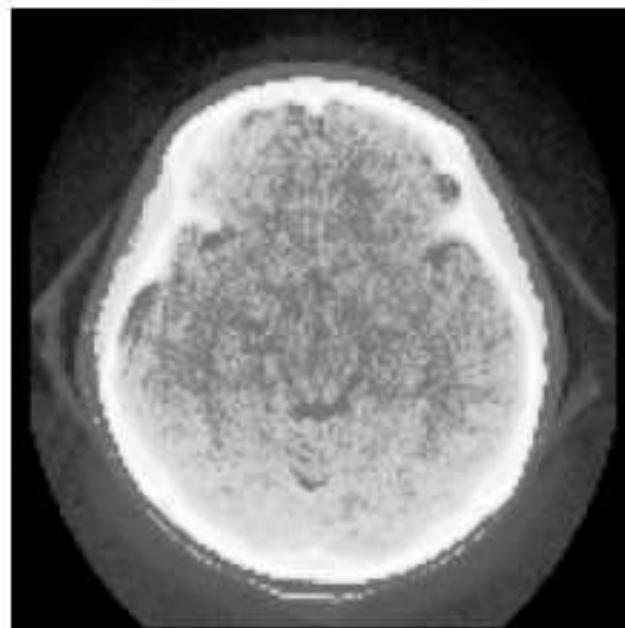
- **Erinnerung:**  $t_{HA}(I) := I_{\max} \times H_k(I)$
- **Begründung:** Wenn  $H_k$  invertierbar ist, erreicht  $t_{HA}(I)$  die gewünschte Gleichverteilung auf  $[0, I_{\max}]$ :

$$\begin{aligned} P(t_{HA}(I) \leq J) &= P(I_{\max} \times H_k(I) \leq J) \\ &= P\left(I \leq H_k^{-1}\left(\frac{J}{I_{\max}}\right)\right) = H_k\left(H_k^{-1}\left(\frac{J}{I_{\max}}\right)\right) = \frac{J}{I_{\max}} \end{aligned}$$

↑  
Definitionsgemäß ist  $H_k(I) = P(i \leq I)$

# Histogrammausgleich in der Praxis

- Dominante Werte im Ausgangshistogramm kann  $t_{HA}(I)$  nicht vollständig ausgleichen
- Wegen Rundung der Ergebnisse und möglicher Lücken im Ausgangshistogramm ist  $t_{HA}(I)$  in der Praxis **nicht invertierbar!**



# Zusammenfassung: Punktweise Bildtransformationen

- Wir können Bilder als **kontinuierliche Funktionen** darstellen, digitale Bilder als **ganzzahlige Matrizen**
- **Bildfilter** sind Operatoren, die Bilder auf Bilder abbilden
- **Histogramme** geben die Häufigkeitsverteilung von Intensitätswerten in Bildern an
- **Punktweise Bildfilter** transformieren die Intensität an derselben Stelle. Sie dienen insb. zur Kontrastverstärkung
  - **Fensterung** relevanter Intensitätsbereiche
  - **Histogrammausgleich** mittels kumulativer Histogramme

## **1.4 Lineare Bildfilterung**

# Motivation: Reduzierung von Bildrauschen

- Wie können wir Bildrauschen reduzieren, wenn wir mit einer Kamera eine statische Szene aufnehmen?

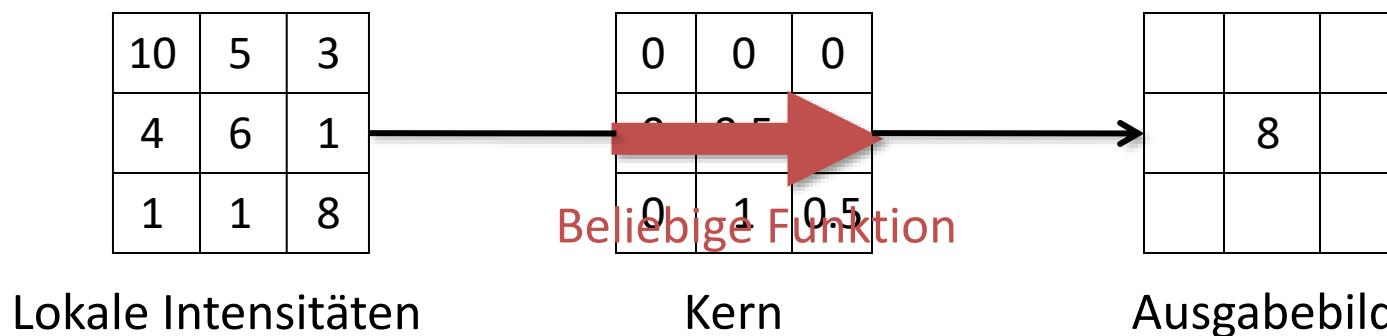


Idee: Mitteln wiederholter Aufnahmen

Was, wenn uns nur eine Aufnahme zur Verfügung steht?

# Lokale Bildfilterung

- **Lokale Bildfilter** berechnen neue Pixelwerte als Funktion der Werte in einer lokalen Nachbarschaft
  - Lokalität reduziert den Rechenaufwand
- *Beispiel:* Linearkombination benachbarter Werte
  - Die Gewichte der Linearkombination werden als “Kern” des Filters bezeichnet

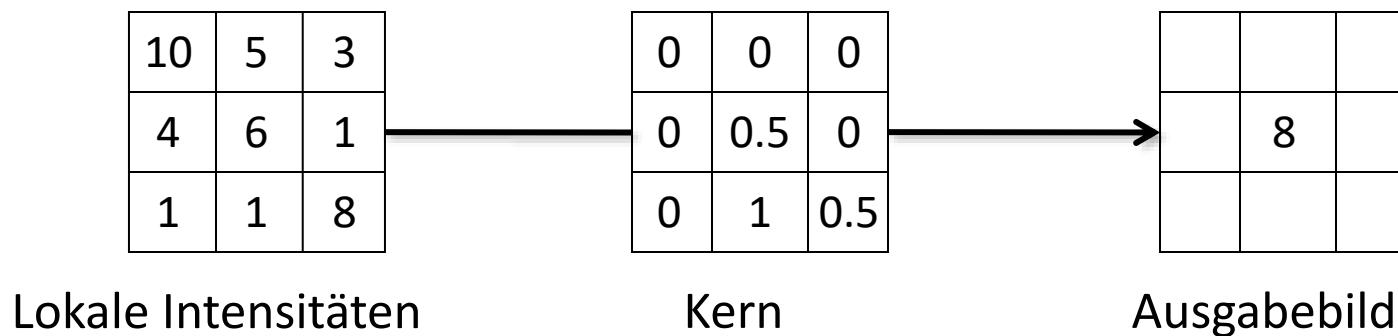


# Lineare Filterung

- Ein Filter  $T$  ist **linear** wenn für Bilder  $f$  und  $g$  gleicher Größe und pixelweiser Addition/Skalierung folgendes gilt:

$$T(f + g) = T(f) + T(g)$$
$$T(\alpha f) = \alpha T(f)$$

- Beispiel:* Lokale Linearkombinationen mit festen (vom Bildinhalt unabhängigen) Werten



# Mittelwertfilter: Erstes Beispiel

$$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$H$



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	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

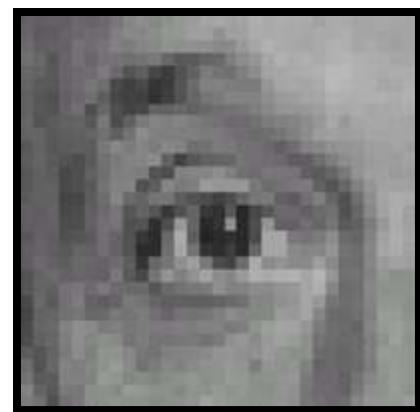
$G$

=

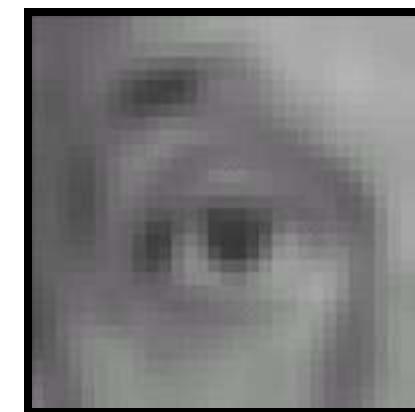
	0	10	20	30	30	30	20	10		
	0	20	40	60	60	60	40	20		
	0	30	60	90	90	90	60	30		
	0	30	50	80	80	90	60	30		
	0	30	50	80	80	90	60	30		
	0	20	30	50	50	60	40	20		
10	20	30	30	30	30	20	10			
10	10	10	0	0	0	0	0	0		

$F$

# Mittelwertfilter: Zweites Beispiel



$$\ast \frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} =$$



Original

Gefiltert

# Faltung: Grundidee

- Die **Faltung** zweier Funktionen  $h, g$  ergibt eine neue Funktion  $f$ 
  - *Vorstellung:*  $f$  ergibt sich als gewichtete Summe unterschiedlich verschobener Kopien von  $g$ . Die Gewichte sind die Werte von  $h$ .
    - *Beispiel:* “Verwackelte” Bildaufnahme
  - Häufig hat  $h$  einen kleineren Träger<sup>1</sup> als  $g$  und wird als **Faltungskern** bezeichnet.
    - Prinzipiell sind  $h$  und  $g$  jedoch austauschbar (Faltung ist *kommutativ*)!

$$h \quad * \quad g = f$$

<sup>1</sup> Erinnerung: Träger := Abgeschlossene Hülle der Nichtnullstellenmenge

# Definition der Faltung im kontinuierlichen Fall

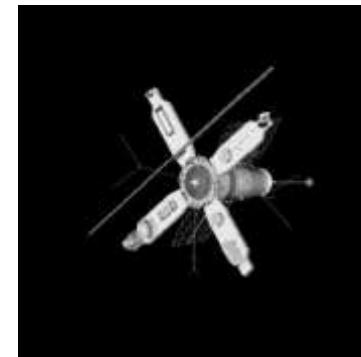
- Die Faltung zweier Funktionen  $g$  und  $h$  ist definiert durch

$$f(x) = (h * g)(x) = \int_{-\infty}^{\infty} h(\xi) \cdot g(x - \xi) d\xi$$

– Beachte: Negative  $\xi$  entsprechen einer Verschiebung von  $g$  nach links

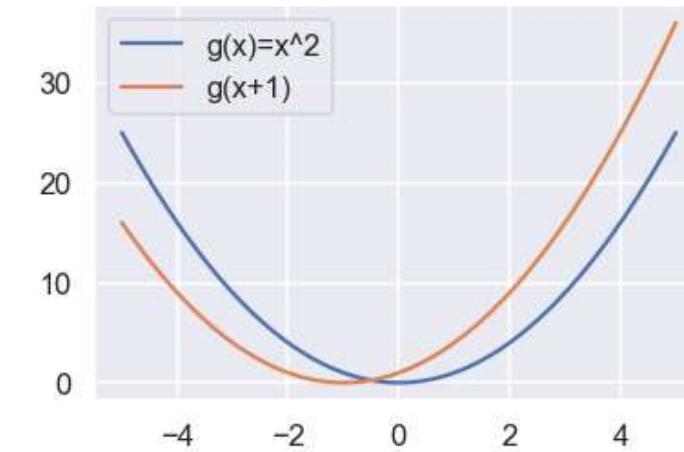


$h$



$g$

\*



- Die Filterung von 2D-Bildern erfordert 2D-Faltungen. In diesem Fall sind  $\mathbf{x}$  und  $\xi$  Vektoren:  $(h * g)(\mathbf{x}) = \iint h(\xi)g(\mathbf{x} - \xi) d\xi$

# Separierbare Faltung in 2D/3D

- Für **separierbare Kerne**  $h$  vereinfacht sich das 2D-Faltungsintegral

$$(h * g)(\mathbf{x}) = \iint h(\xi)g(\mathbf{x} - \xi)d\xi$$

zu

$$(h * g)(\mathbf{x}) = \iint h_1(\xi_1)h_2(\xi_2)g(x_1 - \xi_1, x_2 - \xi_2)d\xi$$

– Faktorisierung von  $h$  ermöglicht Auswertung durch zwei 1D-Integrale

- In der Bildverarbeitung nutzt man häufig  $h_1=h_2$
- Generalisiert entsprechend auch für 3D-Bilder

# Diskrete Faltung

- **Definition:** Faltung diskreter 1D-Funktionen  $h$  und  $g$ :

$$f(i) = (h * g)(i) = \sum_{u=-k}^k h(u) \cdot g(i - u)$$

– Hierbei ist  $h$  ein Faltungskern der Größe  $(2k + 1)$

- **Definition:** Faltung diskreter 2D-Funktionen  $h$  und  $g$ :

$$f(i, j) = (h * g)(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k h(u, v) \cdot g(i - u, j - v)$$

# Vergleich von Faltung und Kreuzkorrelation

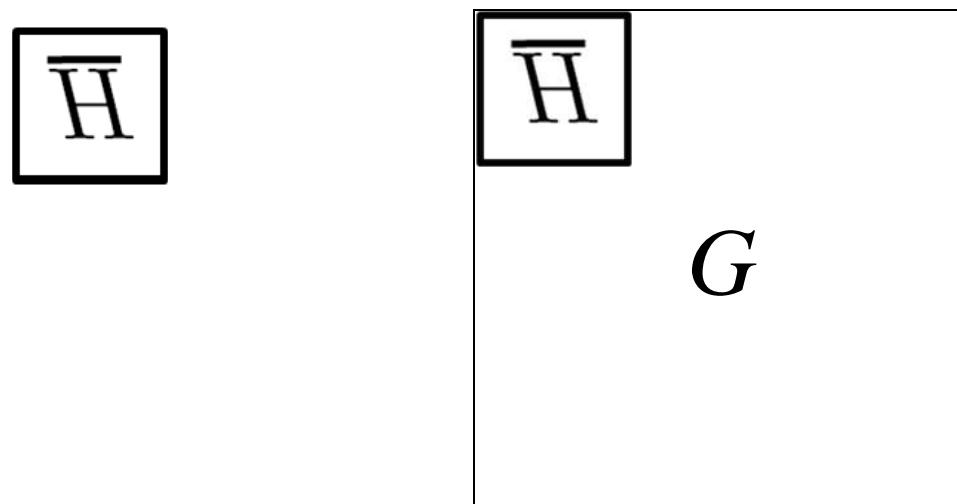
- Die in den ersten Beispielen verwendete **Kreuzkorrelation**

$$f(i, j) = (h \otimes g)(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k h(u, v) \cdot g(i + u, j + v)$$

entspricht einer Faltung mit gespiegeltem Kern  $h$ .

- Faltungen haben gegenüber Kreuzkorrelationen den Vorteil, dass sie *kommutativ* und *assoziativ* sind
- Für spiegelsymmetrische Kerne ergeben Faltung und Kreuzkorrelation identische Ergebnisse
  - *Beispiel:* Mittelwertfilter

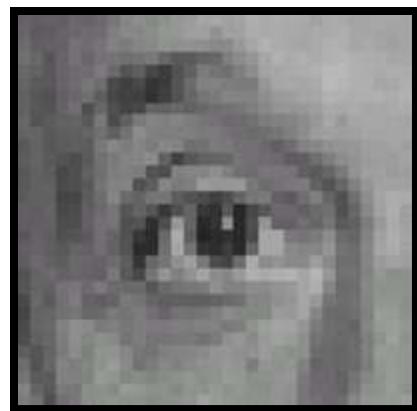
# Illustration: Faltung



# Randbedingungen

- **Problem:** Wenn wir eine Faltung am Bildrand auswerten wollen, ragt der Kern über das Bild hinaus
- **Lösungen:**
  - Wir schneiden den Rand ab (“gültiger” Teil der Faltung)
  - **Dirichlet-Randbedingung:** Wir nehmen jenseits des Bildrands feste Werte an (häufig Null/schwarz)
  - **Neumann-Randbedingung:** Wir nehmen an, dass entlang des Bildrands die Ableitung in Richtung der äußeren Normalen null ist
    - Entspricht einer Spiegelung der Werte am Bildrand
  - **Periodische Randbedingung:** Wir setzen das Bild in alle Richtungen periodisch fort
    - Ergibt sich bei der Berechnung mittels Faltungstheorem, sehen wir später

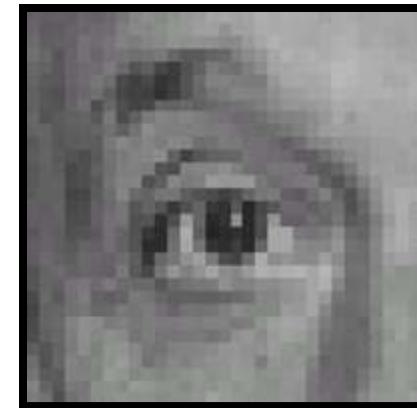
# Faltungsbasierte Filterung: Beispiel 1



\*

0	0	0
0	1	0
0	0	0

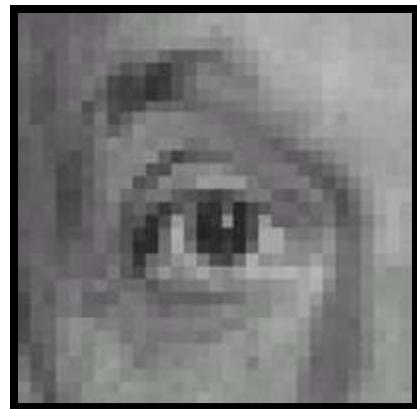
=



Original

Unverändertes Bild

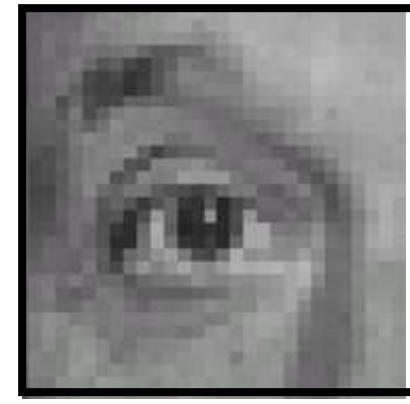
# Faltungsbasierte Filterung: Beispiel 2



\*

0	0	0
1	0	0
0	0	0

=



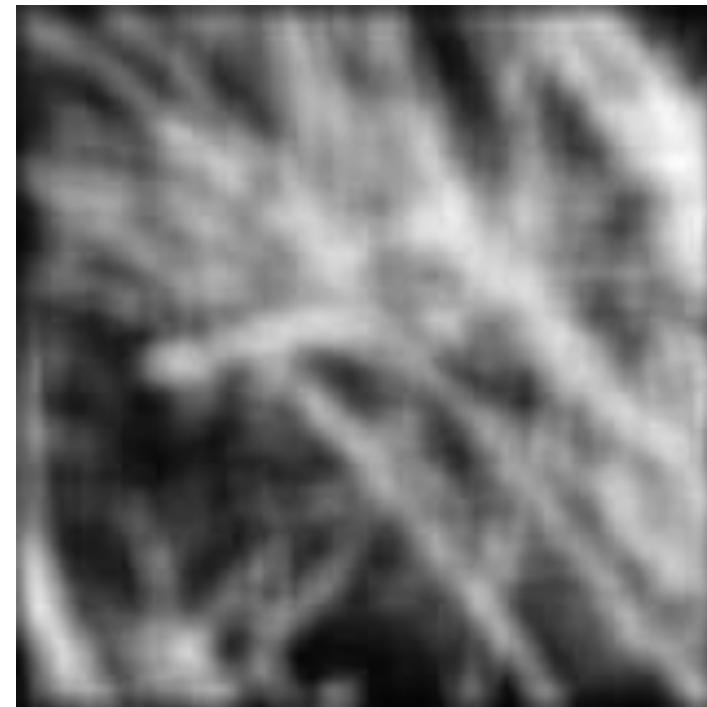
Original

Um 1 Pixel  
nach links verschoben

# Nachteil des Mittelwert-Filters



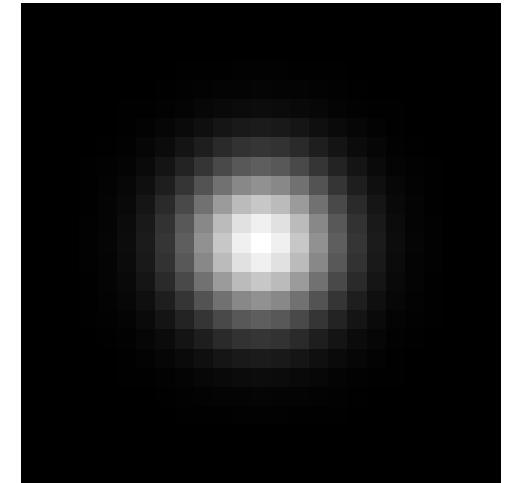
$$\ast \quad \square =$$



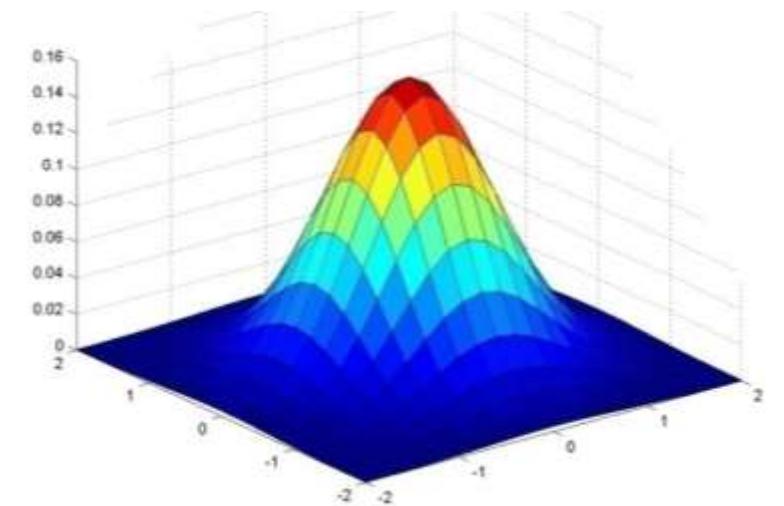
# Filterung mit Gauss-Kern

- Eine rotationssymmetrische Tiefpass-Filterung erreicht man durch Filterung mit dem **Gauss-Kern**

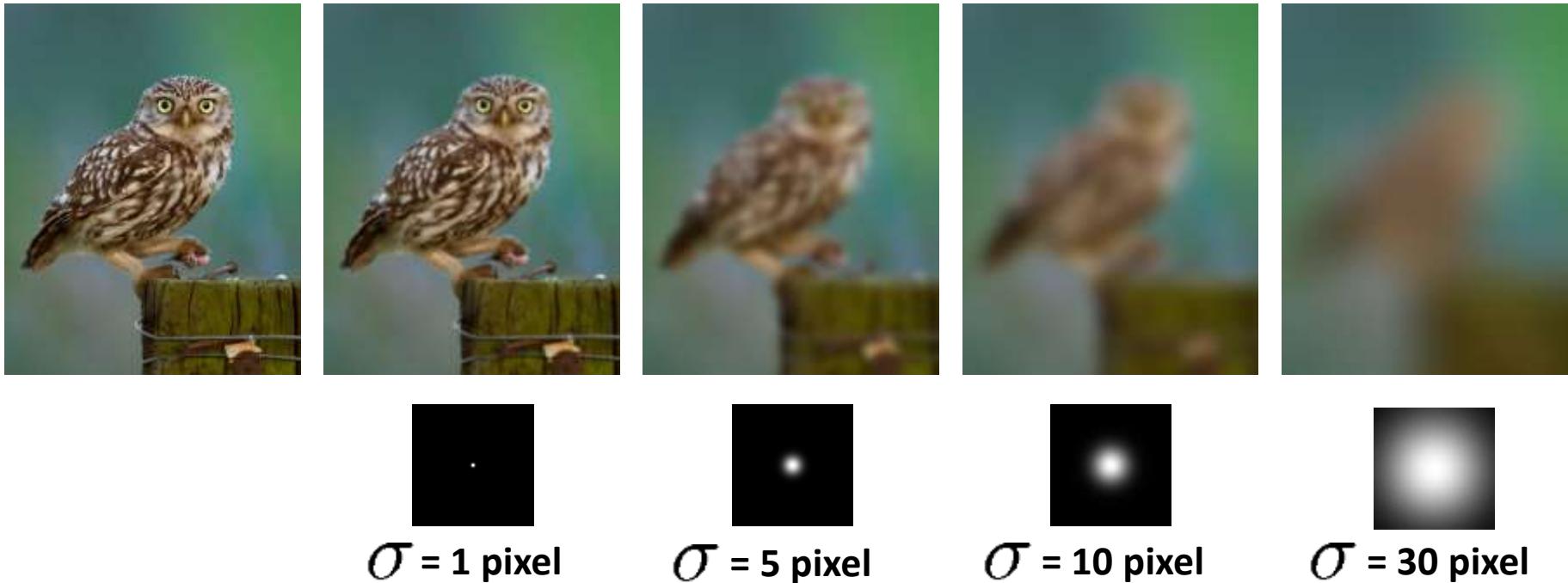
$$G_{\sigma} = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$



- Dämpft hohe Ortsfrequenzen
- Um Rechenzeit zu sparen schneidet man den Filter meist ab, z.B. nach  $3\sigma$
- **Quiz:** Wie wirkt sich ein höheres  $\sigma$  auf das gefilterte Bild aus?

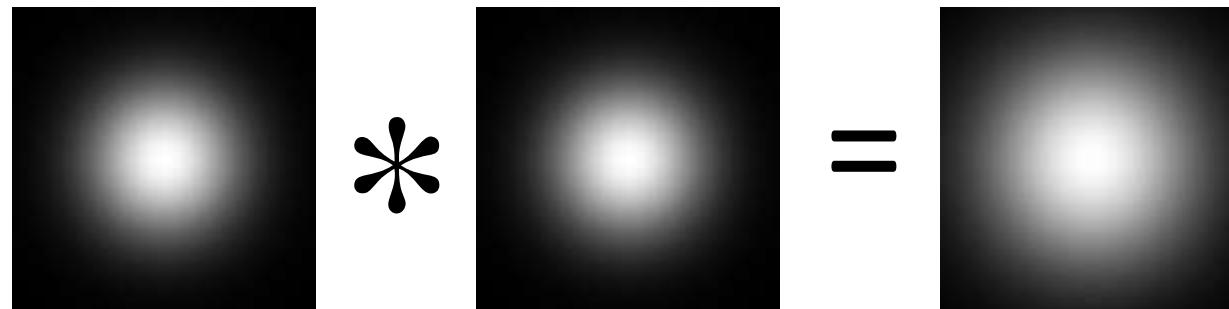


# Gauss-Filter: Effekt der Bandbreite



# Eigenschaften des Gauss-Filters

- Faltung zweier Gauss-Verteilungen mit Standardabweichungen  $\sigma_1$  und  $\sigma_2$  ergibt einen Gauss mit Std.-Abweichung  $\sqrt{\sigma_1^2 + \sigma_2^2}$



- Zusammen mit der Assoziativität der Faltung ergibt sich daraus, dass wiederholtes Gauss-Filtern dasselbe Ergebnis liefert wie ein einmaliges Filtern mit einem entsprechend breiteren Gauss

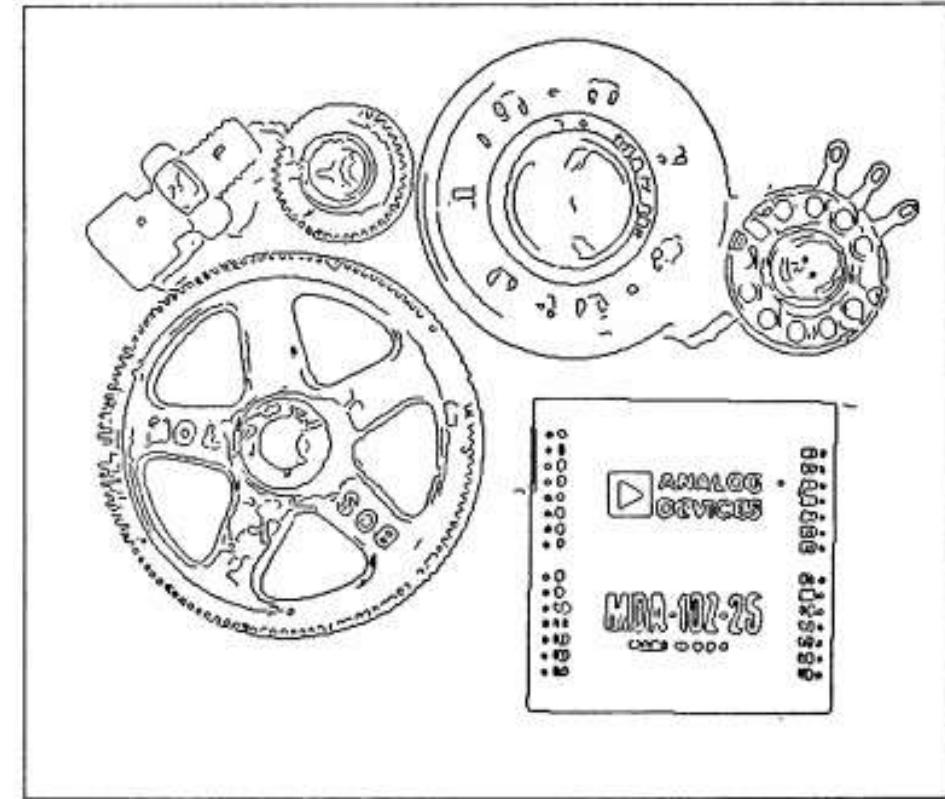
# Zusammenfassung: Lineare Bildfilterung

- **Lineare Bildfilter** lassen sich durch **Kreuzkorrelationen** oder **Faltungen** ausdrücken
  - *Unterschiede:*
    - Spiegelung des Kerns
    - Faltung ist assoziativ und kommutativ
  - Matrixdarstellung von Faltungskernen
- Beliebte lineare Filter zur **Entrauschung** sind
  - Mittelwert-Filter
  - Gauss-Filter

# **1.5 Kantenerkennung**

# Motivation: Kantenerkennung

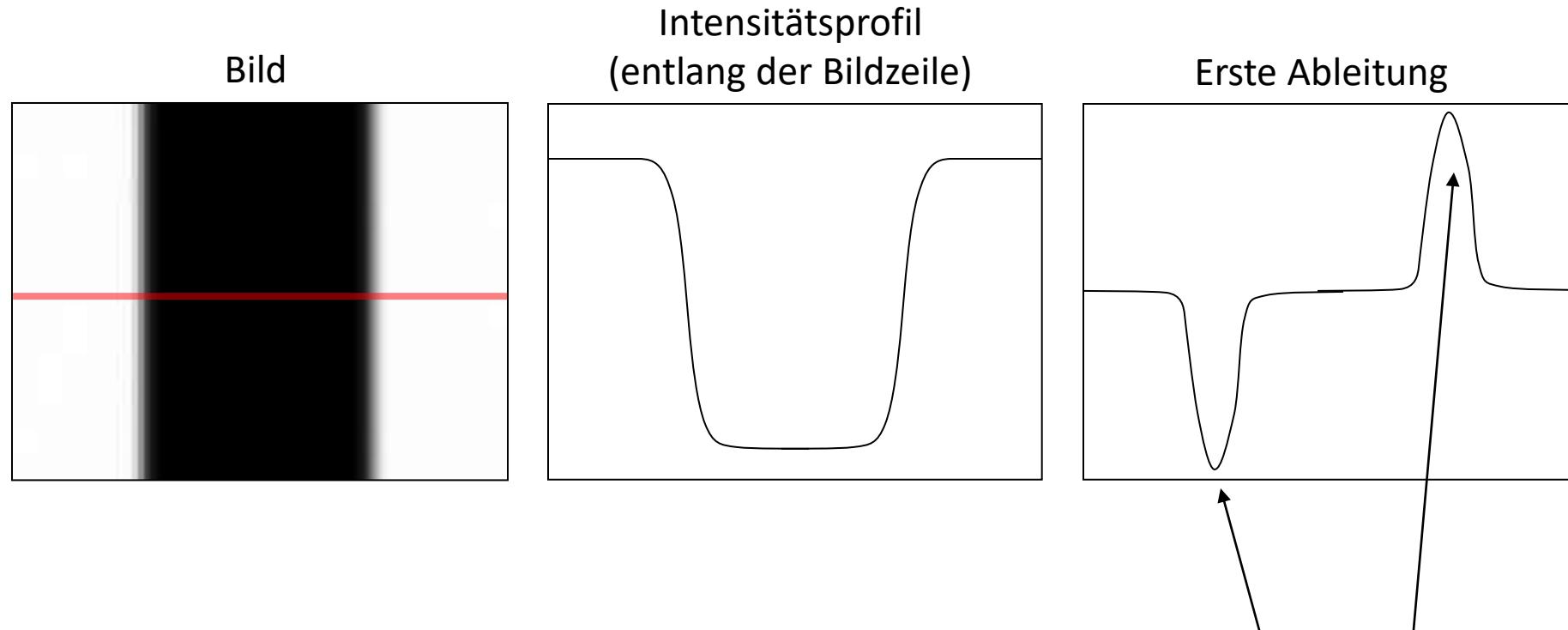
*Image from: John Canny, "A Computational Approach to Edge Detection" IEEE Trans. on Pattern Analysis and Machine Intelligence 8(6):679-698, 1986*



- **Ziel der Kantenerkennung:** Extrahiere aus einem 2D-Bild eine Menge von Kurven entlang kontraststarker Kanten
  - *Bedeutung:* Kanten folgen häufig dem Umriss von Objekten oder anderen wichtigen Bildinhalten

# Was macht eine Kante aus?

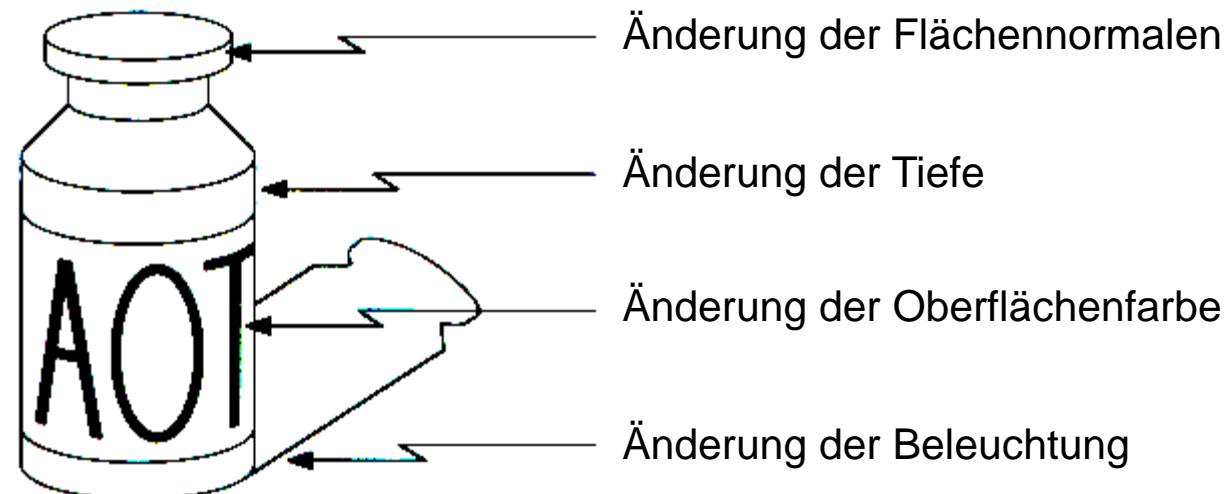
- Kanten erkennt man daran, dass sich die Bildintensität senkrecht zu ihnen schnell verändert



Kanten lokalisieren wir dort,  
wo die erste Ableitung extremal ist

# Woher kommen Kanten?

- Kanten entstehen durch verschiedene Faktoren:



# Ableitungen in Bildern

- Wie können wir Ableitung in einem *digitalen* Bild  $F[x,y]$  bilden?
  - Option 1: Rekonstruktion eines kontinuierlichen Bildes  $f$ , für das wir herkömmliche Ableitungen berechnen können
  - Option 2: Berechnung diskreter Ableitungen (finiter Differenzen)

$$\frac{\partial f}{\partial x}[x, y] \approx F[x + 1, y] - F[x, y]$$

Wie würde man das als Faltungskern schreiben?

$$\frac{\partial f}{\partial x} : \begin{array}{|c|c|c|}\hline & & \\ \hline \end{array}$$

$H_x$

$$\frac{\partial f}{\partial y} : \begin{array}{|c|c|c|}\hline & & \\ \hline \end{array}$$

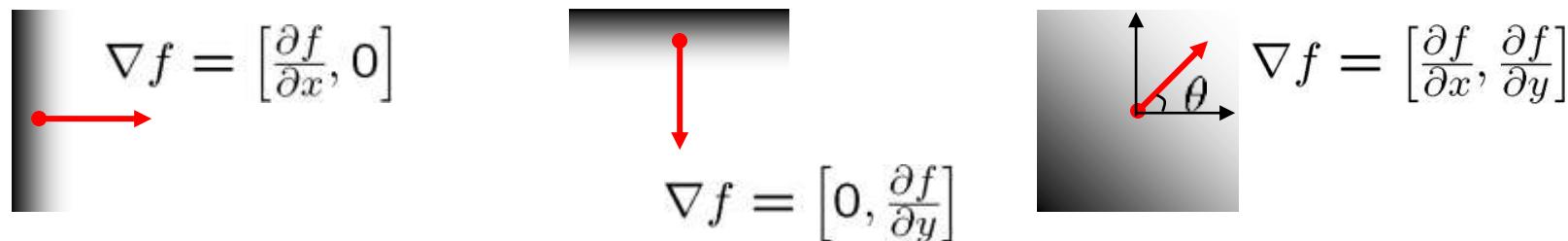
$H_y$

*Hinweis:* Wir definieren die Ecke oben links als Ursprung (Matrix-Notation)

# Bildgradienten

- Der *Gradient* eines Bildes ist definiert als  $\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$

Der Gradient weist in die Richtung der steilsten Intensitätsveränderung



Die Norm des Gradienten gibt die *Kantenstärke* an:

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

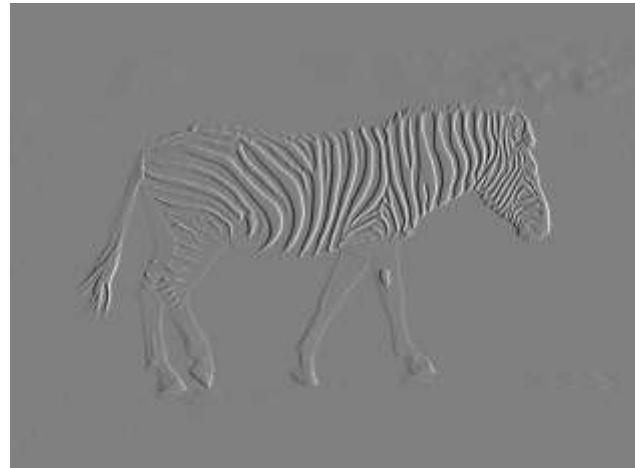
Die Richtung des Gradienten ist  $\theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$

- Frage: In welcher Richtung verläuft die Kante?

# Illustration: Bildgradient



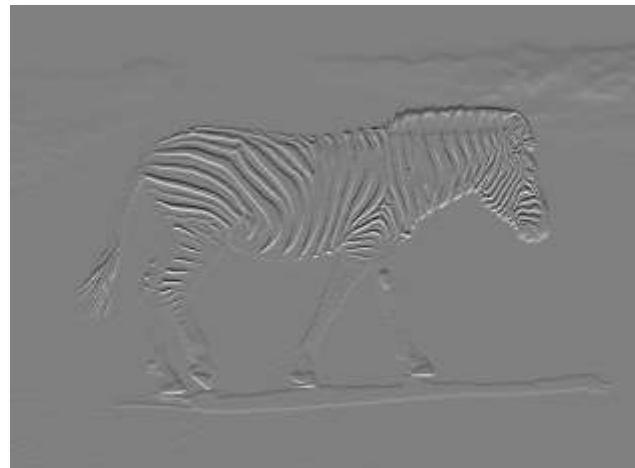
$f$



$\frac{\partial f}{\partial x}$

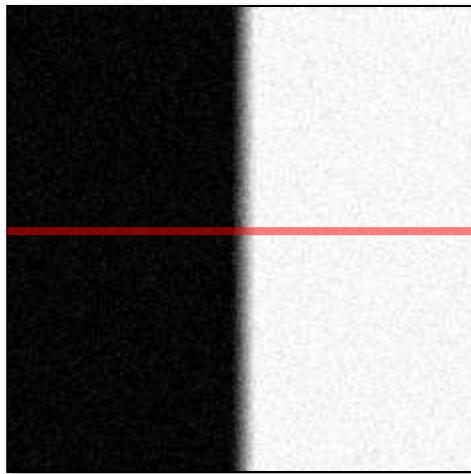


$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$



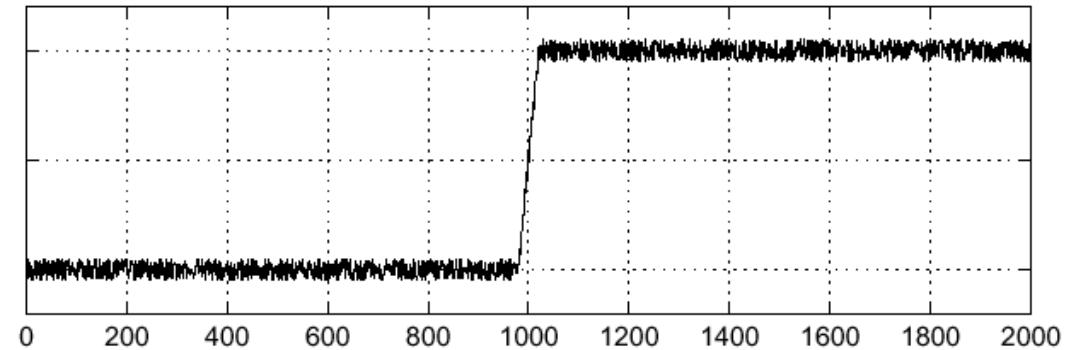
$\frac{\partial f}{\partial y}$

# Auswirkung von Rauschen

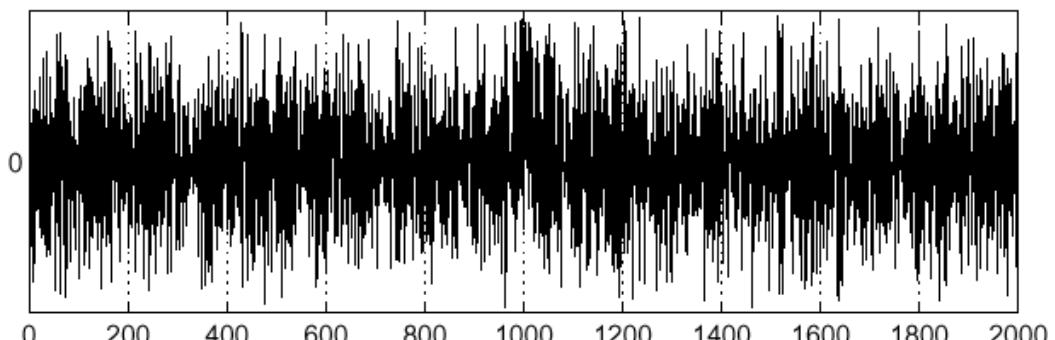


Verrausches Eingabebild

$$f(x)$$

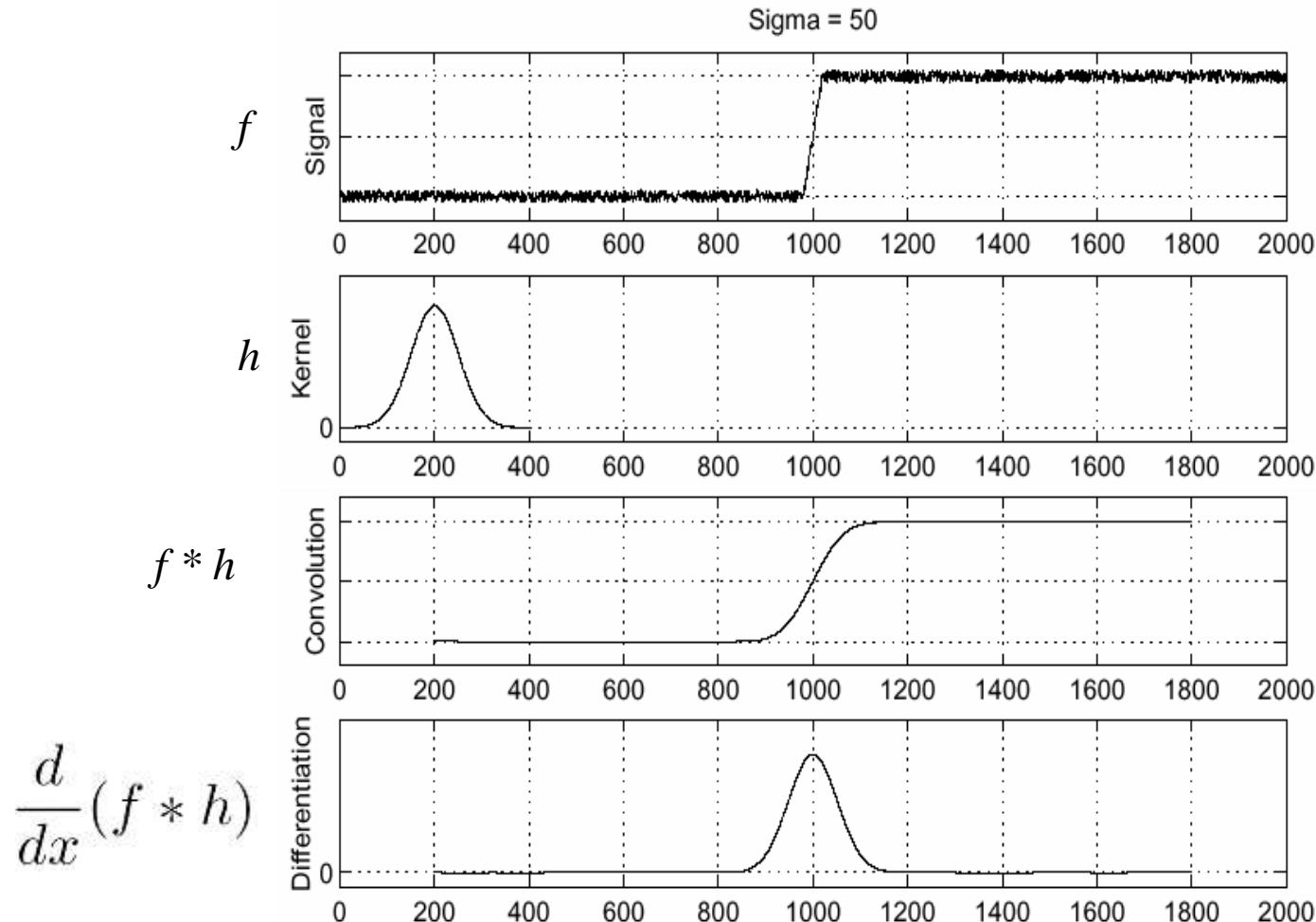


$$\frac{d}{dx}f(x)$$



Wo ist die Kante?

# Lösung: Vor Kantenerkennung immer glätten!



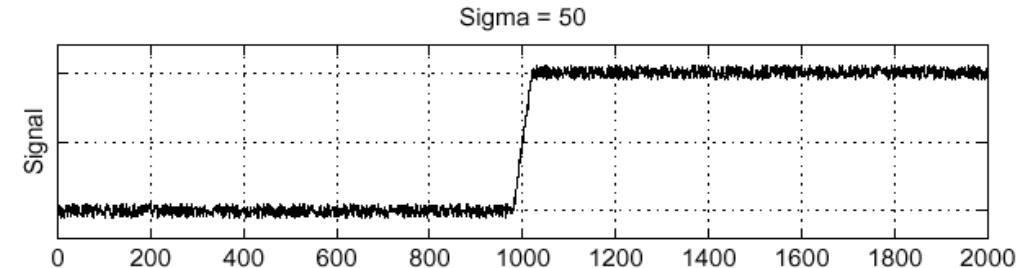
Wir bestimmen Kanten als Extrema von  $\frac{d}{dx}(f * h)$

# Glättung und Ableitung in Einem

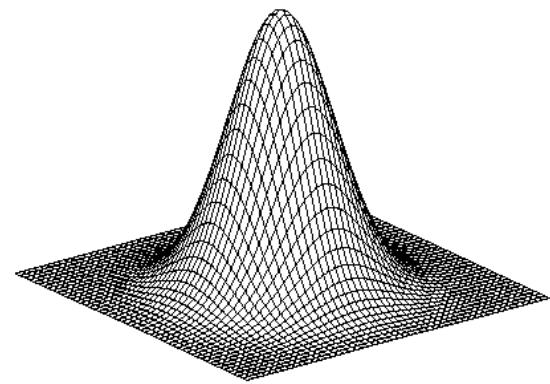
- Wir wissen bereits, dass
  1. Ableitungen als Faltungen implementiert werden können
  2. Faltungen assoziativ und kommutativ sind

$$\frac{d}{dx}(f * h) = f * \frac{d}{dx}h$$

- Damit können wir beide Operationen kombinieren:

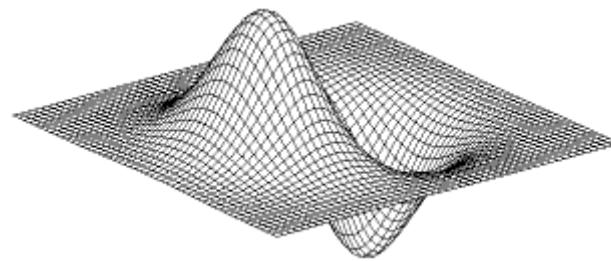


# Kantenerkennung in zwei Dimensionen



Gauss

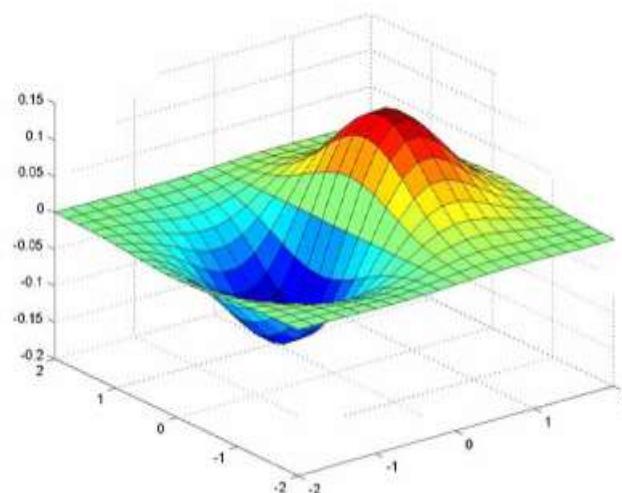
$$h_{\sigma}(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



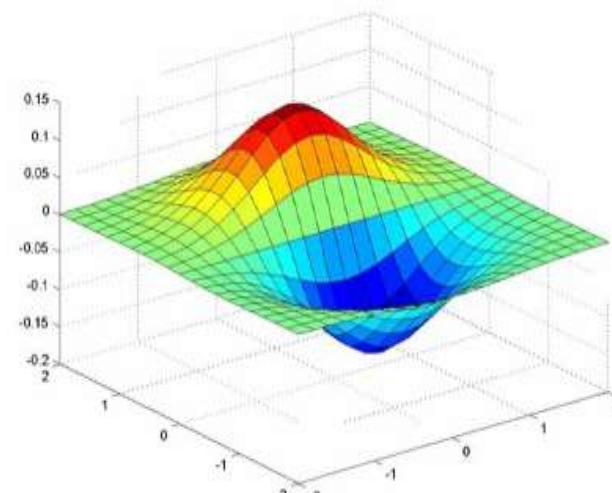
Ableitung des Gauss (x)

$$\frac{\partial}{\partial x} h_{\sigma}(u, v)$$

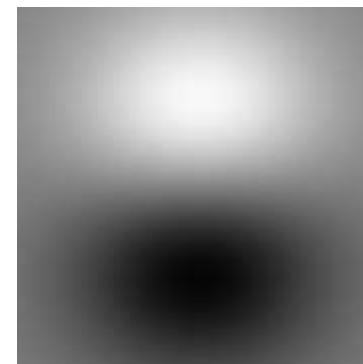
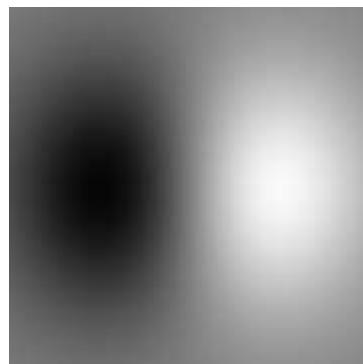
# Ableitungen des Gauss-Filters



x-Richtung



y-Richtung



# Der Sobel-Operator

- Aufgrund seines großen Trägers ist der Gauss-Filter rechenaufwendig. Der **Sobel-Operator** ist eine beliebte und relative günstig zu berechnende Approximation seiner Ableitung:

$$\frac{1}{8} \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 2 & 0 & -2 \\ \hline 1 & 0 & -1 \\ \hline \end{array}$$

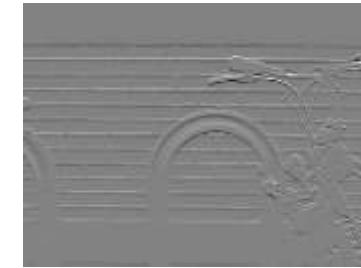
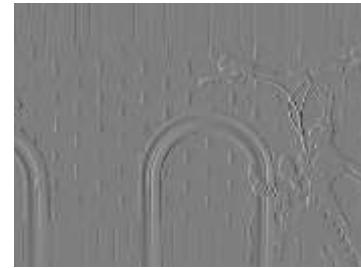
$s_x$

$$\frac{1}{8} \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

$s_y$

- *Anmerkung:* Häufig findet man Definitionen ohne den Faktor 1/8
  - Ohne diesen Faktor haben die berechneten Bildgradienten eine zu hohe Norm
  - Der Faktor 1/8 geht von einem Abstand von eins zwischen den Pixeln aus
  - Konstante Faktoren sind bei der Kantenerkennung jedoch häufig vernachlässigbar

# Sobel-Operator: Beispiel



Source: Wikipedia

# Grundlage der Kantenerkennung: Der Gradient

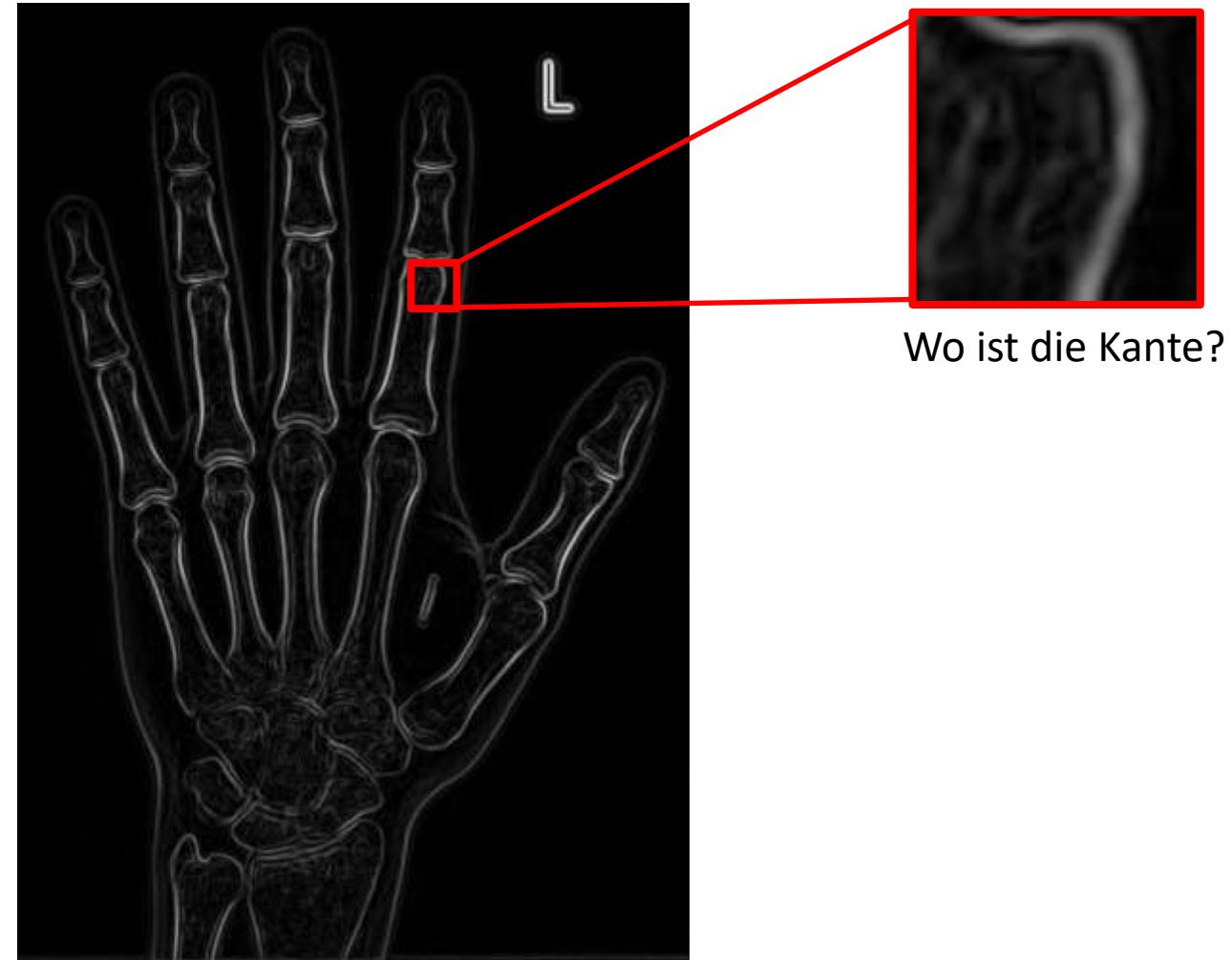


Eingabebild



Norm der Gradienten

# Problem: Genaue Lokalisierung der Kante

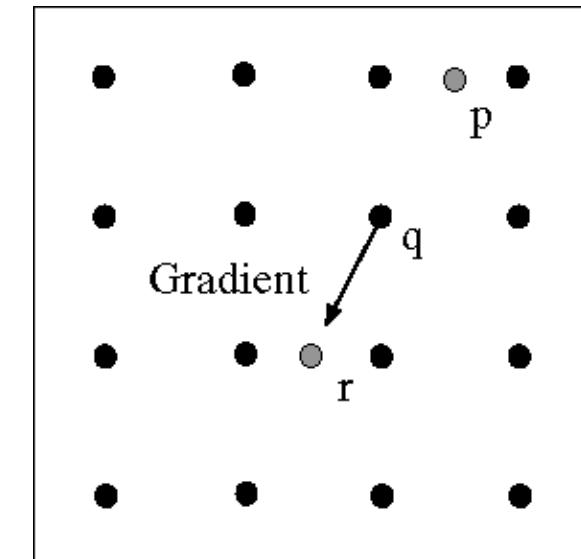
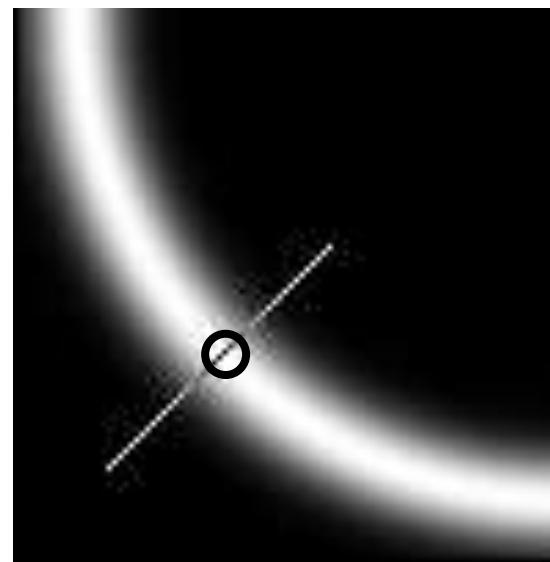
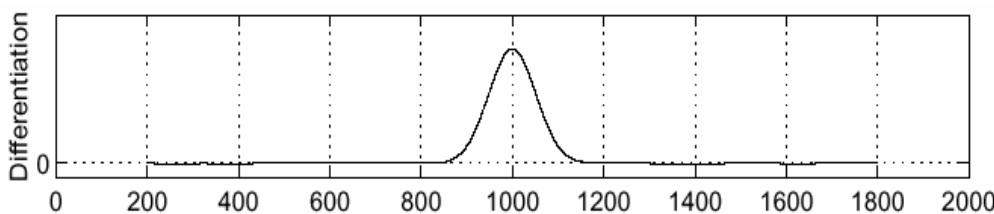


Wo ist die Kante?

Norm der Gradienten

# Lösung: Unterdrückung der Nicht-Maxima

- Die **Nicht-Maxima-Unterdrückung** setzt Pixel, an denen die Gradientennorm entlang der Gradientenrichtung kein lokales Maximum ist auf Null
  - Erfordert eine Interpolation an den Punkten p und r
  - Englisch „non maximum suppression“

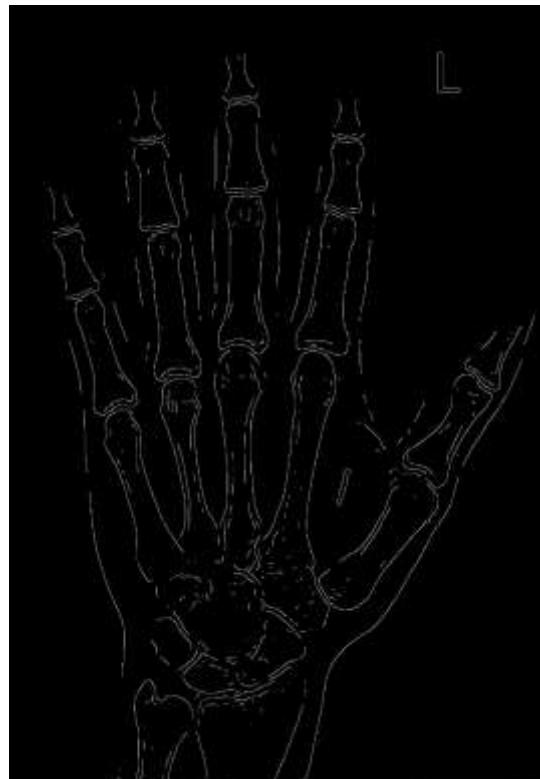


# Letzter Schritt: Verkettung und Schwellenwertbildung

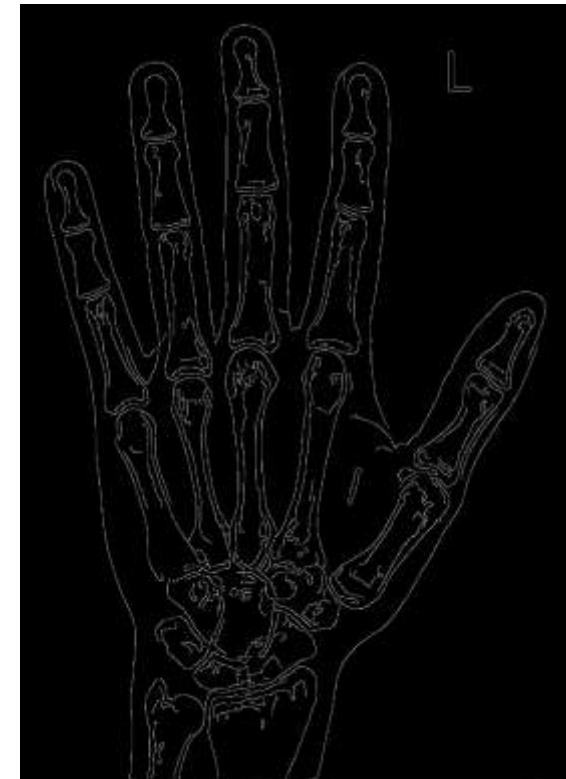
- Canny's Algorithmus **verkettet** Pixel zu Kanten (*linking*) und nutzt dabei **zwei Schwellenwerte (Hysterese)**:
  - Höherer Wert zum Start einer neuen Kante, kleinerer Wert zur Fortsetzung



Skelettierung (unterer Wert)



Skelettierung (oberer Wert)



Kombination

# Canny-Kantenerkennung: Rolle von $\sigma$



Original



Canny mit  $\sigma = 2$



Canny mit  $\sigma = 3$

- Canny's Algorithmus berechnet Gradienten durch Faltung mit der Ableitung von Gauss-Kernen der Bandbreite  $\sigma$ 
  - großes  $\sigma$  erfasst gröbere Kanten (“large-scale” / “hohe Skala”)
  - kleines  $\sigma$  erfasst feinere Kanten (“small-scale” / “niedrige Skala”)

# Zusammenfassung: Canny-Algorithmus zur Kantenerkennung

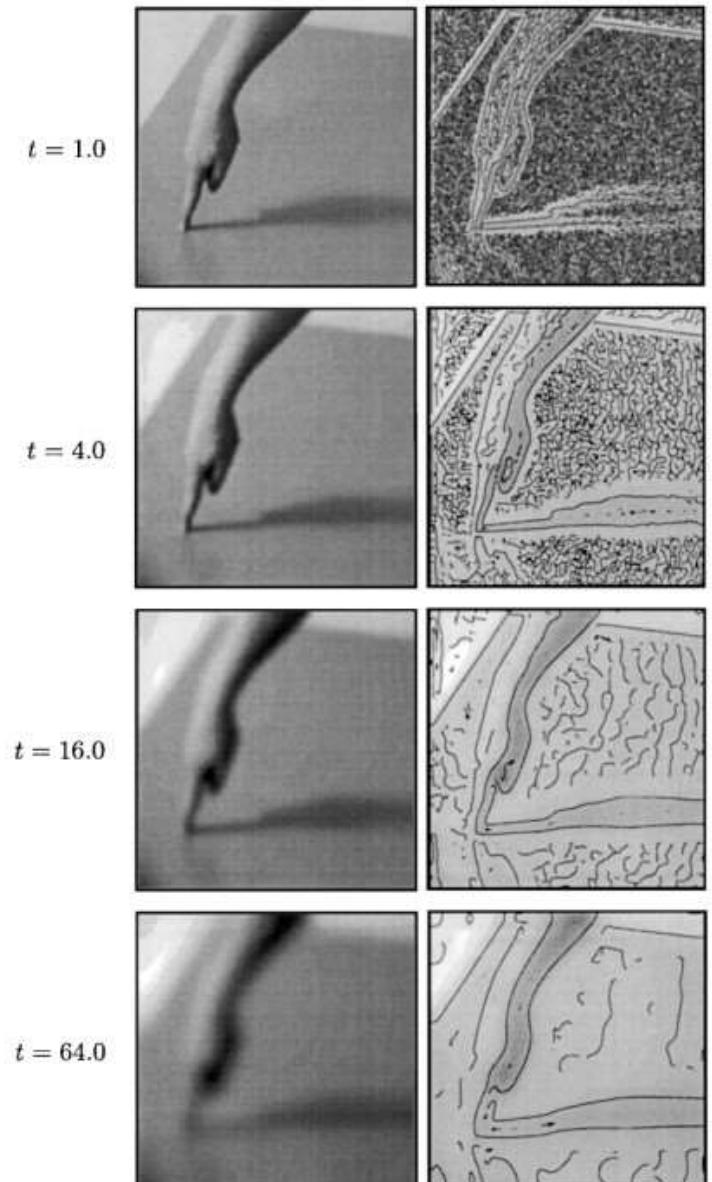


Die Schritte des **Canny-Algorithmus'** sind:

1. Filterung des Bildes mit Ableitungen des Gauss-Kerns
  - Skalen-Parameter  $\sigma$
2. Berechnung von Stärke und Richtung des Gradienten
3. Unterdrückung von nicht-Maxima
4. Verkettung und Filterung
  - Kombination eines oberen und unteren Schwellenwerts der Kantenstärke

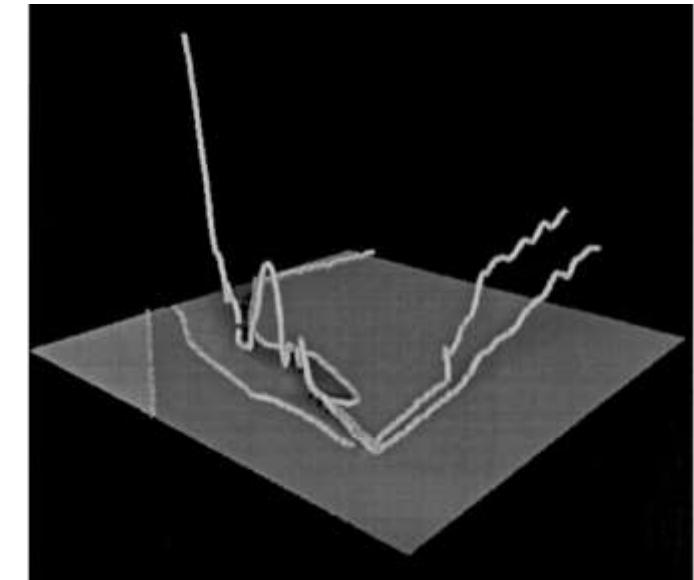
# Grundidee der Skalenraumanalyse

- **Problem:** Bilder enthalten häufig Strukturen verschiedener Größe, die mit einem festen  $\sigma$  nicht adäquat erkannt werden
- **Grundidee der Skalenraumanalyse:** Untersuche die Familie aller möglichen geglätteten Bilder
  - $\{T_t f \mid t \geq 0\}$  mit Ausgangsbild  $f$ , Glättungsoperator  $T_t$ , Glättungsparameter  $t$
  - Häufige Wahl von  $T_t$ :  
Gauss-Glättung mit  $\sigma = \sqrt{2t}$

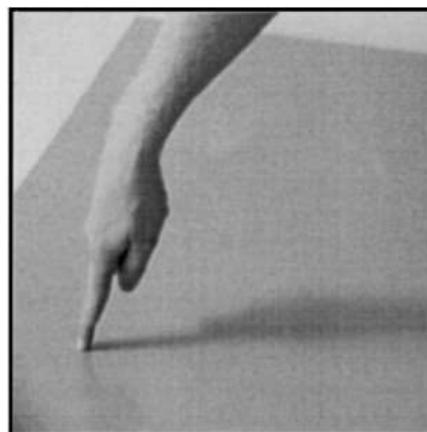


# Details: Kantenerkennung im Skalenraum

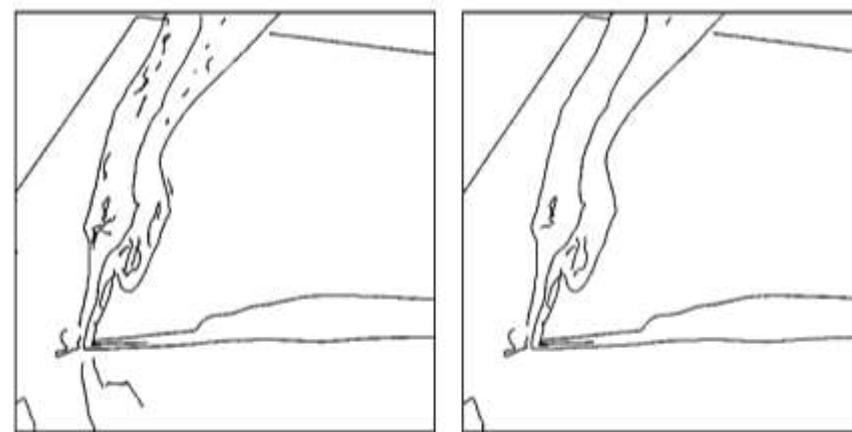
- Die Kanten aller Skalen zusammengenommen ergeben im (2D+ $t$ ) Skalenraum zunächst Flächen
- Auf diesen Flächen wählen wir lokale Maxima der **Kantenstärke**  $\sqrt{t} \|\nabla(T_t f)\|^2$  entlang der  $t$ -Achse aus
  - Faktor  $\sqrt{t}$  kompensiert Kontrastabschwächung



Kanten im Skalenraum



Ausgangsbild



2D-Projektionen der 50/20/10 stärksten Kanten

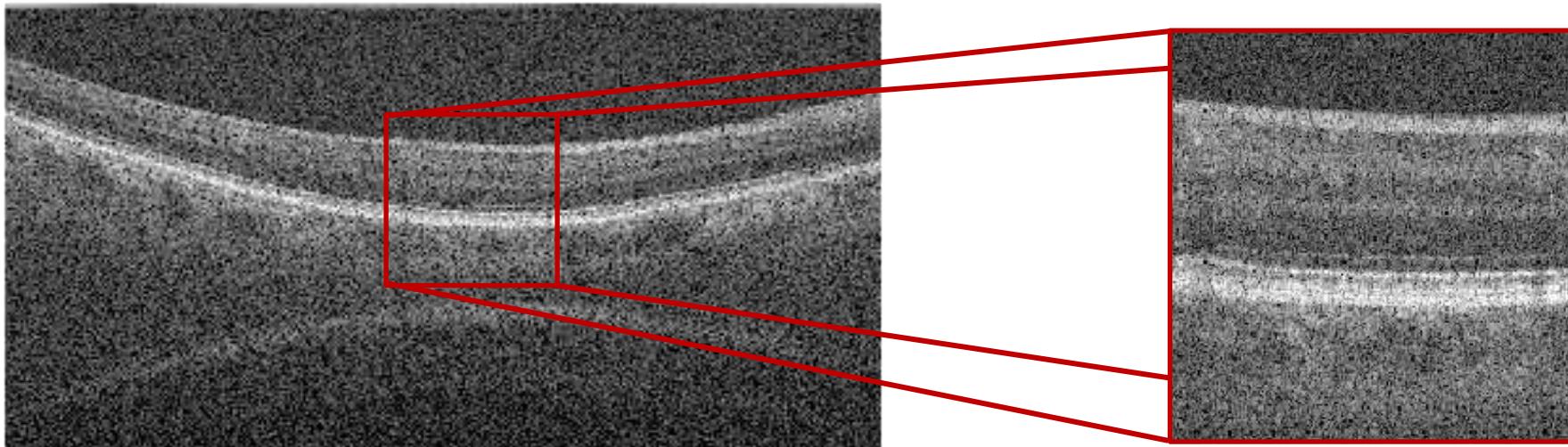
# Zusammenfassung: Kantenerkennung

- **Kanten** sind plötzliche Änderungen der Bildintensität
- Wir berechnen sie über den **Gradienten**, konkret mittels
  - Finiten Differenzen,
  - Faltung mit Ableitungen des Gauss-Kerns oder
  - Sobel-Operator
- Der **Canny-Algorithmus** zur Kantenerkennung ist weit verbreitet
  - Als Nutzer kann man gewünschte Skala und Stärke einstellen
- Eine **Skalenraumanalyse** betrachtet das Bild auf allen Skalen gleichzeitig und wählt pixelweise die passende Skala aus

## **1.6 Nichtlineare Bildfilterung**

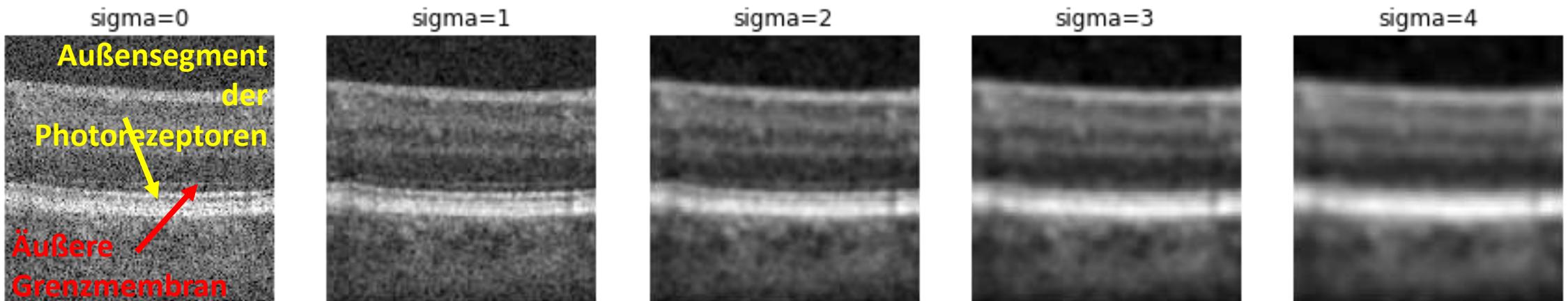
# Motivation

- **Nichtlineare Bildfilter** ermöglichen es beim Entfernen von Rauschen relevante Bildstrukturen zu erhalten, insbesondere Kanten
- *Beispiel:* Querschnitt der Retina, aufgenommen mittels Optischer Kohärenz-Tomographie (OCT)



# Warum Gauss-Glättung oft nicht reicht

- **Problem:** Gauss-Glättung entfernt nicht nur Rauschen, sondern auch relevante Bildstrukturen
  - Gewichte des Filterkerns sind in jeder Richtung gleich
  - In der Nähe von Kanten mitteln wir über Pixel, die verschiedene Strukturen zeigen
  - Wenn wir wüssten, welche Pixel zu welchem Objekt gehören könnten wir das vermeiden – wissen wir aber meist nicht!



# Bilaterale Filterung

- Der **bilaterale Filter** mittelt nur Pixel, die sowohl räumlich nah beieinander sind, als auch ähnliche Intensitäten haben

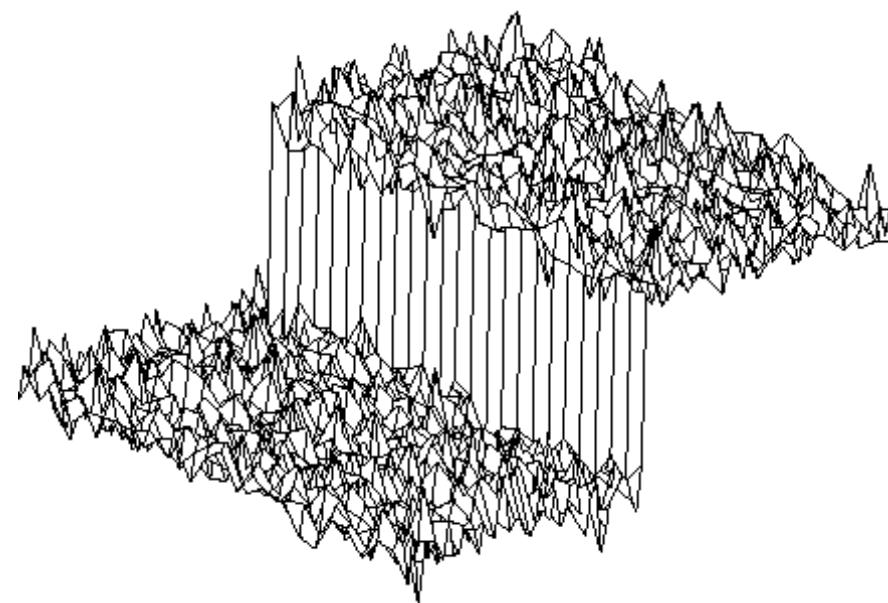
- Separate Parameter  $\sigma_c, \sigma_s$  steuern entsprechende Gewichte

$$c(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{\sigma_c^2}\right), s(g(\mathbf{x}), g(\mathbf{x}')) = \exp\left(-\frac{|g(\mathbf{x}) - g(\mathbf{x}')|^2}{\sigma_s^2}\right)$$

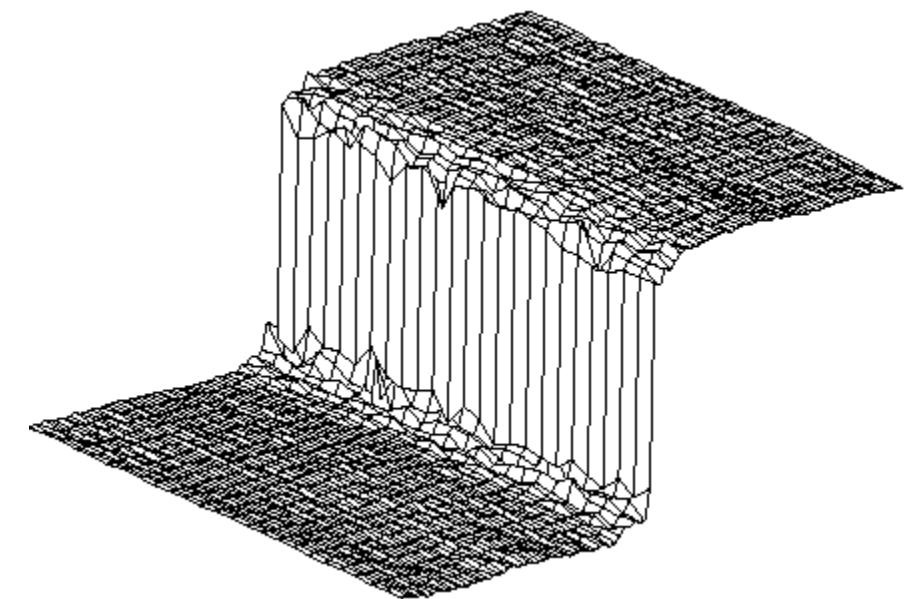
- Das Ergebnis  $f(\mathbf{x})$  einer bilateralen Filterung von  $g(\mathbf{x})$  in einer gegebenen Nachbarschaft  $\omega_{\mathbf{x}}$  um jeden Punkt  $\mathbf{x}$  ergibt sich durch Multiplikation beider Gewichte und entsprechende Normierung:

$$f(\mathbf{x}) = \frac{\sum_{\mathbf{x}' \in \omega_{\mathbf{x}}} g(\mathbf{x}') c(\mathbf{x}, \mathbf{x}') s(g(\mathbf{x}), g(\mathbf{x}'))}{\sum_{\mathbf{x}' \in \omega_{\mathbf{x}}} c(\mathbf{x}, \mathbf{x}') s(g(\mathbf{x}), g(\mathbf{x}'))}$$

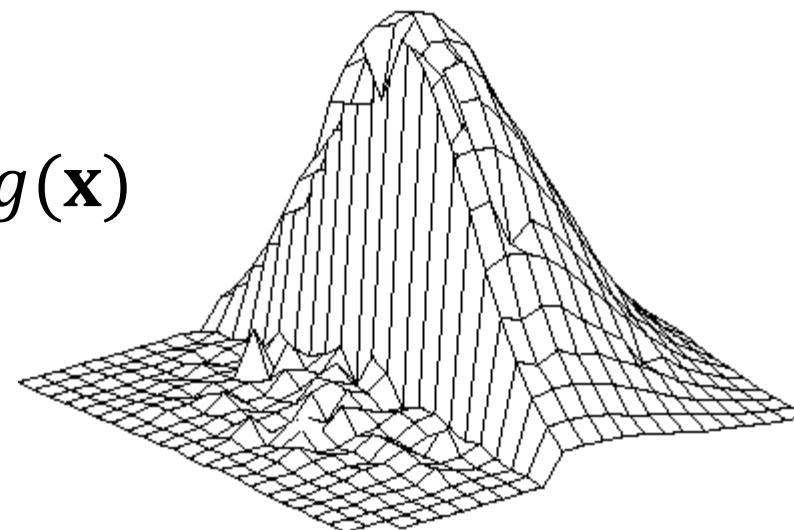
# Bilaterale Filterung: Illustration



Verrausches Signal  $g(\mathbf{x})$



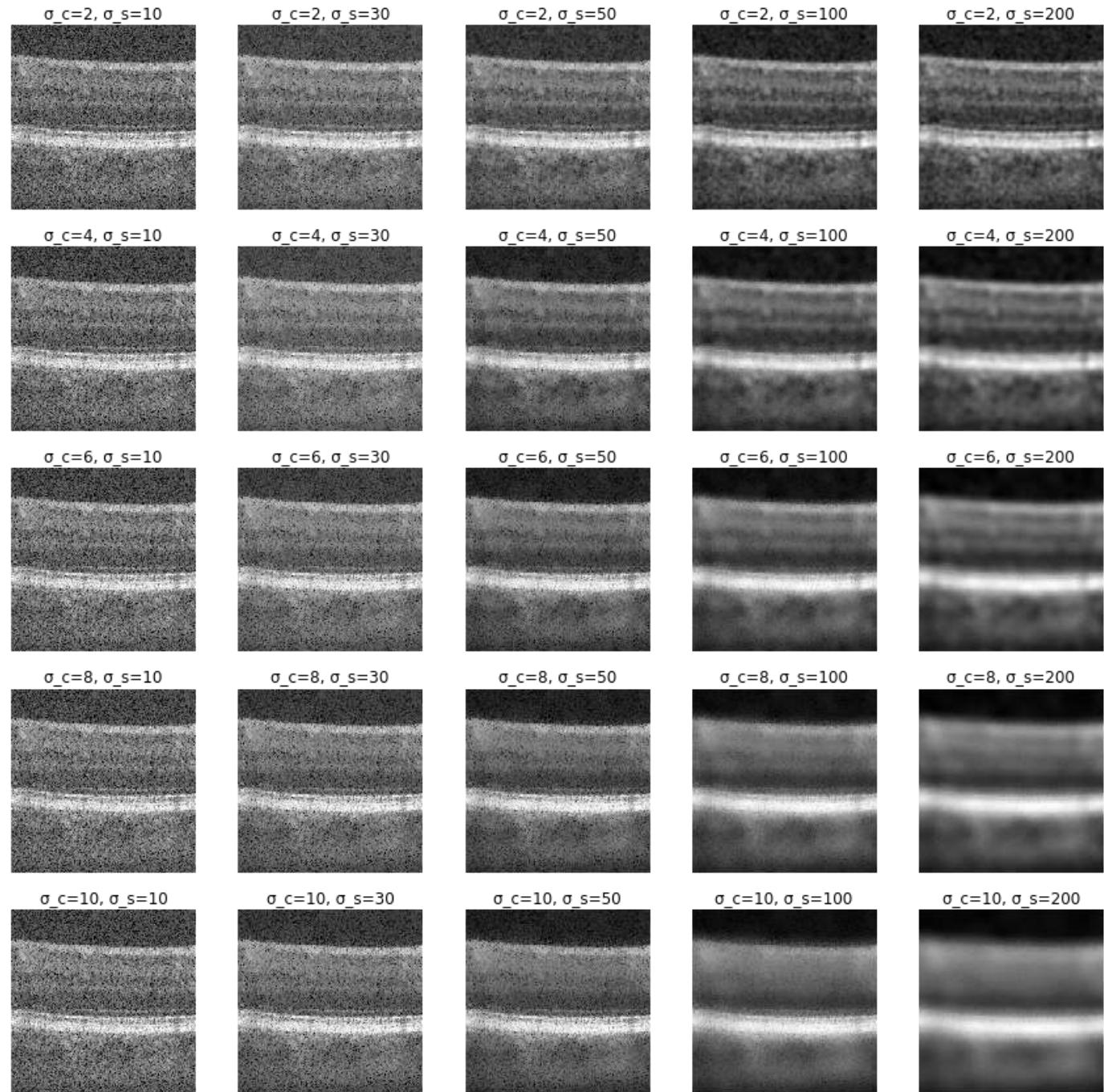
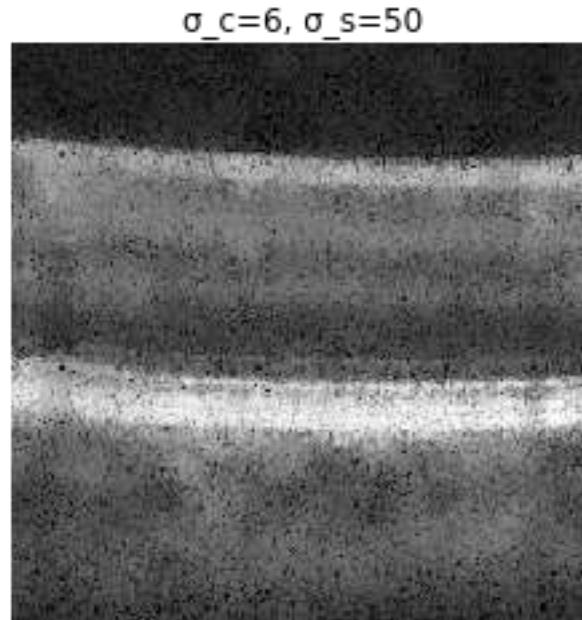
Gefiltertes Signal  $f(\mathbf{x})$



Bilaterales Gewicht  $c(\mathbf{x}, \mathbf{x}') \times s(g(\mathbf{x}), g(\mathbf{x}'))$

# Bilateraler Filter: Parameter-Wahl

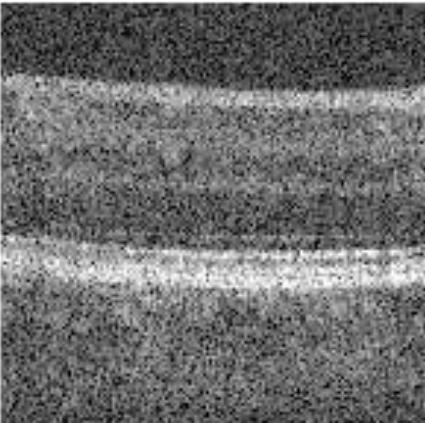
- Was, wenn  $\sigma_s$  hohe Werte annimmt?
- Wie würden Sie die Parameter hier setzen?



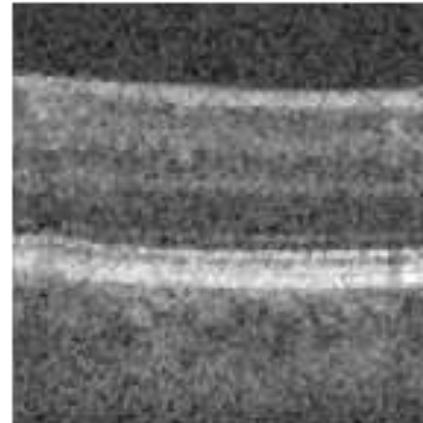
# Median-Filter

- **Bemerkung:** Pixel, die viel heller oder dunkler sind als ihre Umgebung, werden von bilateralen Filtern nicht entrauscht
  - *Beispiel: Specklemuster*, ein Interferenzphänomen, das in manchen Modalitäten (z.B. OCT) auftritt
- **Median-Filters** entfernen solche Ausreißer, indem sie jeden Pixel durch den Median einer lokalen Nachbarschaft ersetzen

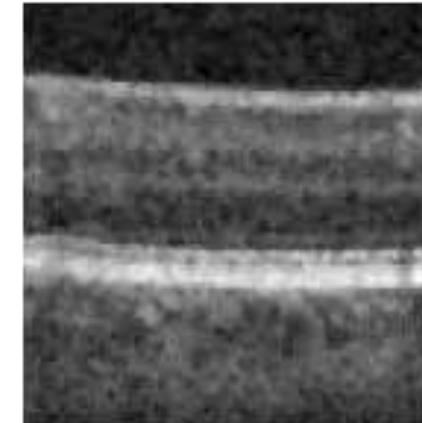
size=1



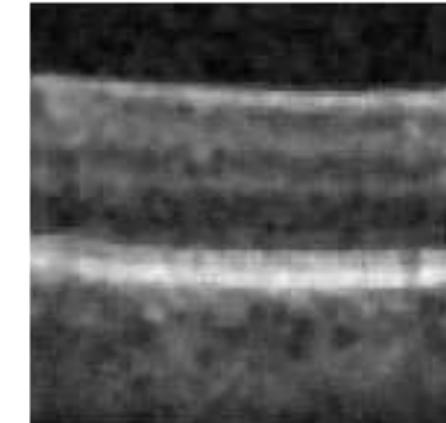
size=3



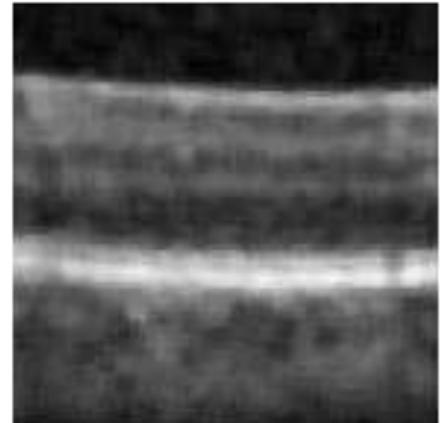
size=5



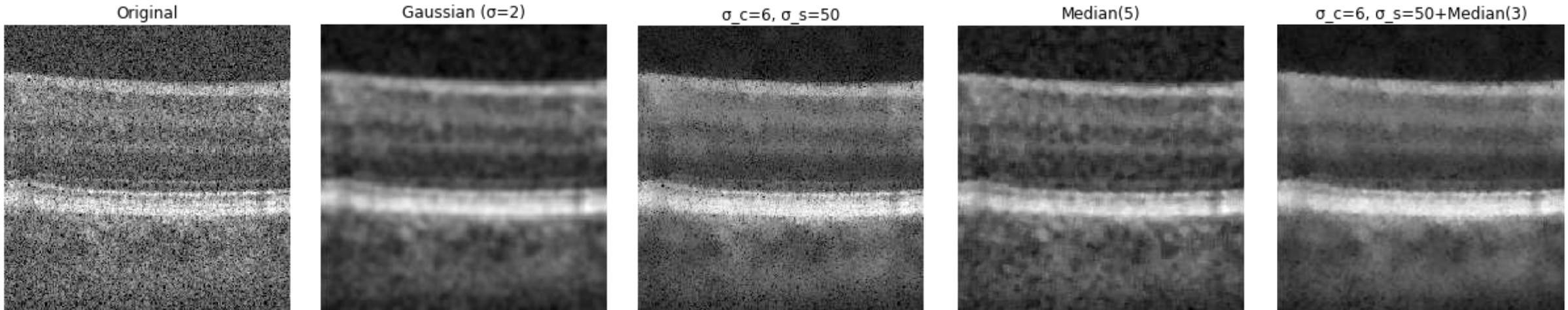
size=7



size=9



# Visueller Vergleich verschiedener Filter



- *Hinweis:* In der Praxis kann es sinnvoll sein mehrere Filter zu kombinieren, z.B. durch einen Median-Filter Ausreißer zu entfernen, die nach einer bilateralen Filterung verbleiben

# Zusammenfassung: Nichtlineare Bildfilter

- Im Gegensatz zu linearen Filtern haben **nichtlineare Bildfilter** die Möglichkeit sich an Bildinhalte anzupassen
- In der Praxis beliebte nichtlineare Filter sind u.a.
  - **Bilaterale Filter**, die eine kantenerhaltende Filterung ermöglichen
  - **Median-Filter** zur Beseitigung von Ausreißern

# Zum Nach- und Weiterlesen

- Heinz Handels: Medizinische Bildverarbeitung.  
Vieweg+Teubner, 2. Auflage, 2009
- Ramesh Jain, Rangachar Kasturi, Brian G. Schunck: *Machine Vision*. McGraw-Hill 1995
- John Canny: *A Computational Approach to Edge Detection*.  
IEEE Trans. on Pattern Analysis and Machine Intelligence  
8(6):679-698, 1986
- Carlo Tomasi, Roberto Manduchi: *Bilateral Filtering for Gray and Color Images*. In: Proc. Int'l Conf. on Computer Vision (ICCV), pp. 839-846, 1998

## 2. Grundlagen der Signaltheorie

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

21./28. Oktober und 4. November 2024

# Begriffsklärung

- **Signale** dienen dem Austausch von Informationen über das Verhalten oder die Natur physikalischer Phänomene wie z.B. Temperatur oder Druck. Mathematisch dargestellt werden sie als Funktion einer oder mehrerer unabhängigen Variablen.

*D. Sundararajan: „Digital Signal Processing. An Introduction“*

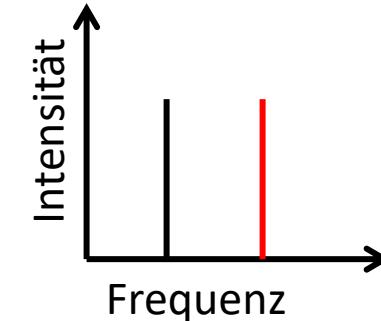
- Der Begriff „**Signal**“ umfasst unter anderem Audio-, Video-, Sprach-, Bild-, Kommunikations-, geophysikalische, Sonar-, Radar-, medizinische und musikalische Signale.

*“Aims and Scope” von IEEE Transactions on Signal Processing*

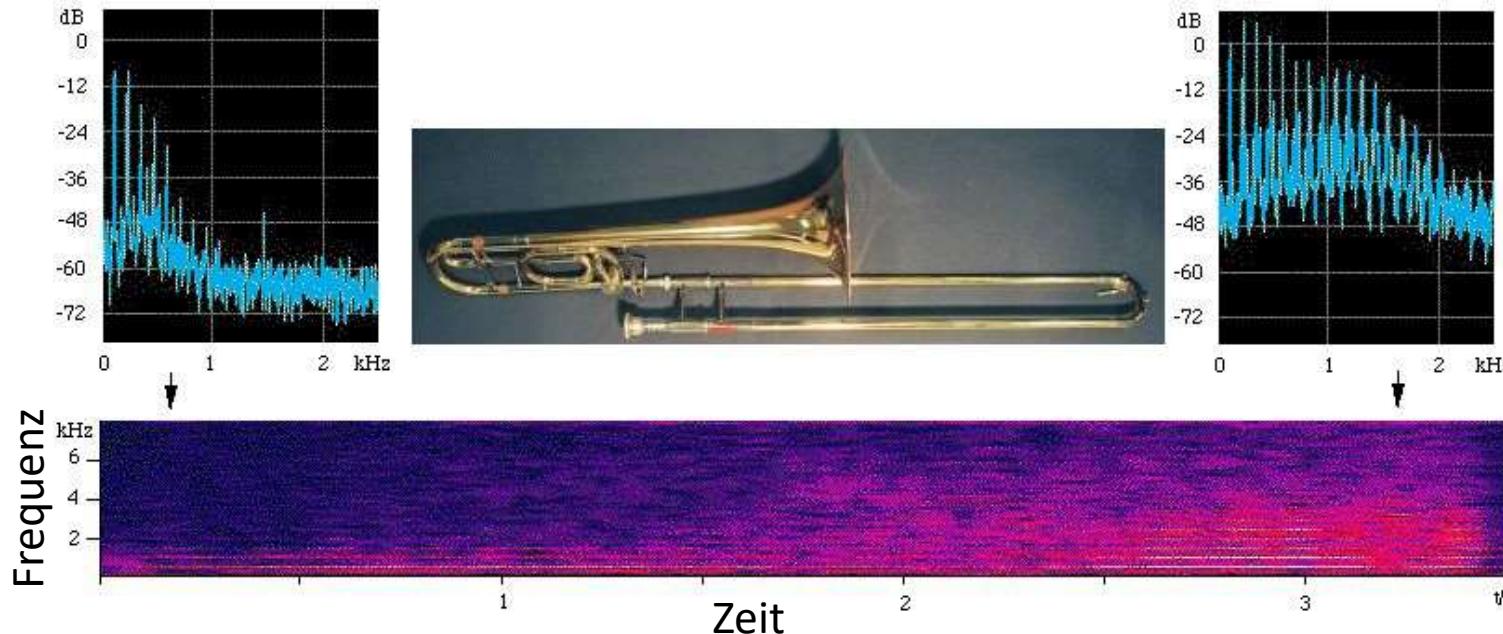
## **2.1 Diskrete Fourier-Transformation**

# Anschauung: Klänge als Mischung reiner Töne

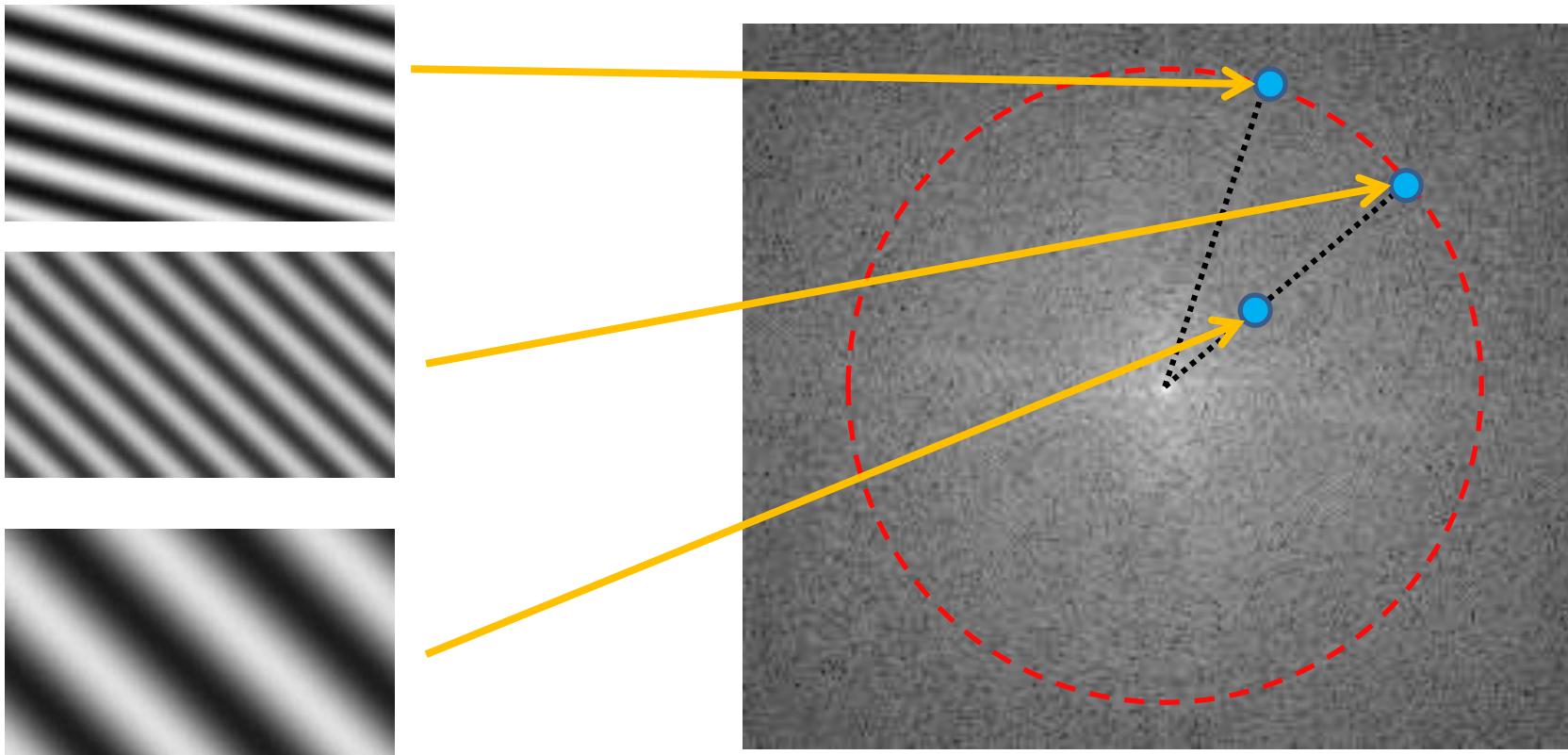
Frequenz entspricht Tonhöhe:



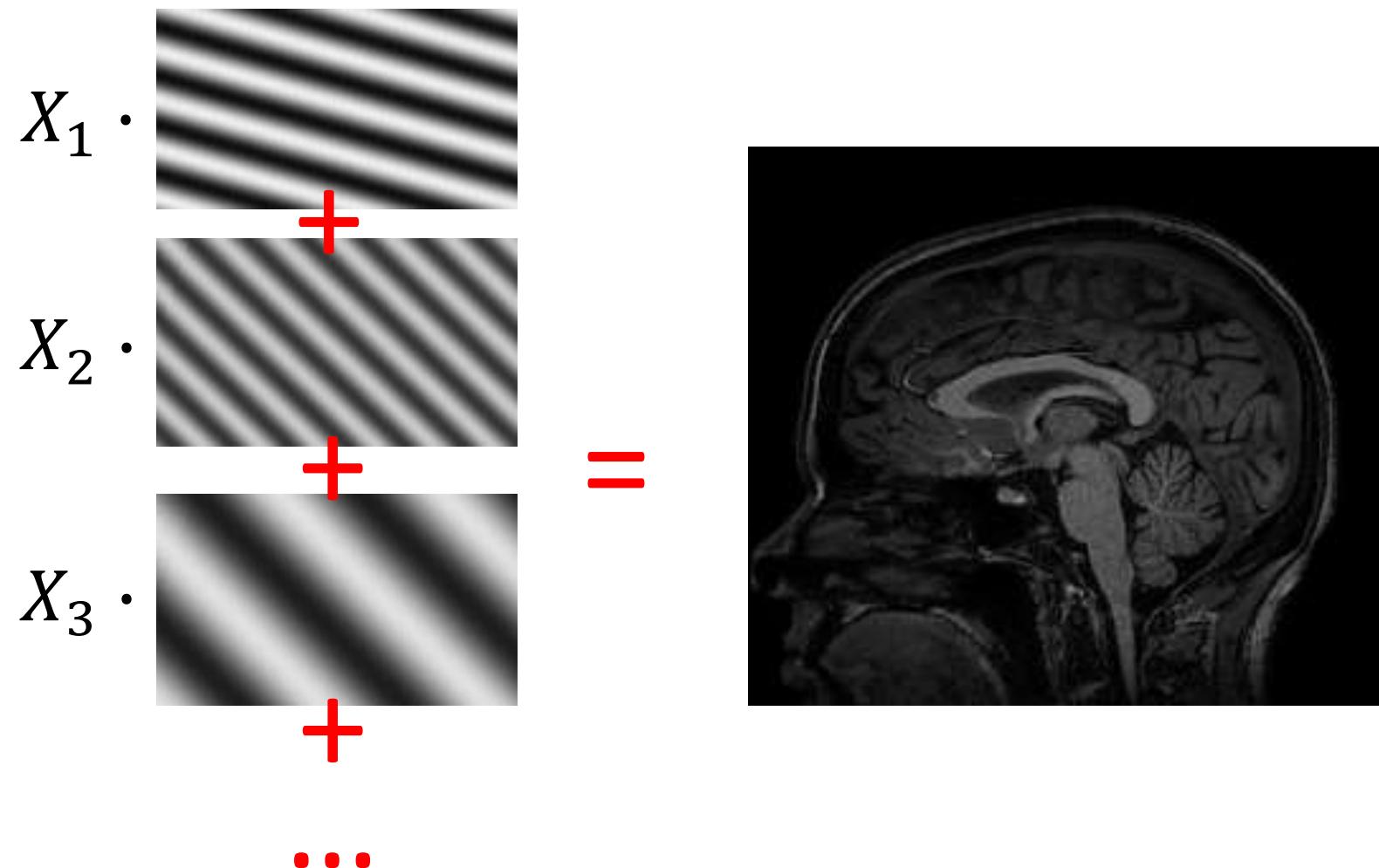
Mischung von Obertönen ergibt verschiedene Klangfarben beim selben Grundton:



# Anschauung: Orts-/Raumfrequenzen



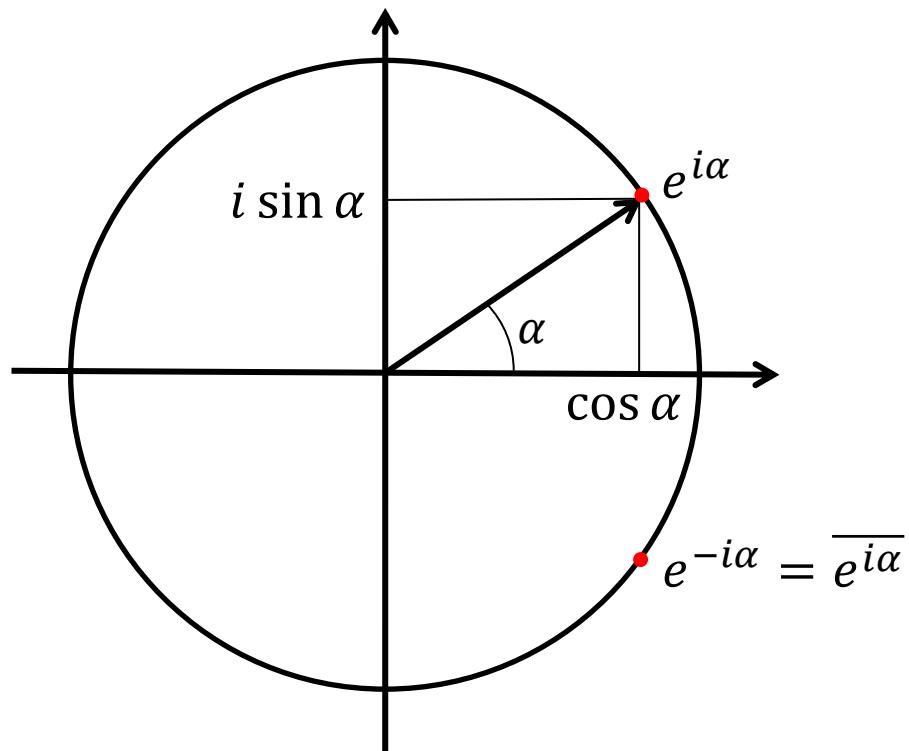
# Anschauung: Bilder als Mischung reiner Ortsfrequenzen

$$X_1 \cdot \begin{matrix} \text{+} \\ \text{+} \\ \text{+} \\ \dots \end{matrix} + X_2 \cdot \begin{matrix} \text{+} \\ \text{+} \\ \text{+} \\ \dots \end{matrix} + X_3 \cdot \begin{matrix} \text{+} \\ \text{+} \\ \text{+} \\ \dots \end{matrix} = \text{Brain MRI}$$


# Komplexe Notation: Euler'sche Formel

Die **Euler'sche Formel** verknüpft Sinus und Kosinus mit der komplexen Exponentialfunktion ( $i = \sqrt{-1}$ ):

$$e^{i\alpha} = \cos \alpha + i \sin \alpha$$



$$\cos \alpha = \frac{e^{i\alpha} + e^{-i\alpha}}{2}$$

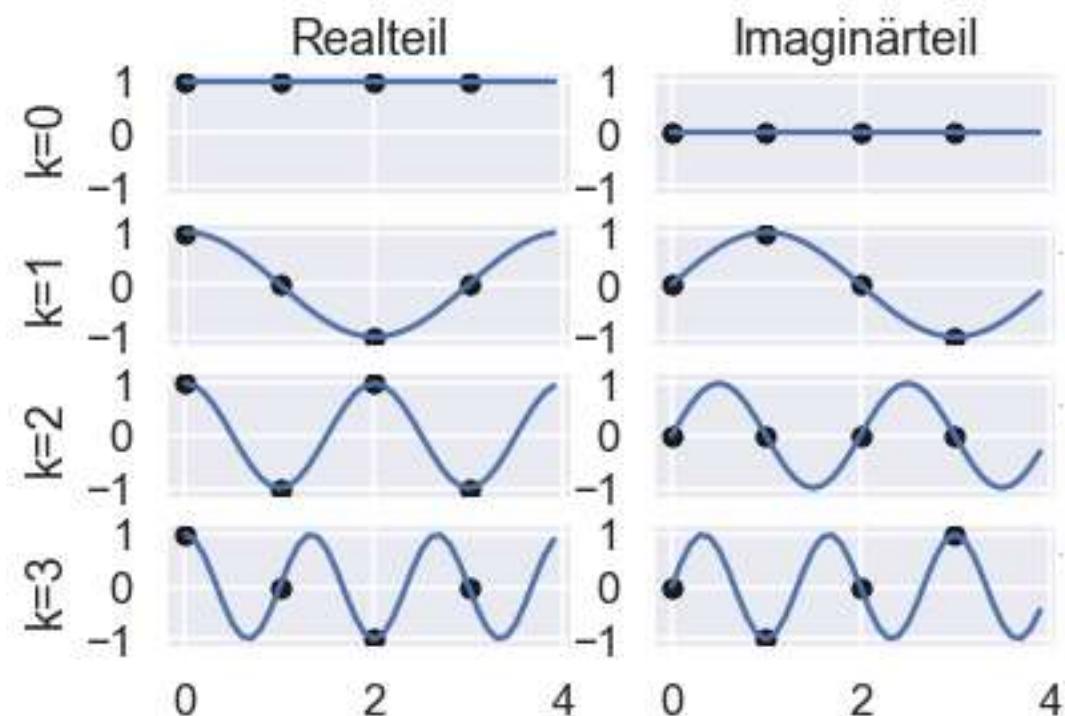
$$\sin \alpha = -i \frac{e^{i\alpha} - e^{-i\alpha}}{2}$$

**Polarform** komplexer Zahlen mit Absolutbetrag  $r$  und Winkel  $\phi$ :

$$z = r e^{i\phi} = r(\cos \phi + i \sin \phi)$$

# Grundidee: Diskrete Fourier-Transformation

- Die **diskrete Fourier-Transformation** (DFT) stellt ein diskretes Signal mit  $N$  Messwerten  $x_n \in \mathbb{C}$ ,  $n = 0, \dots, N - 1$ , durch  $N$  Fourierkoeffizienten  $X_k \in \mathbb{C}$ ,  $k = 0, \dots, N - 1$ , dar.
- Als Basis nutzt sie die Funktionen
$$e^{2\pi i k \frac{n}{N}}$$
- Beispiel für  $N = 4$ :



# Definition: Diskrete Fourier-Transformation

- Gegeben sei eine Folge von  $N$  diskreten Messwerten  $x_n \in \mathbb{C}$ ,  $n = 0, \dots, N - 1$
- Die **diskrete Fourier-Transformation** (DFT) bildet diese wie folgt auf  $N$  Fourierkoeffizienten  $X_k \in \mathbb{C}$ ,  $k = 0, \dots, N - 1$  ab:

$$X_k = \sum_{n=0}^{N-1} e^{-2\pi i k \frac{n}{N}} x_n$$

- Die entsprechende **Inverse DFT** ermöglicht es, aus den  $X_k$  wieder die  $x_n$  zu rekonstruieren:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} e^{2\pi i k \frac{n}{N}} X_k$$

# DFT Reeller Signale

- Komplexe Konjugation der DFT-Koeffizienten zeigt uns eine Symmetrie der DFT reeller Signale:

$$X_k = \sum_{n=0}^{N-1} e^{-2\pi i k \frac{n}{N}} x_n \Rightarrow \bar{X}_k = \sum_{n=0}^{N-1} e^{2\pi i k \frac{n}{N}} \bar{x}_n$$

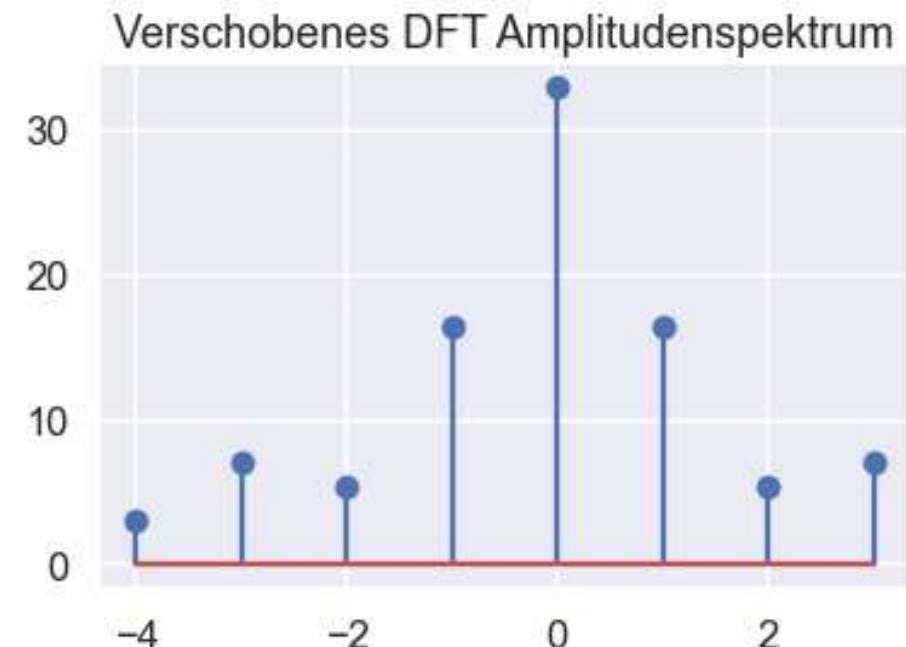
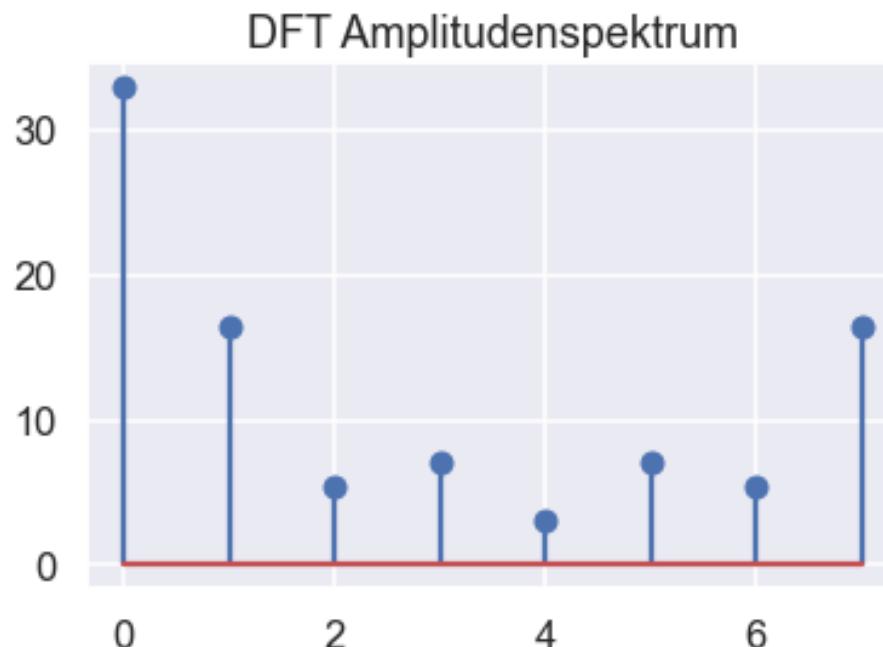
- Für reelle Signale ist  $x_n = \bar{x}_n$ . Somit ist:

$$\bar{X}_{N-k} = \sum_{n=0}^{N-1} e^{2\pi i (N-k) \frac{n}{N}} x_n = \sum_{n=0}^{N-1} \underbrace{e^{2\pi i n}}_{=1} e^{-2\pi i k \frac{n}{N}} x_n = X_k$$

– *Schlussfolgerung:* Für reelle Signale benötigen wir nur eine Hälfte der komplexen DFT-Koeffizienten, die andere Hälfte ist redundant.

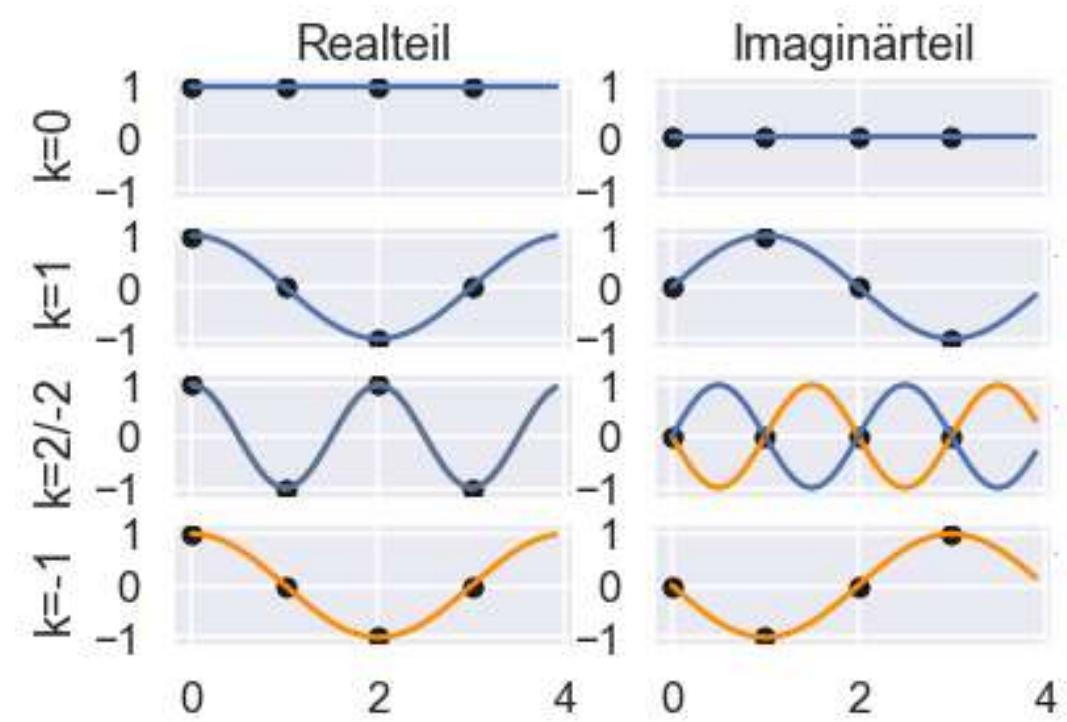
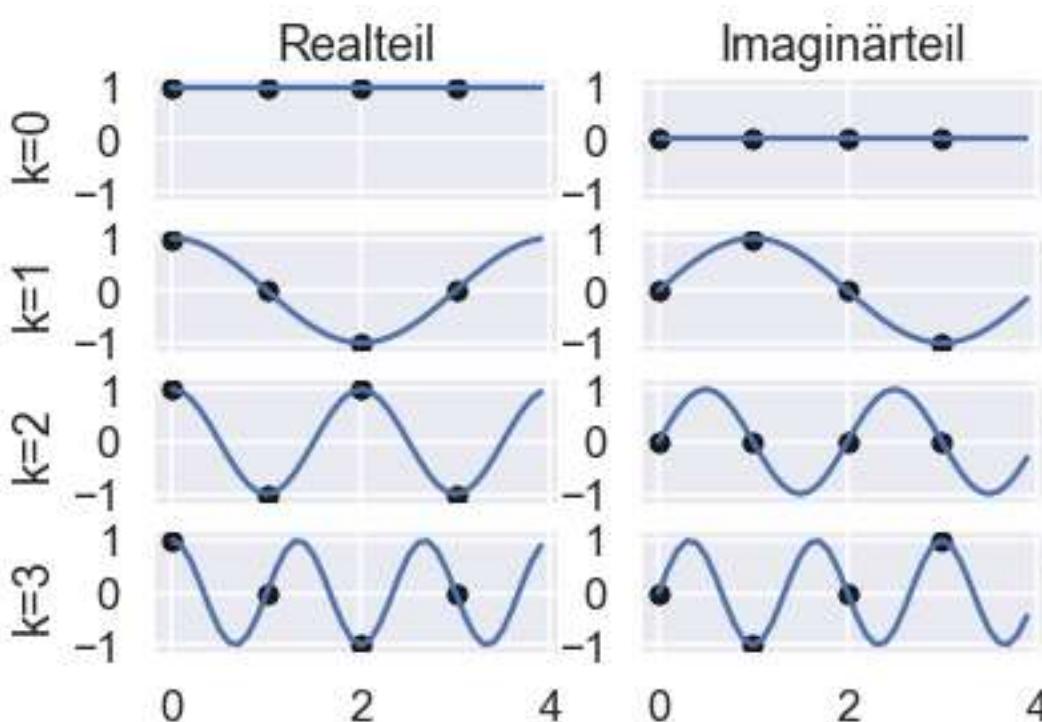
# Grafische Darstellung des Spektrums

- Zur grafischen Darstellung verschiebt man Spektren häufig so, dass sie bei Null zentriert sind
  - Für gerade  $N$  wird die höchste Frequenz ganz links dargestellt
  - In Bibliotheken häufig als `fftshift` bezeichnet. Vor der Rücktransformation ggf. durch `ifftshift` rückgängig machen!



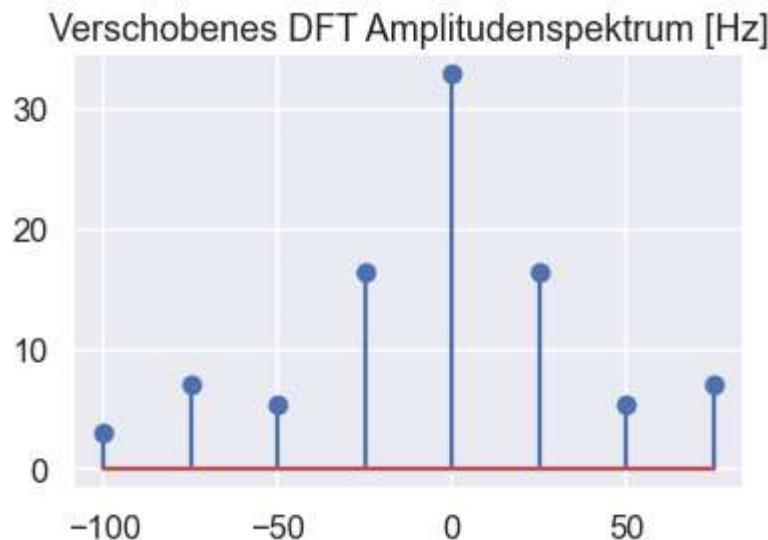
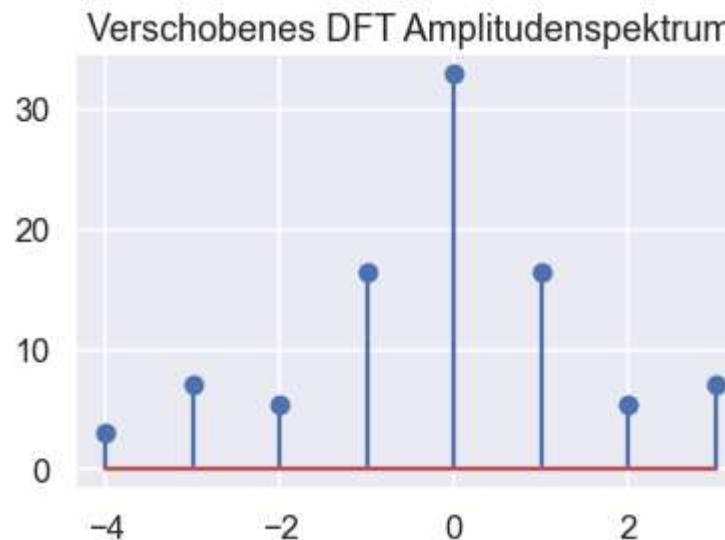
# Anschauung: Negative Frequenzen

- Die Äquivalenz von  $e^{2\pi i(N-k)\frac{n}{N}}$  und  $e^{-2\pi ik\frac{n}{N}}$  können wir auch grafisch veranschaulichen:



# Frequenzraster der DFT

- Decken die  $N$  Messpunkte einen Zeitraum von  $T$  Sekunden ab, entspricht der DFT-Koeffizient  $X_k$  der Frequenz  $\frac{k}{T}$  Hz
  - Bei einer Abtastfrequenz von  $\frac{N}{T}$  Hz ergibt sich als höchste „verwertbare“ Frequenz  $\frac{N}{2T}$  Hz
  - In der Bildverarbeitung analog mit Ortsfrequenzen
  - *Beispiel:* Abtastfrequenz von 200 Hz:



# Interpretation der DFT-Koeffizienten reeller Signale

- *Erinnerung:* **Polarform** mit Absolutbetrag  $r$  und Winkel  $\phi$ :

$$z = re^{i\phi} = r(\cos \phi + i \sin \phi)$$

- Einsetzen von  $X_k = \frac{N}{2}re^{i\phi}$ ,  $X_{N-k} = \frac{N}{2}re^{-i\phi}$ ,  
 $X_{k'} = 0$  für  $k' \notin \{k, N - k\}$  in die inverse DFT:

$$\begin{aligned}x_n &= \frac{1}{N} \left( \frac{N}{2}re^{i\phi} e^{2\pi ik \frac{n}{N}} + \frac{N}{2}re^{-i\phi} e^{2\pi i(N-k) \frac{n}{N}} \right) \\&= \frac{1}{2}r \left( e^{i(2\pi k \frac{n}{N} + \phi)} + e^{-i(2\pi k \frac{n}{N} + \phi)} \right) = r \cos \left( 2\pi k \frac{n}{N} + \phi \right)\end{aligned}$$

- *Schlussfolgerung:* Der Betrag des DFT-Koeffizienten  $X_k$  eines reellen Signals gibt (bis auf den Normierungsfaktor) die *Amplitude* einer Kosinus-Schwingung ( $k \times$  Grundfrequenz) an, der Winkel seine *Phase*

# Zusammenfassung

- Die **Diskrete Fourier-Transformation** (DFT) zerlegt diskrete Signale in ihre spektralen Anteile
  - Komplexe Notation: Interpretation als Amplitude und Phase
  - Redundanz bei der Transformation reeller Signale: Die zweite Hälfte der Koeffizienten ergibt sich durch Konjugation aus der ersten
- Die **inverse DFT** ermöglicht eine Synthese aus den spektralen Komponenten
  - Hierzu müssen wir Amplituden *und Phasen* kennen!
- *Hinweis:* **FFT** (Fast Fourier Transform) bezeichnet einen schnellen Algorithmus zur Berechnung der DFT.
  - Details in der Übung

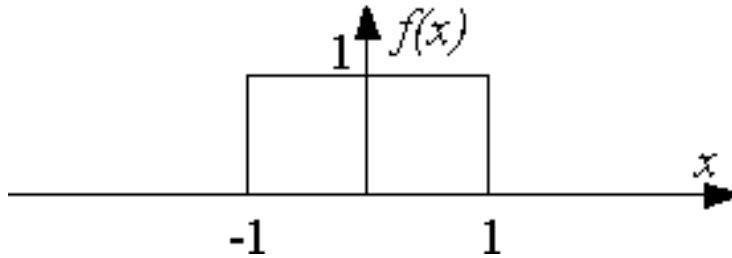
## **2.2 Eigenschaften der Fourier-Transformation**

# Kontinuierliche Fourier-Transformation

Analog zur DFT für endliche Folgen von Messwerten lässt sich eine kontinuierliche Fourier-Transformation für stückweise stetige, absolut integrierbare Funktionen  $f(x)$  herleiten:

<i>Vorwärts-Transformation <math>\mathcal{F}\{\cdot\}</math></i>	
$X_k = \sum_{n=0}^{N-1} e^{-2\pi i k \left(\frac{n}{N}\right)} x_n$	$F(u) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i u x} dx$
$x_n = \frac{1}{N} \sum_{k=0}^{N-1} e^{2\pi i k \left(\frac{n}{N}\right)} X_k$	$f(x) = \int_{-\infty}^{\infty} F(u) e^{2\pi i u x} du$
<i>Inverse Transformation <math>\mathcal{F}^{-1}\{\cdot\}</math></i>	

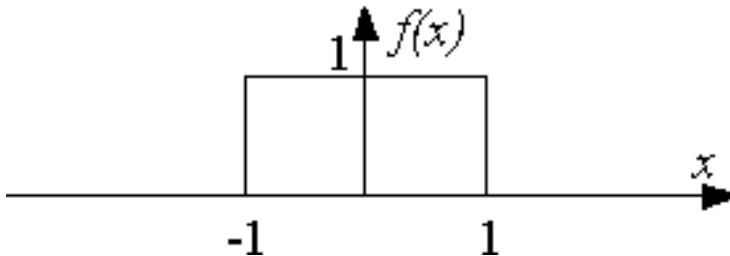
# Beispiel: Fourier-Transformation einer Rechteckfunktion



$$\begin{aligned} F(u) &= \int_{-\infty}^{\infty} f(t) e^{-2\pi i u t} dt = \int_{-1}^{1} e^{-2\pi i u t} dt = -\frac{1}{2\pi i u} [e^{-2\pi i u t}]_{-1}^1 \\ &= \frac{i}{2\pi u} (e^{-2\pi i u} - e^{2\pi i u}) = \frac{1}{\pi u} (-i) \frac{e^{2\pi i u} - e^{-2\pi i u}}{2} \end{aligned}$$

$$\text{Erinnerung: } \sin \alpha = -i \frac{e^{i\alpha} - e^{-i\alpha}}{2}$$

# Beispiel: Fourier-Transformation einer Rechteckfunktion



$$\begin{aligned} F(u) &= \int_{-\infty}^{\infty} f(t) e^{-2\pi i u t} dt = \int_{-1}^{1} e^{-2\pi i u t} dt = -\frac{1}{2\pi i u} [e^{-2\pi i u t}]_{-1}^1 \\ &= \frac{i}{2\pi u} (e^{-2\pi i u} - e^{2\pi i u}) = \frac{1}{\pi u} (-i) \frac{e^{2\pi i u} - e^{-2\pi i u}}{2} = 2 \frac{\sin(2\pi u)}{2\pi u} \end{aligned}$$

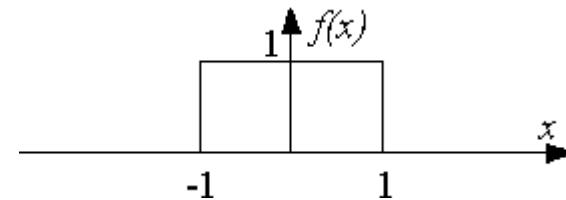
$$\text{Erinnerung: } \sin \alpha = -i \frac{e^{i\alpha} - e^{-i\alpha}}{2}$$

$$\text{sinc}(x) = \frac{\sin(x)}{x}$$

# Rechteckfunktion im Orts- und Frequenzraum

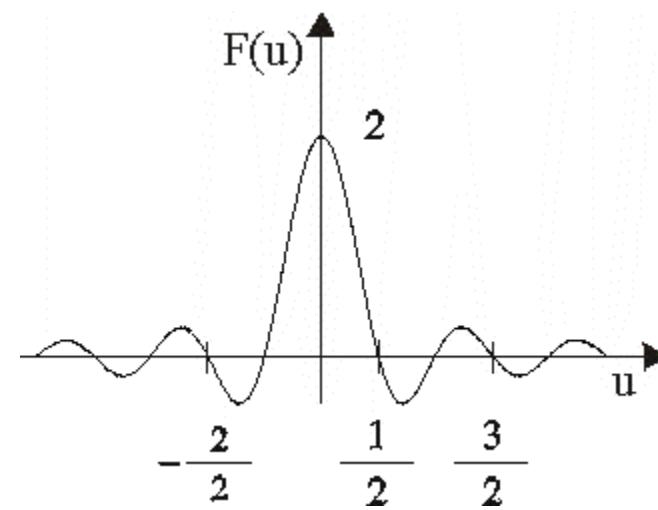
- Ortsraum

$$f(x) = \begin{cases} 1, & x \in [-1, 1] \\ 0, & \text{otherwise} \end{cases}$$



- Frequenzraum

$$F(u) = 2 \frac{\sin(2\pi u)}{2\pi u}$$



# Dualität zwischen Orts- und Frequenzraum

- **Satz:** Wenn  $F(u)$  die Fourier-Transformation von  $f(x)$  ist, dann ist  $f(-x)$  die Fourier-Transformation von  $F(u)$ .
- **Beweis:** Ergibt sich direkt aus der Ähnlichkeit der Definitionen von Fourier-Transformation und ihrer Inversen:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i ux} dx \quad \text{Vorwärts-Transformation von } f(x)$$

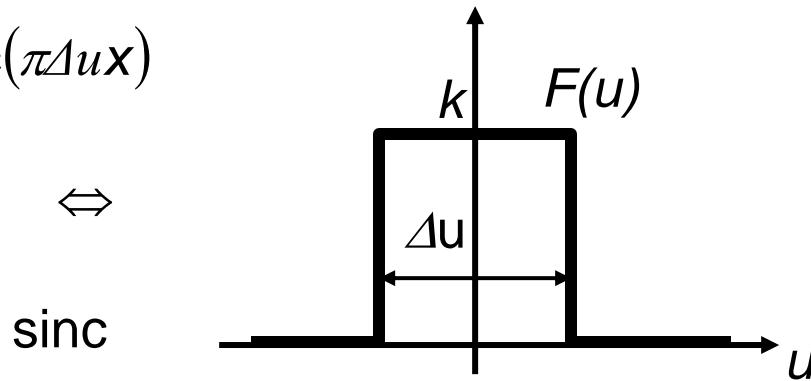
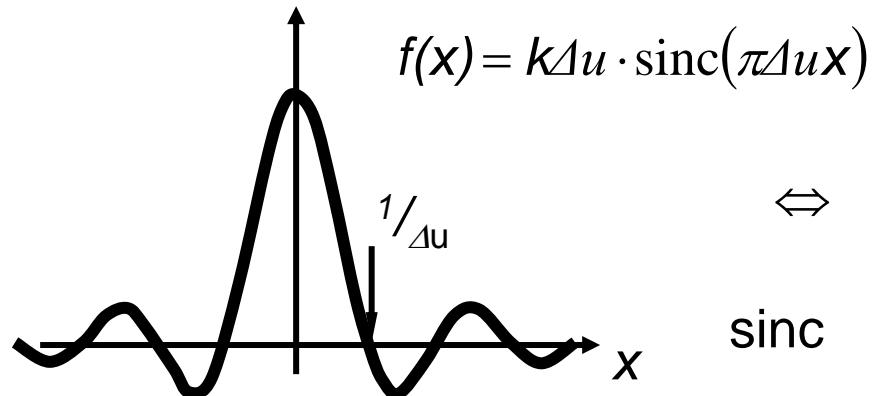
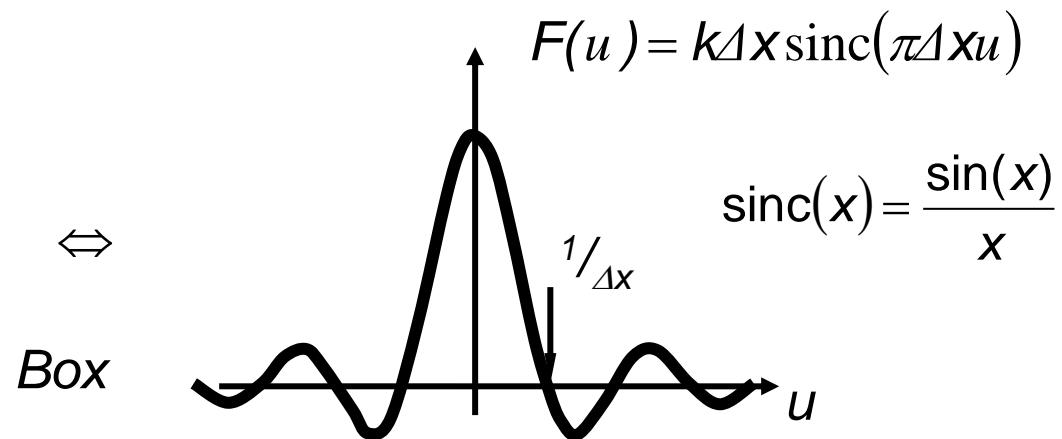
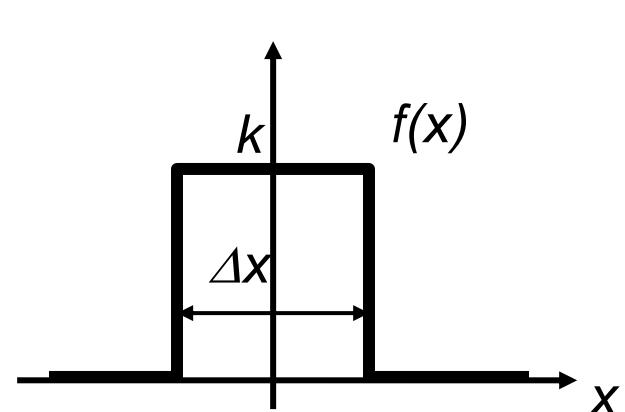
$$\Leftrightarrow f(x) = \int_{-\infty}^{\infty} F(u) e^{2\pi i ux} du \quad \text{Inverse Transformation von } F(u)$$

Daher:

$$f(-x) = \int_{-\infty}^{\infty} F(u) e^{-2\pi i ux} du$$

= Vorwärts-Transformation von  $F(u)$

# Fourier-Paar: Rechteck und Sinus cardinalis (sinc)

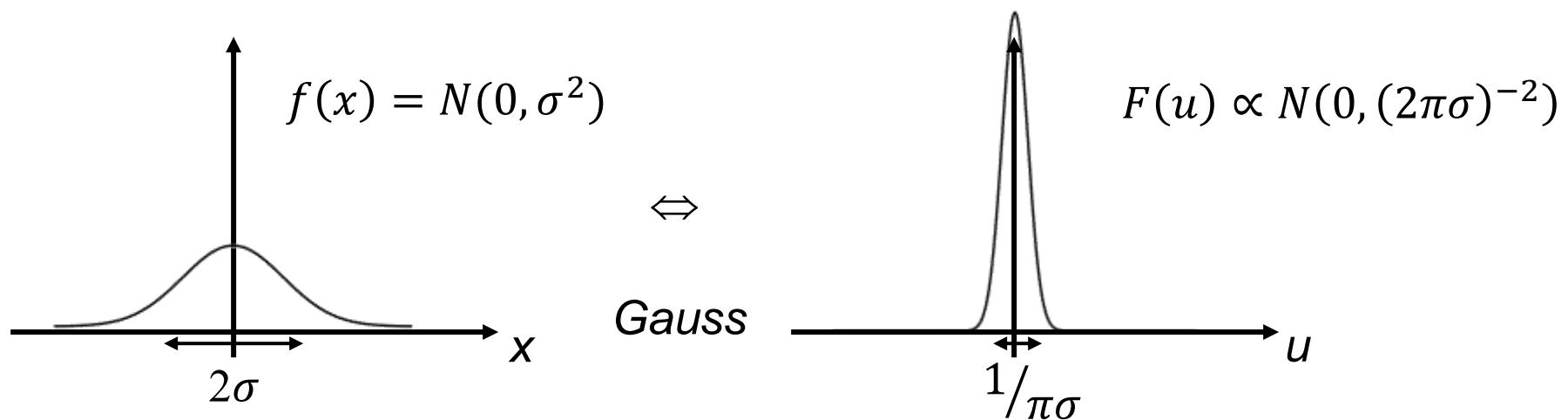


**Bemerkung:** Für gerade Funktionen  $f(-x)=f(x)$  spielt der Vorzeichenwechsel in der Fourier-Dualität keine Rolle.

# Fourier-Transformation der Normalverteilung

Die Fourier-Transformation einer Gauss-Verteilung  $\mathcal{N}(0, \sigma^2)$  ist

$$\mathcal{F}\{\mathcal{N}(0, \sigma^2)\} = e^{-\frac{1}{2} \frac{u^2}{\sigma^2}} \text{ mit } \sigma_u = \frac{1}{2\pi\sigma}$$



# Faltungssatz

- **Satz:**  $f(x) = h(x)^* g(x)$  entspricht  $F(u) = H(u) \cdot G(u)$
- **Beweis:** 
$$\begin{aligned} F(u) &= \int_{x=-\infty}^{\infty} \int_{\xi=-\infty}^{\infty} h(\xi) g(x - \xi) d\xi e^{-2\pi i u x} dx \\ &= \int_{\xi=-\infty}^{\infty} h(\xi) \int_{x=-\infty}^{\infty} g(x - \xi) e^{-2\pi i u x} dx d\xi \end{aligned}$$

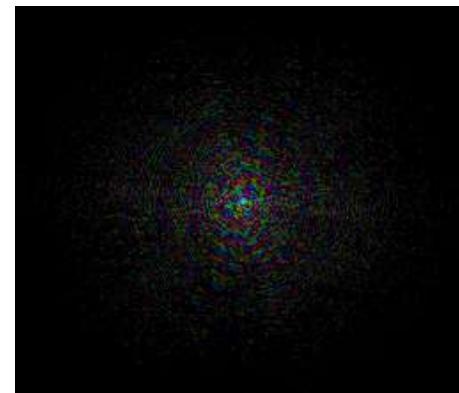
mit  $y := x - \xi$  ( $dy = dx$ ):

$$\begin{aligned} &= \int_{\xi=-\infty}^{\infty} h(\xi) \int_{y=-\infty}^{\infty} g(y) e^{-2\pi i u(y+\xi)} dy d\xi \\ &= \underbrace{\int_{\xi=-\infty}^{\infty} h(\xi) e^{-2\pi i u \xi} d\xi}_{=:H(u)} \underbrace{\int_{y=-\infty}^{\infty} g(y) e^{-2\pi i u y} dy}_{=:G(u)} \end{aligned}$$

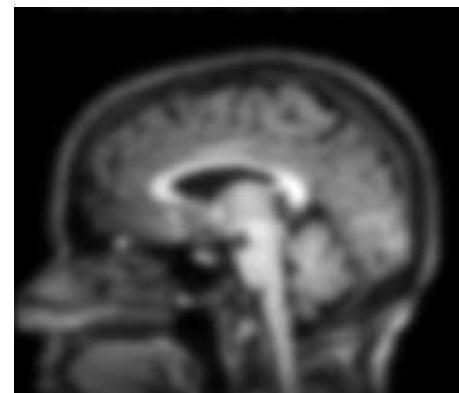
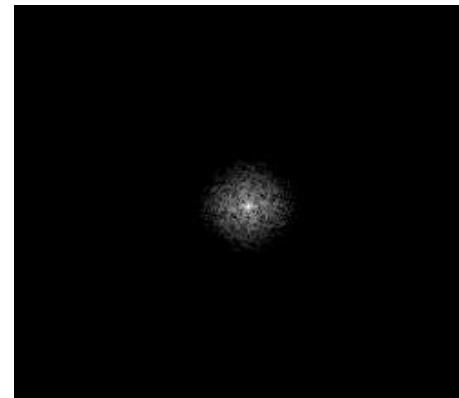
**Hinweis:** Ebenso gilt:  $f(x) = h(x) \cdot g(x)$  entspricht  $F(u) = H(u) * G(u)$

# Erste Anwendung des Faltungssatzes

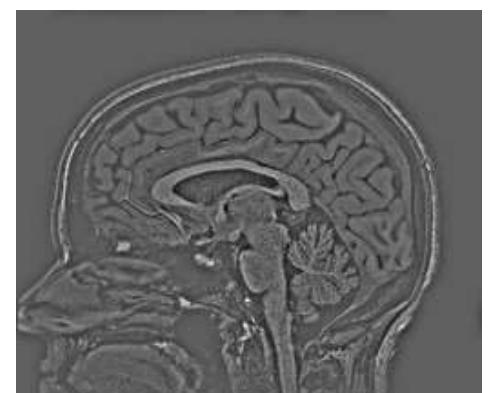
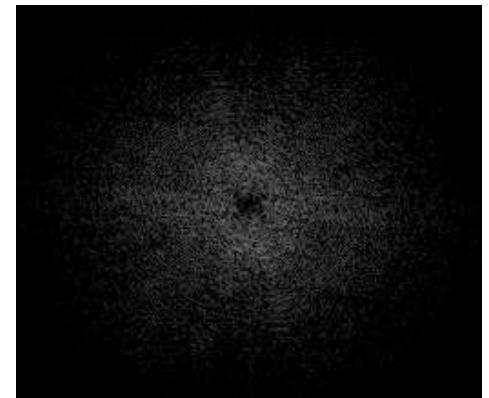
- Bildfilterung durch Faltung mit einem Gauss-Kern entspricht einer gaussförmigen Abschwächung höherer Frequenzen
  - Multiplikation mit  $1 - e^{-\frac{\|s\|^2}{2\sigma_u^2}}$  im Frequenzraum ermöglicht Hochpass-Filter
  - Filterung im Frequenzraum impliziert periodische Randbedingung



Eingabe



Tiefpass



Hochpass

# Fourier-Transformation in mehreren Dimensionen

- In **Vektor-Notation** generalisiert die Fourier-Transformation wie folgt auf mehrere Dimensionen (z.B. Bilder):
  - Vorwärts (Beispiel: 3D):

$$F(\mathbf{s}) = \iiint_{-\infty}^{\infty} f(\mathbf{r}) e^{-2\pi i (\mathbf{r} \cdot \mathbf{s})} d\mathbf{r}$$

- Invers:

$$f(\mathbf{r}) = \iiint_{-\infty}^{\infty} F(\mathbf{s}) e^{2\pi i (\mathbf{r} \cdot \mathbf{s})} d\mathbf{s}$$

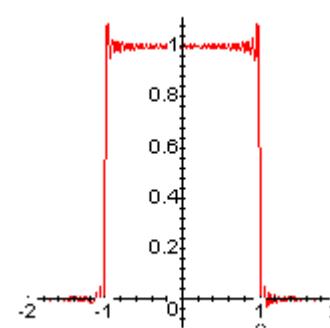
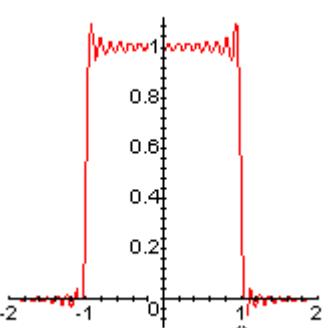
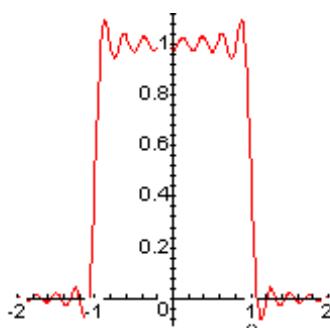
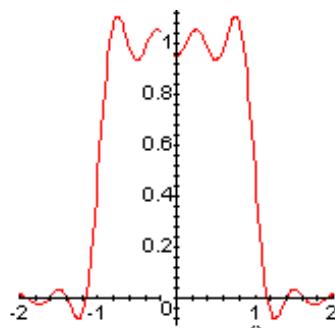
- In **Koordinaten-Notation** ist die Separierbarkeit leicht zu erkennen:
  - Beispiel: Vorwärts-Transformation in 2D

$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-2\pi i (ux + vy)} dx dy$$

# Beispiel: Unvollständige Rekonstruktion der Rechteckfunktion

- Frage: Wie verhält sich die inverse Fourier-Transformation, wenn wir die Integrationsgrenzen einschränken?

$$f(x) = \int_{-\infty}^{\infty} 2 \frac{\sin(2\pi u)}{2\pi u} e^{2\pi iux} du$$



$$\int_{-1}^1 \dots du$$

$$\int_{-2}^2 \dots du$$

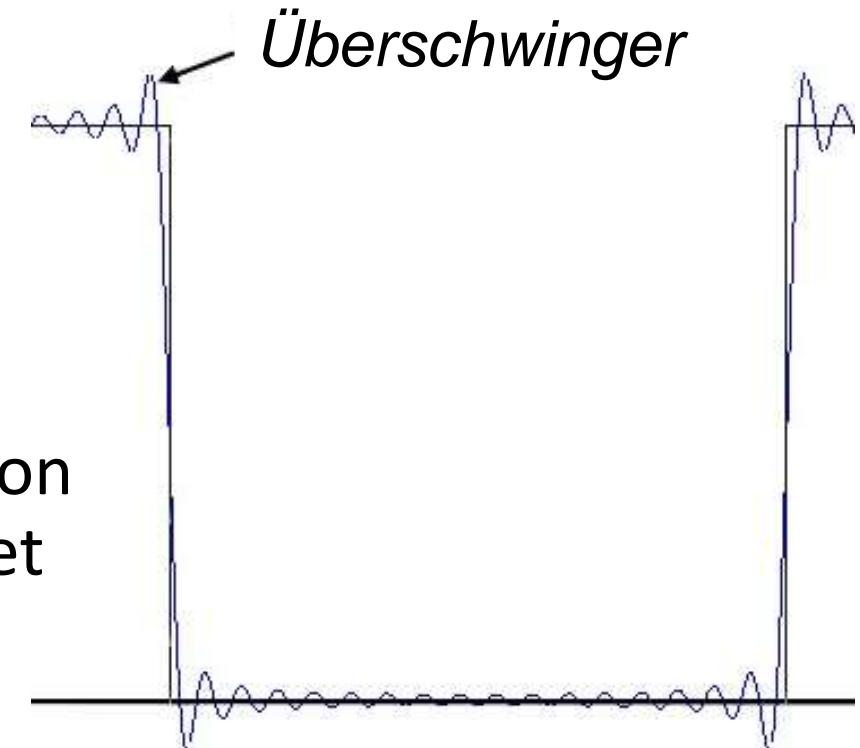
$$\int_{-4}^4 \dots du$$

$$\int_{-8}^8 \dots du$$

# Das Gibbs'sche Phänomen

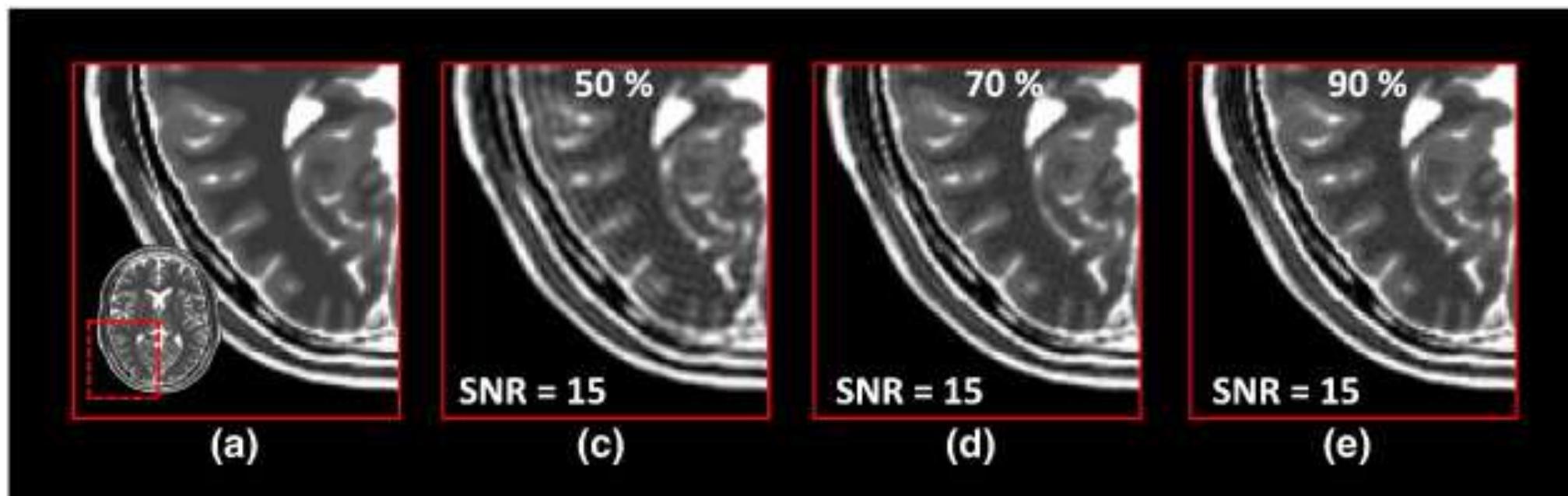
Beobachtung von **Josiah Willard Gibbs**:

- Berücksichtigt man bei der inversen Fourier-Transformation nur Frequenzen bis zu einer beliebig hohen, aber endlichen Grenze, treten in der Umgebung von Sprungstellen **Überschwinger** auf
- Der prozentuale Fehler beträgt stets ca. 18% der Sprunghöhe
- Konvergenz im Hinblick auf die Breite, nicht aber die Höhe der Überschwinger
- *Erklärung:* Abschneiden hoher Frequenzen entspricht Multiplikation mit Rechteckfunktion im Frequenzraum. Laut Faltungssatz bedeutet das Faltung mit sinc im Ortsraum.



# Beispiel: Das Gibbs'sche Phänomen

- In manchen bildgebenden Verfahren (insbesondere Magnetresonanztomographie) kann das Gibbs'sche Phänomen Überschwinger in der Nähe scharfer Kanten erzeugen
  - Warum genau sehen wir in Kapitel 3



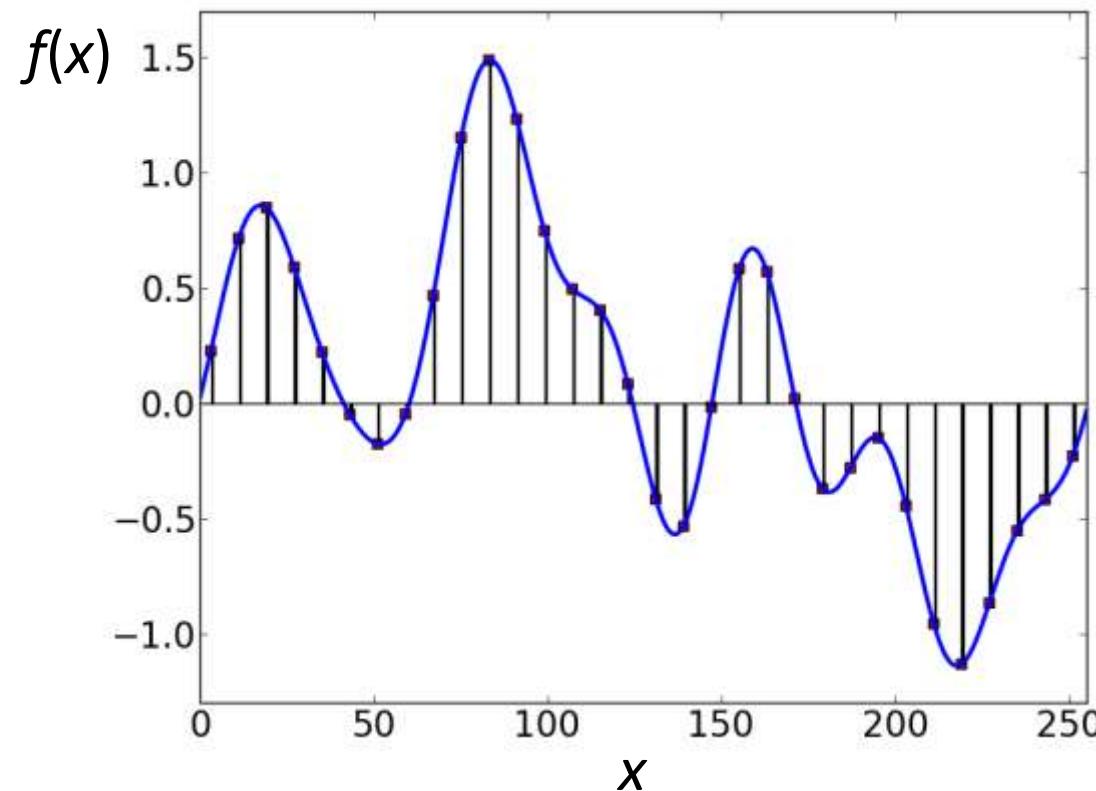
# Zusammenfassung

- Die **kontinuierliche Fourier-Transformation** ermöglicht eine Frequenzanalyse und –synthese auch für integrierbare Funktionen
- Für uns wichtige **Eigenschaften** sind
  - Dualität von Orts- und Frequenzraum und bestimmte **Fourier-Paare**
    - Rechteckfunktion vs. Sinc
    - Gauss und (nicht normierter) Gauss
  - Der **Faltungssatz**. Er ermöglicht
    - Verständnis und Implementierung von **Bildfiltern**
    - Verständnis des **Gibbs'schen Phänomens**

## **2.3 Das Abtasttheorem**

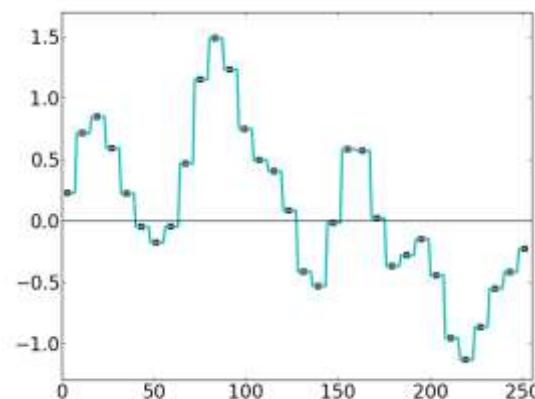
# Abtastung kontinuierlicher Signale

- Um ein durch eine kontinuierliche Funktion  $f(x)$  beschriebenes Signal zu diskretisieren, tasten wir in gleichmäßigen Abständen eine Folge von Messwerten ab

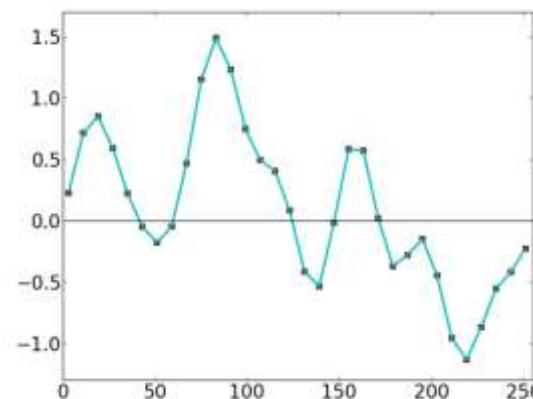


# Rekonstruktion diskreter Signale

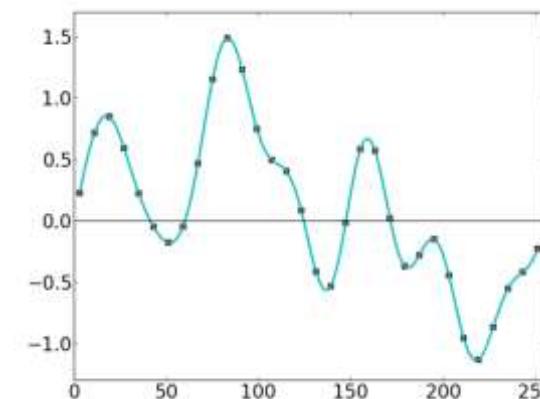
- **Interpolation** ermöglicht die Rekonstruktion einer kontinuierlichen Funktion  $f(x)$  aus Abtastwerten
- *Beispiel:* Verschiedene Interpolationen derselben Werte



Nächster Nachbar



Linear



Spline

# Grundlegende Fragen digitaler Signalverarbeitung

- **Wie dicht müssen wir das Signal abtasten?**
  - Welche Fehler haben wir zu erwarten, wenn wir nicht eng genug abtasten?
- **Wie sollten wir interpolieren?**
  - Können wir insbesondere das **ursprüngliche Signal exakt rekonstruieren?** Unter welchen Bedingungen?
- **Das Abtasttheorem** wird uns diese Fragen beantworten
  - Auch wenn man in der Praxis z.T. davon abweicht ist es wichtig, sich über die Konsequenzen im Klaren zu sein!

# Notation: Delta-Distribution

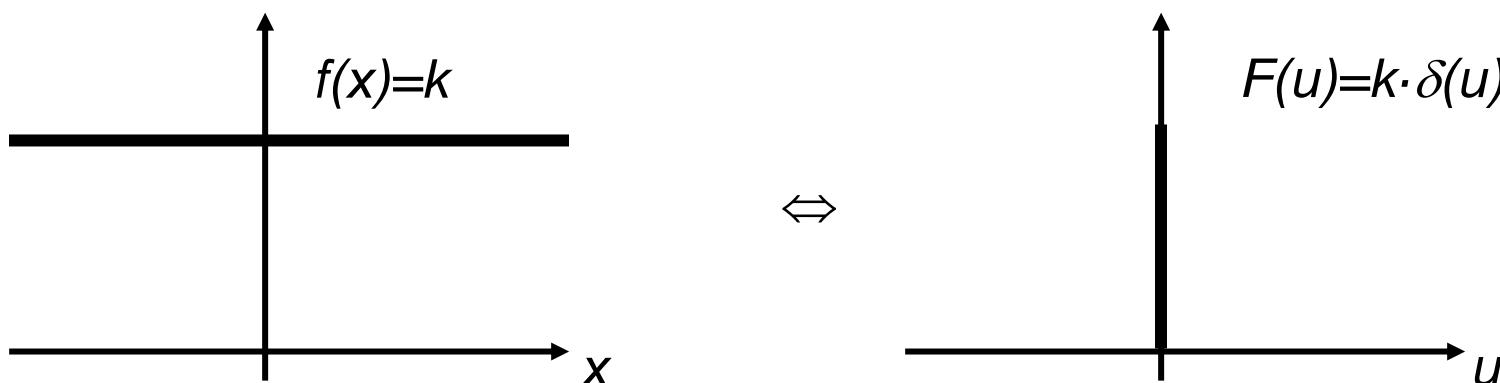
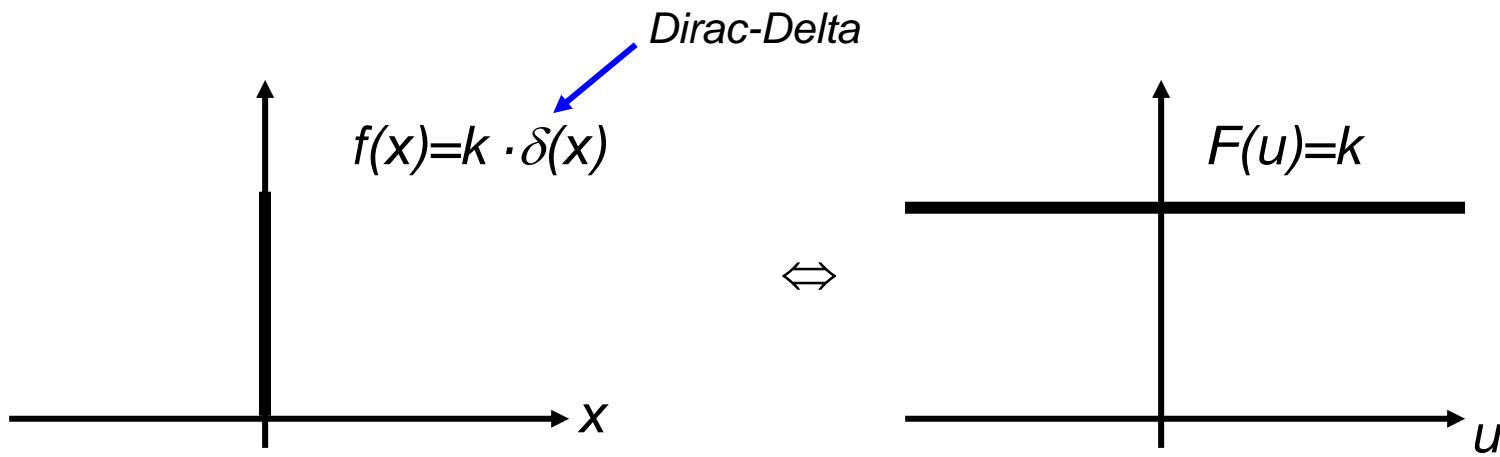
- Die Dirac'sche **Delta-Distribution**  $\delta$  wertet Funktionen  $f(x)$  an der Stelle  $x = 0$  aus
- Verbreitete Notation in der Signalverarbeitung:

$$\int_{-\infty}^{\infty} f(x) \delta(x - a) dx = f(a)$$

- Anschaulich (aber nicht ganz korrekt) stellt man sich  $\delta(x)$  häufig als Funktion vor, die für alle  $x \neq 0$   $\delta(x) = 0$  annimmt,  $\delta(0) = \infty$ ,  $\int_{-\infty}^{\infty} \delta(x) dx = 1$ 
  - z.B. Grenzwert der Gauss-Verteilung für  $\sigma \rightarrow 0$
  - Grafische Darstellung von  $\delta(x - a)$  durch senkrechte Linie bei  $x = a$

# Fourier-Paar: Konstante Funktion und Delta

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i ux} dx$$

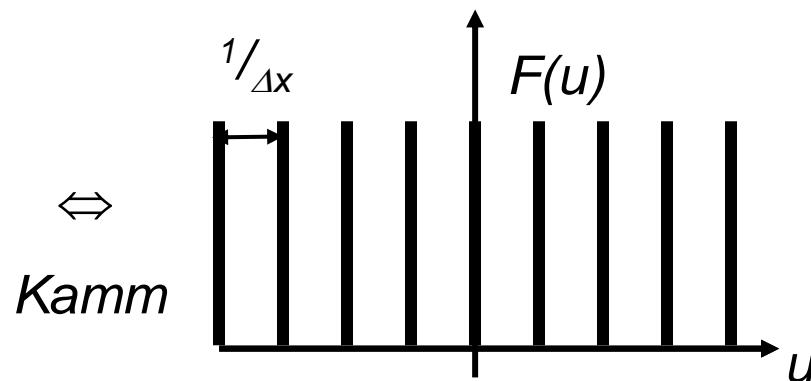
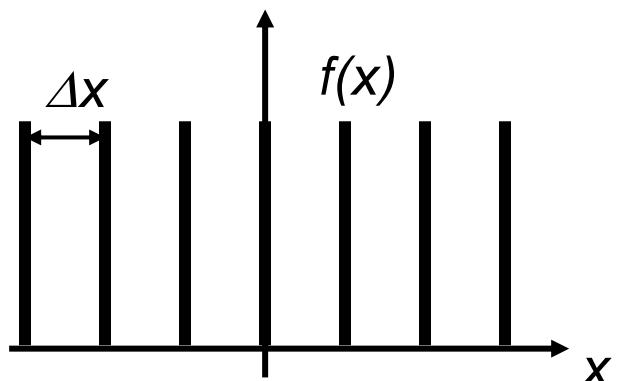


# Fourier-Transformation des Dirac-Kamms

- Als **Dirac-Kamm**  $\mathbb{W}_{\Delta x}$  bezeichnen wir eine unendliche Folge von Deltas mit Abstand  $\Delta x$ :

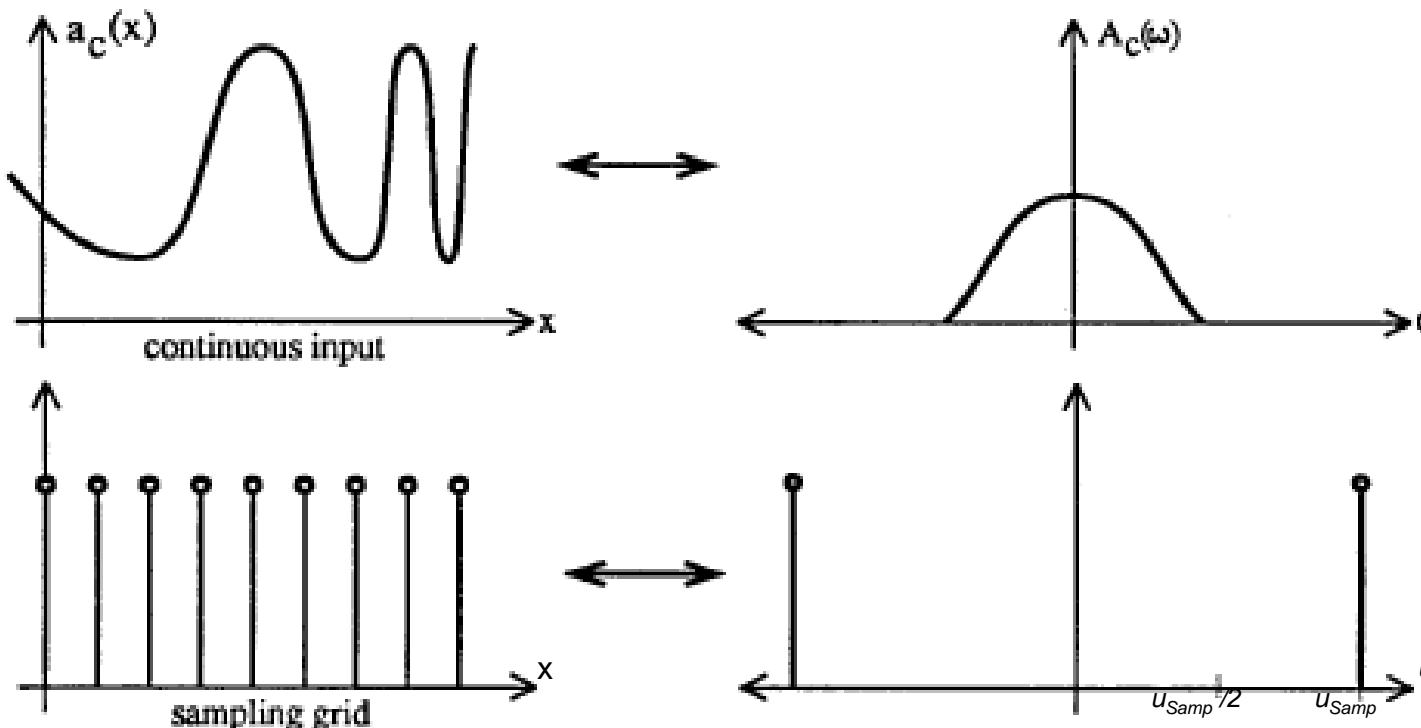
$$\mathbb{W}_{\Delta x}(x) = \sum_{k=-\infty}^{\infty} \delta(x - k\Delta x)$$

- Die Fourier-Transformierte von  $\mathbb{W}_{\Delta x}$  ist  $\mathcal{F}\{\mathbb{W}_{\Delta x}\} = \frac{1}{\Delta x} \mathbb{W}_{\frac{1}{\Delta x}}$



# Auswirkung der Abtastung im Frequenzraum

- Eine regelmäßige Abtastung  $x_k = f(k \cdot \Delta x)$  von  $f(x)$  entspricht einer Multiplikation von  $f$  mit  $\mathbb{W}_{\Delta x}$
- Laut Faltungssatz ergibt sich im Frequenzraum eine Faltung des Spektrums  $F(u)$  mit  $\frac{1}{\Delta x} \mathbb{W}_{\frac{1}{\Delta x}}$



= Nyquist frequency      = sampling frequency      38

# Faltung mit dem Dirac-Kamm

1. Eine Faltung von  $f(x)$  mit  $\delta(x - a)$  ergibt  $f(x - a)$ :

$$f(x) * \delta(x - a) = \int_{-\infty}^{\infty} f(\xi) \delta(x - a - \xi) d\xi$$

$\delta(x) = \delta(-x)$  

$$= \int_{-\infty}^{\infty} f(\xi) \delta(\xi - (x - a)) d\xi \stackrel{\text{Definition von } \delta \text{ (Folie 35)}}{=} f(x - a)$$

2. Aufgrund der Linearität der Faltung ergibt sich für  $\mathbb{W}_{\Delta x}$

$$f(x) * \mathbb{W}_{\Delta x} = f(x) * \sum_{k=-\infty}^{\infty} \delta(x - k\Delta x) = \sum_{k=-\infty}^{\infty} f(x - k\Delta x)$$

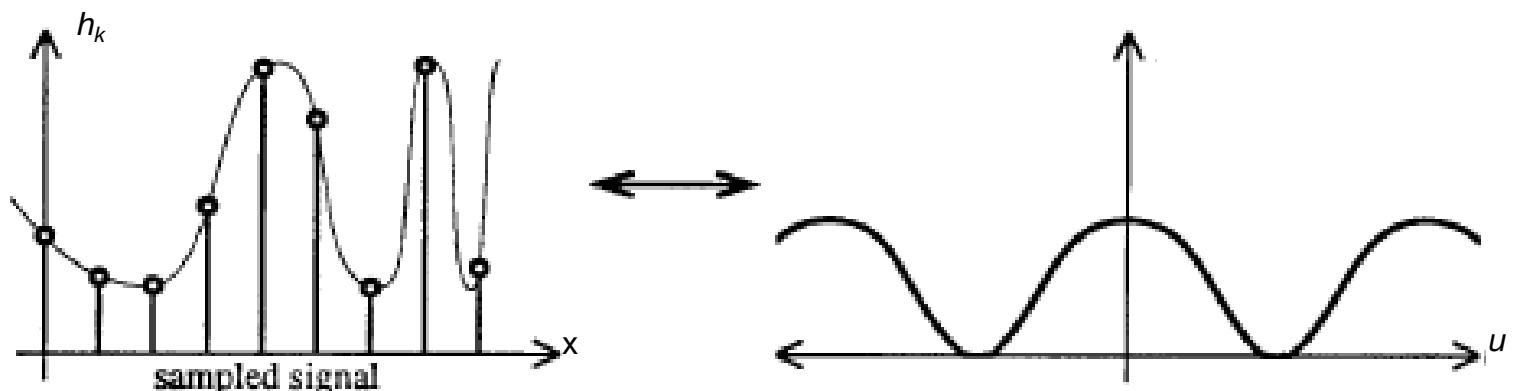
d.h. wir summieren um  $k\Delta x$  verschobene Kopien von  $f$

# Alias-Effekt

- Faltung mit  $\frac{1}{\Delta x} \text{Ш}_{\frac{1}{\Delta x}}$  im Frequenzraum erzeugt Kopien des Spektrums im Abstand  $\Delta u = 1/\Delta x$

– **Alias-Effekt:** Frequenzanteile von  $f$  werden bei anderen Frequenzen dupliziert (lateinisch *alias*=anderswo)

- Wenn das Spektrum keine Frequenzen größer als  $\Delta u/2$  enthält, überlappen diese Kopien nicht
  - **Bandbeschränkung** mit Bandbreite (Grenzfrequenz)  $B \leq \Delta u/2$
- Das ursprüngliche Spektrum lässt sich durch Multiplikation einer Rechteckfunktion der Breite  $\Delta u$  im Frequenzraum rekonstruieren

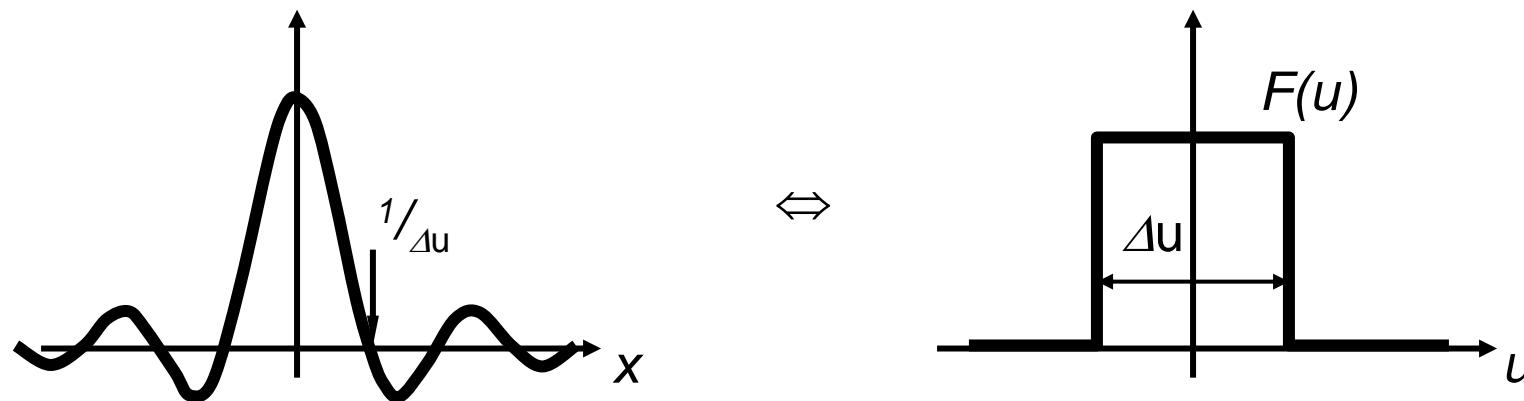


# Nyquist-Frequenz

- Die Abtastung eines bandbeschränkten Signals  $f$  mit Bandbreite  $B$  oberhalb der **Nyquist-Frequenz**  $\omega=2B$  verhindert das Auftreten von Aliasing-Effekten.
- In der Praxis ist eine entsprechende Bandbegrenzung häufig nicht gegeben
  - Hohe Frequenzen sollten durch eine Vorfilterung entfernt werden

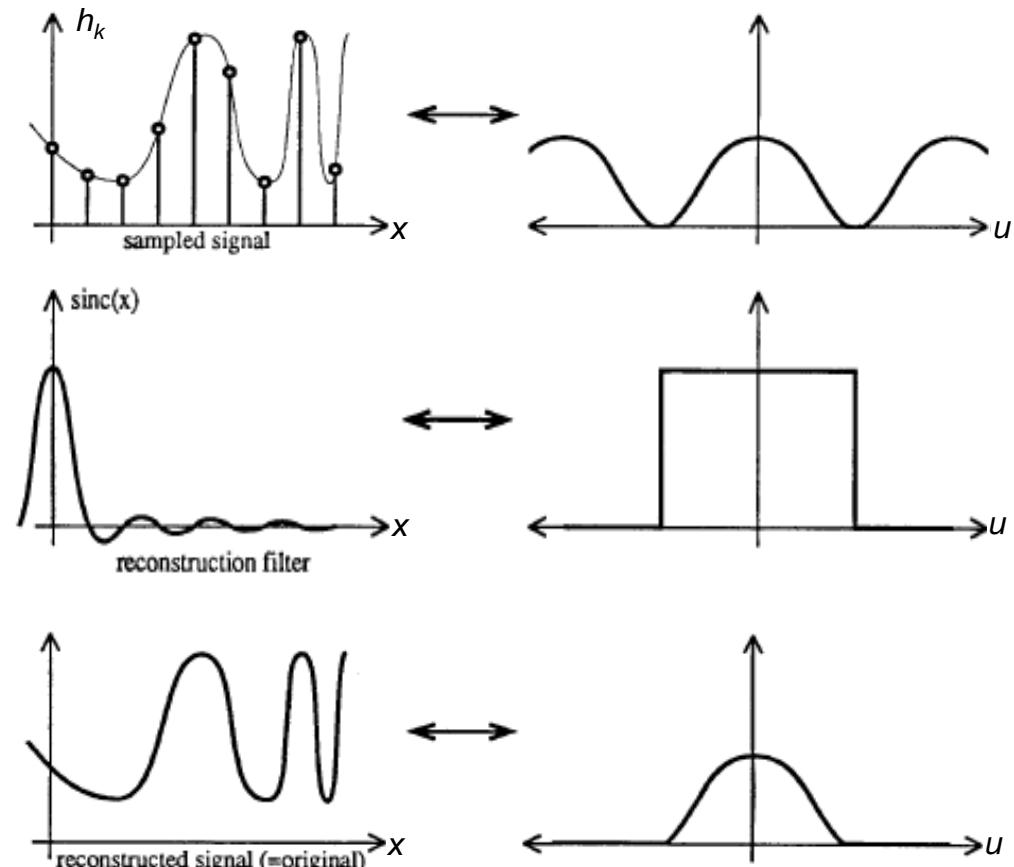
# Rekonstruktion des Signals

- Der Multiplikation mit einer Rechteckfunktion, mit der die überflüssigen Kopien des Spektrums wieder entfernt werden, entspricht eine Faltung mit dem sinc im Ortsraum
  - Rekonstruiert aus Messpunkten ein kontinuierliches Signal
  - Beispiel für faltungsbasierte Interpolation (siehe 2.4)



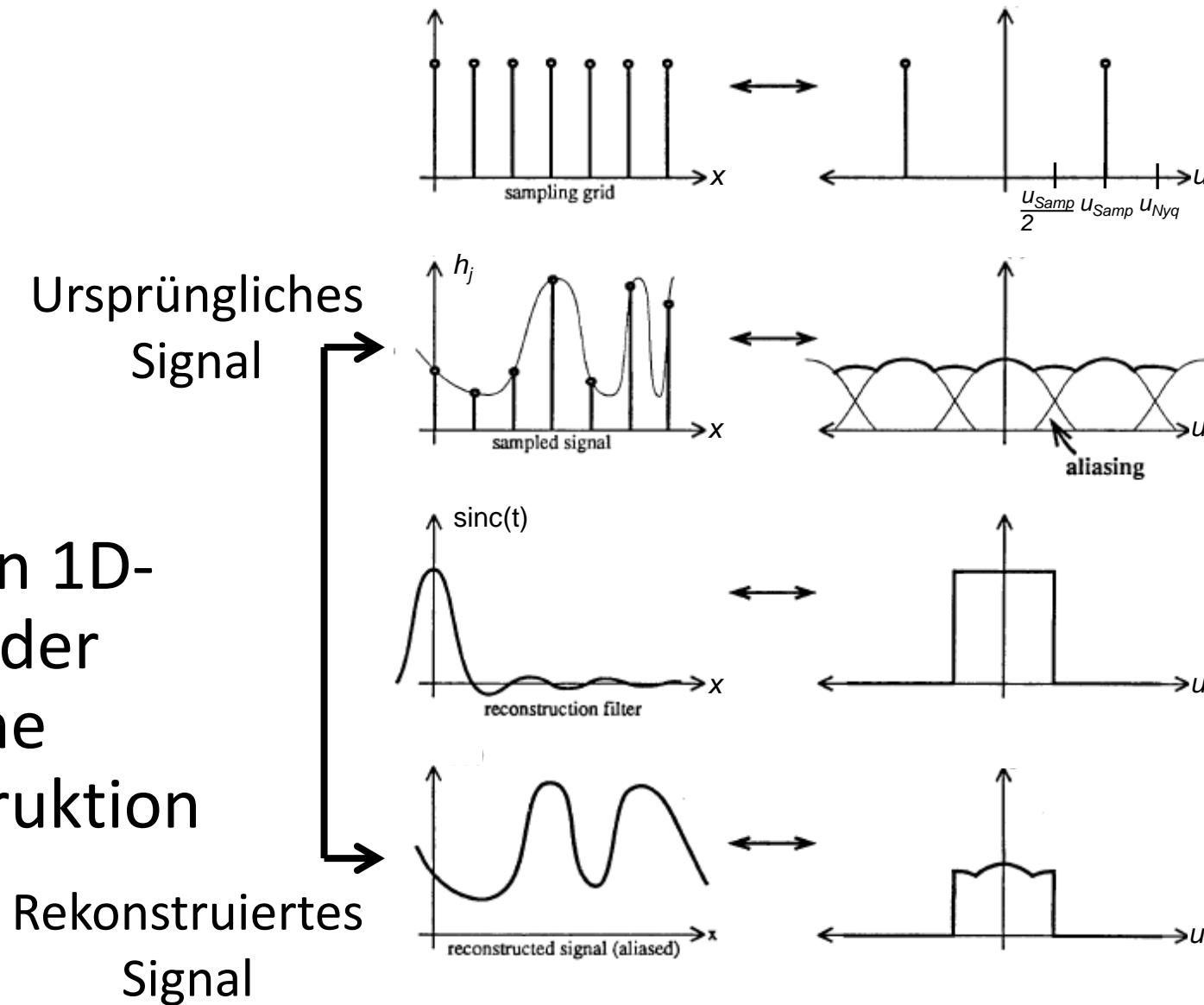
# Abtast-Theorem

- **Theorem** (Nyquist/Shannon): Wenn  $f(x)$  bandbegrenzt ist und oberhalb der Nyquist-Frequenz abgetastet wurde, kann es aus den Messwerten  $x_k=f(k\cdot\Delta x)$  exakt rekonstruiert werden.
  - Vernachlässigt Auswirkungen der Quantisierung
  - Geht bei endlicher Messdauer von einer periodischen Fortsetzung des Signals aus



# Gegenbeispiel: Unterabtastung

**Illustration:**  
In einem unterabgetasteten 1D-Signal verhindert der Aliasing-Effekt eine korrekte Rekonstruktion



# Zusammenfassung

- Das **Abtasttheorem** besagt, dass ein kontinuierliches Signal durch *Abtastung* in ein diskretes Signal überführt und durch *Interpolation* wieder exakt rekonstruiert werden kann
  - *Voraussetzung*: Bandbegrenzung  $B$  und Einhalten der **Nyquist-Frequenz**  $\omega=2B$
  - Interpolation durch Faltung mit dem Sinus Cardinalis
- Eine Unterschreitung der Nyquist-Frequenz führt zu **Aliasseffekten**, die sich z.B. in störenden Bildmustern äußern können
  - Beispiel in Kapitel 2.4

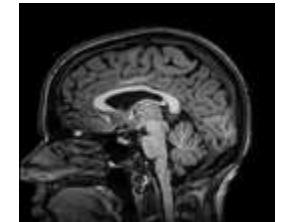
## **2.4 Resampling von Bildern**

# Motivation

- Eine Erhöhung der Bildauflösung bezeichnet man als **Upsampling**
  - *Beispiel:* Darstellung auf einem größeren Monitor
- Eine Reduzierung der Bildauflösung bezeichnet man als **Downsampling**
  - *Beispiel:* Sparen von Speicherplatz
- Bildtransformationen wie Rotationen oder Verschiebungen (um Bruchteile von Pixeln) erfordern ein **Resampling**
  - *Beispiel:* Bildregistrierung



Upsampling ↑



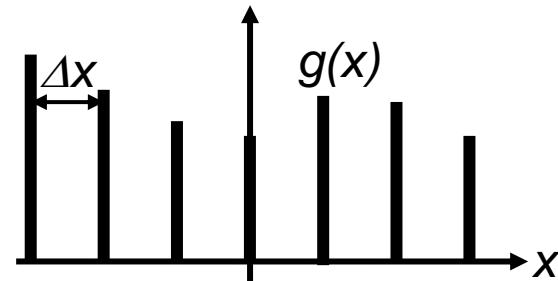
Downsampling ↓



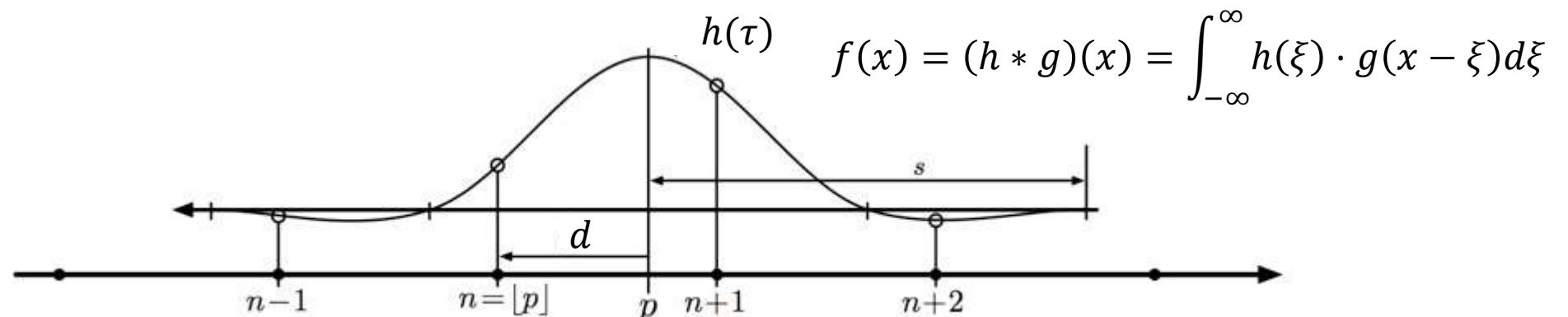
# Faltungsbasierte Rekonstruktion

- Ein diskretes Signal  $(x_i, y_i), i = 1, \dots, n$  lässt sich als gewichtete Summe von Dirac-Deltas schreiben:

$$g(x) = \sum_{i=1}^n y_i \delta(x - x_i)$$



- Ein kontinuierliches Signal  $f(x)$  erhalten wir durch Faltung von  $g(x)$  mit einem Kern  $h(x)$ 
  - Qualität des Ergebnisses und Rechenaufwand hängen vom Kern ab

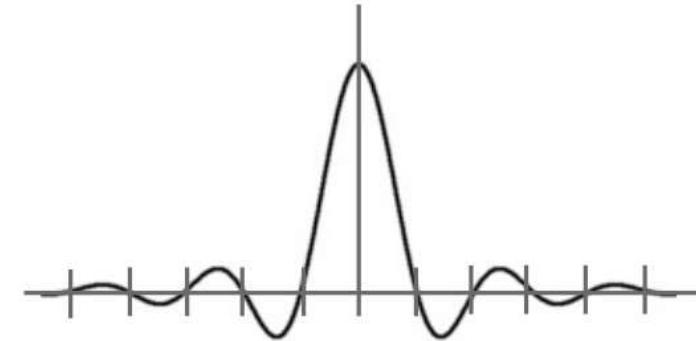


$$f(p) = \Delta x \times [y_{n-1}h(d + \Delta x) + y_nh(d) + y_{n+1}h(d - \Delta x) + y_{n+2}h(d - 2\Delta x)]$$

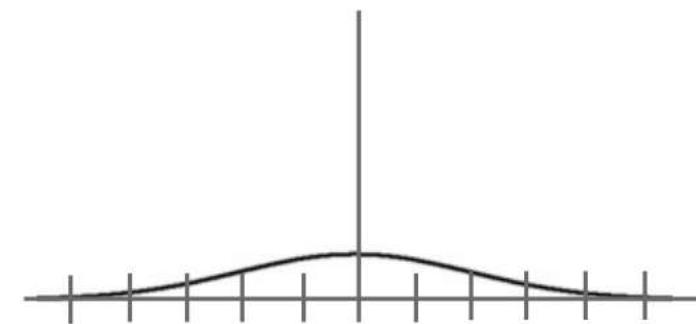
# Überlegungen zu Interpolations-Faltungskerne

- Laut **Abtasttheorem** ermöglicht Faltung mit dem Sinc eine exakte Rekonstruktion
  - *Nachteil:* Hoher Rechenaufwand
- In der Praxis zur Interpolation genutzte **Faltungskerne** sind ein Kompromiss zwischen
  - Rechenaufwand (möglichst kleiner Träger)
  - glatten Ergebnissen (erfordern Berücksichtigung der Nachbarschaft)
- Es folgen einige beliebte **Beispiele**

# Faltungskerne: Erste Ideen

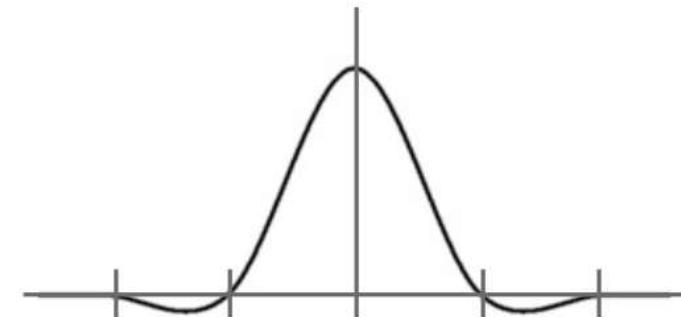


Hann-windowed sinc  
 $\text{sinc}(x) = \sin(\pi x)/(\pi x)$



Gaussian (stdv=2,  
cutoff = 4 stdv)

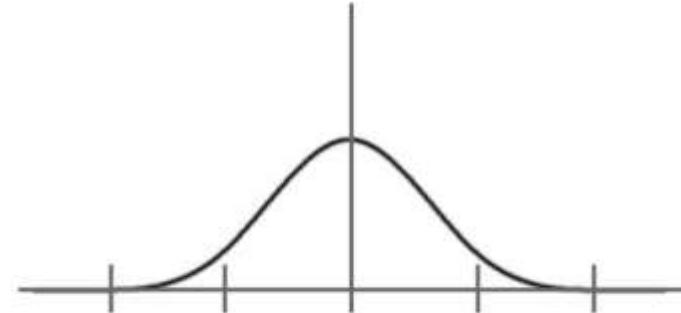
# Faltungskerne: Der Catmull-Rom-Spline



Interpolating cubic spline  
“Catmull-Rom”

$$w_{\text{crm}}(x) = \frac{1}{2} \begin{cases} 3|x|^3 - 5|x|^2 + 2 & \text{for } 0 \leq |x| < 1 \\ -|x|^3 + 5|x|^2 - 8|x| + 4 & \text{for } 1 \leq |x| < 2 \\ 0 & \text{for } |x| \geq 2 \end{cases}$$

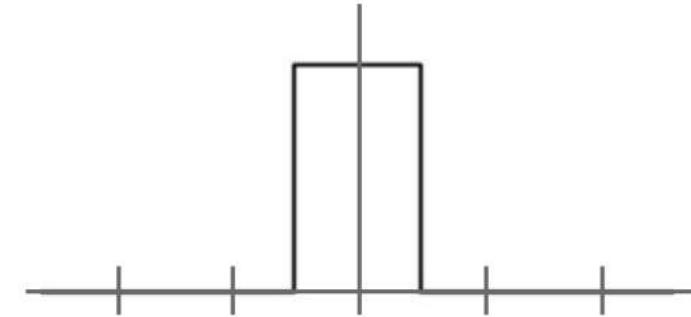
# Faltungskerne: Der kubische B-Spline



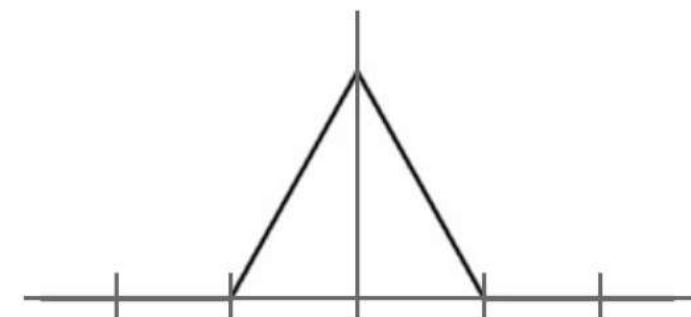
(non-interpolating)  
Cubic B-spline

$$w_{cbs}(x) = \frac{1}{6} \begin{cases} 3|x|^3 - 6|x|^2 + 4 & \text{for } 0 \leq |x| < 1 \\ -|x|^3 + 6|x|^2 - 12|x| + 8 & \text{for } 1 \leq |x| < 2 \\ 0 & \text{for } |x| \geq 2 \end{cases}$$

# Faltungskerne: Wenn es schnell gehen soll...



box, nearest neighbor



tent, linear

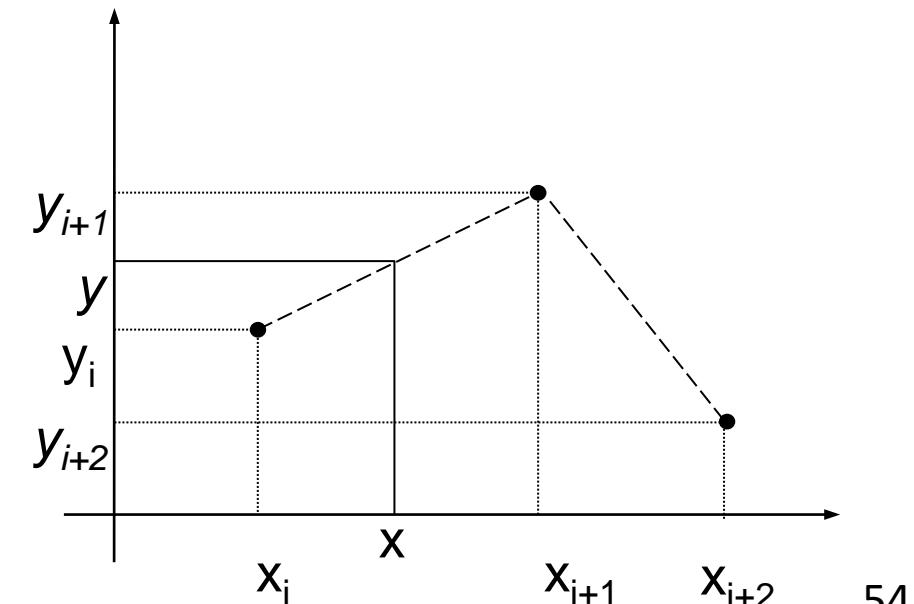
# Stückweise Lineare Interpolation

- **Stückweise lineare Interpolation** interpoliert diskrete Signale  $(x_i, y_i), i = 1, \dots, n$  einfach und effizient
  - Nachteil: An Messpunkten erreichen wir nur  $C^0$ -Stetigkeit
- **Vorgehen:** Zur Interpolation am Punkt  $x$  betrachten wir die Nachbarn  $x_i \leq x \leq x_{i+1}$  und berechnen

$$f(x) = (1 - \alpha)y_i + \alpha y_{i+1}$$

mit

$$\alpha = \frac{x - x_i}{x_{i+1} - x_i} \in [0, 1]$$



# Bilineare Interpolation in rechteckigen Pixeln

- Berechnung von  $f(x,y)$  mit **bilinearer Interpolation**:
  1. Lineare Interpolation entlang der oberen und unteren Kanten zur horizontalen Position  $x$

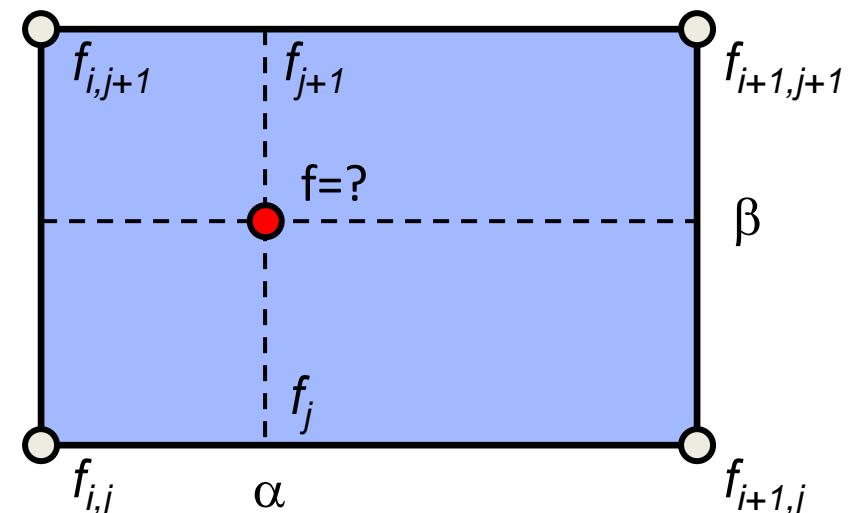
$$f_j = (1-\alpha) f_{i,j} + \alpha f_{i+1,j}$$

$$f_{j+1} = (1-\alpha) f_{i,j+1} + \alpha f_{i+1,j+1}$$

2. Erneute lineare Interpolation der Zwischenergebnisse zur vertikalen Position  $y$

$$f(x, y) = (1 - \beta) f_j + \beta f_{j+1}$$

$$\alpha = \frac{x - x_i}{x_{i+1} - x_i}, \quad \beta = \frac{y - y_i}{y_{i+1} - y_i},$$



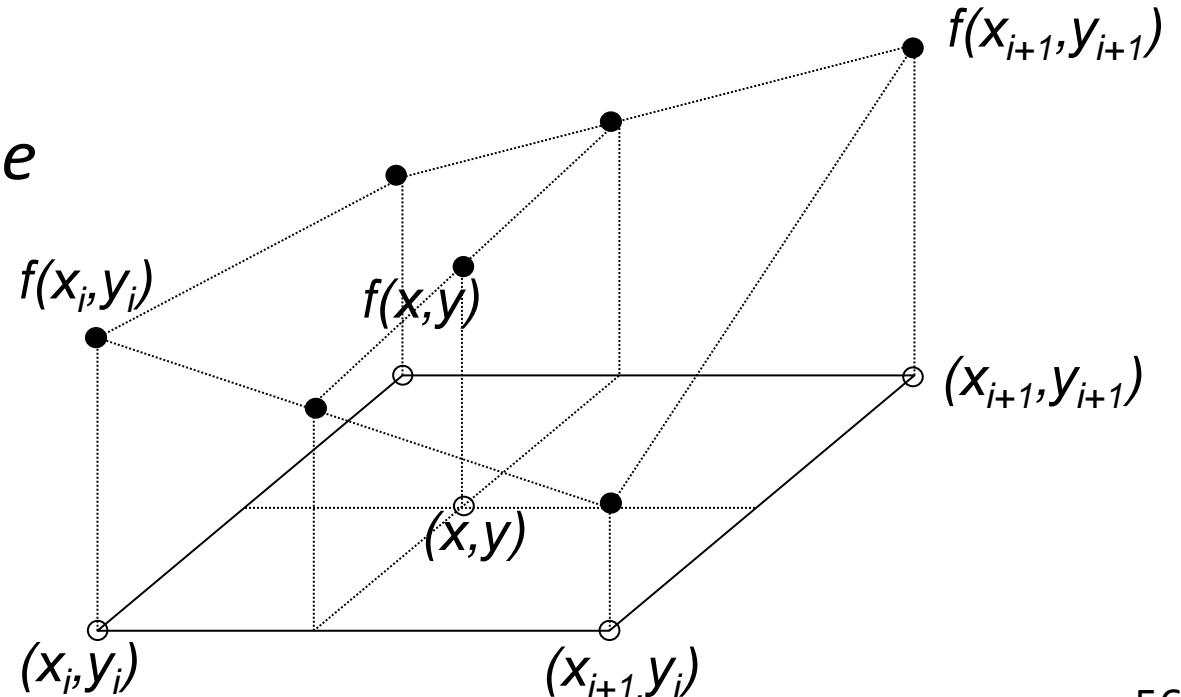
$$\alpha, \beta \in [0,1]$$

# Bilineare Interpolation: Reihenfolge?

- Einsetzen von  $f_j$  und  $f_{j+1}$  und Ausmultiplizieren ergibt:

$$f(x, y) = (1 - \alpha)(1 - \beta)f_{i,j} + \alpha(1 - \beta)f_{i+1,j} \\ + (1 - \alpha)\beta f_{i,j+1} + \alpha\beta f_{i+1,j+1}$$

- Es spielt keine Rolle, ob wir zuerst horizontal oder vertikal interpolieren
- Bilineare Interpolation ergibt *keine* lineare Funktion (Ebene)!

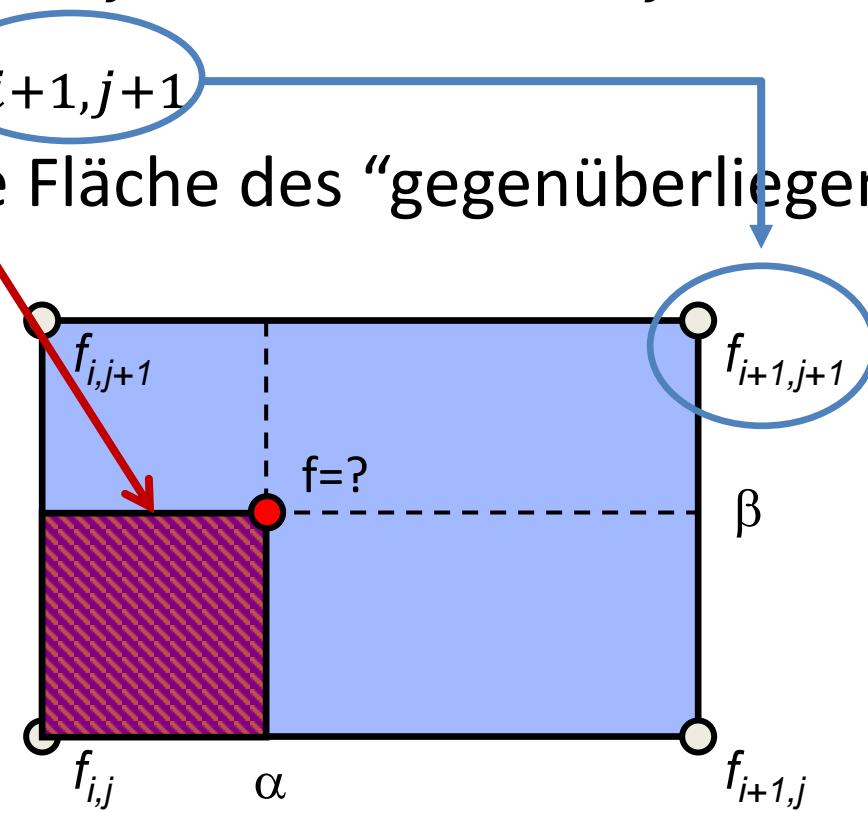


# Geometrische Interpretation

- Geometrische Interpretation der bilinearen Interpolation

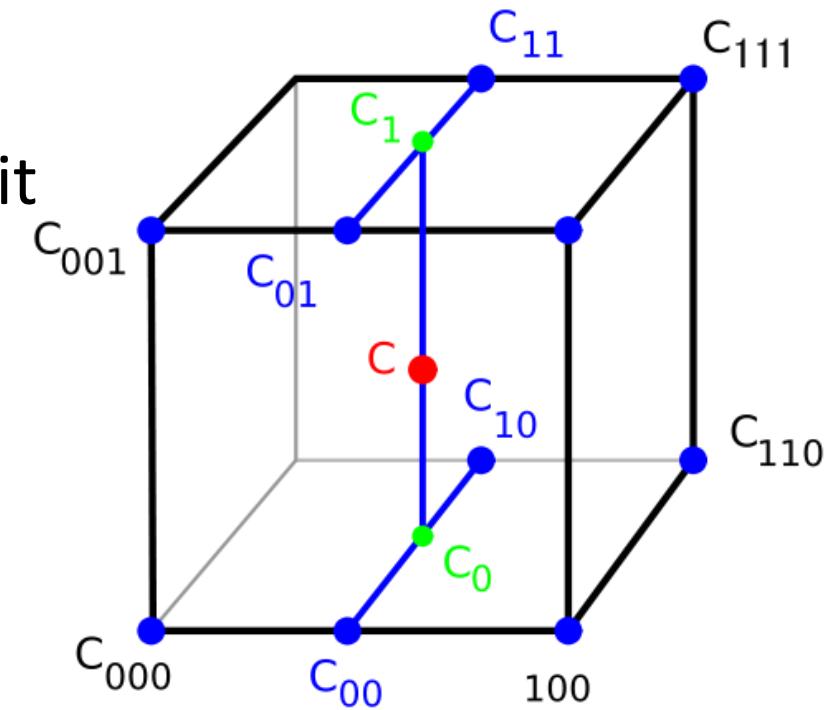
$$f(x, y) = (1 - \alpha)(1 - \beta)f_{i,j} + \alpha(1 - \beta)f_{i+1,j} + (1 - \alpha)\beta f_{i,j+1} + \alpha\beta f_{i+1,j+1}$$

- Wert an jeder Ecke wird durch die Fläche des “gegenüberliegenden” Rechtecks gewichtet



# Trilineare Interpolation

- **Trilineare Interpolation** ist die direkte Generalisierung von rechteckigen 2D-Pixeln auf quaderförmige 3D-Voxel
  - Drei lokale Koordinaten:  $\alpha, \beta, \gamma$
  - Ergibt eine kubische Funktion, keine lineare
  - Genau wie lineare Interpolation führen auch bi-/trilineare Interpolation nur zu  $C^0$ -Stetigkeit zwischen den Pixeln/Voxeln



# Downsampling

- Die einfachste Möglichkeit die Auflösung um Faktor  $k$  zu verringern ist es, nur jede  $k$ te Zeile und Spalte zu behalten
  - Problem: Aliasing-Artefakte wegen Verletzung des Abtasttheorems
  - Lösungen: Pooling (Mitteln über  $k \times k$  Pixel) oder Vorfilterung



Original



Nach naiver Reduktion  
um Faktor 2



Nach Pooling über  
 $2 \times 2$  Fenster

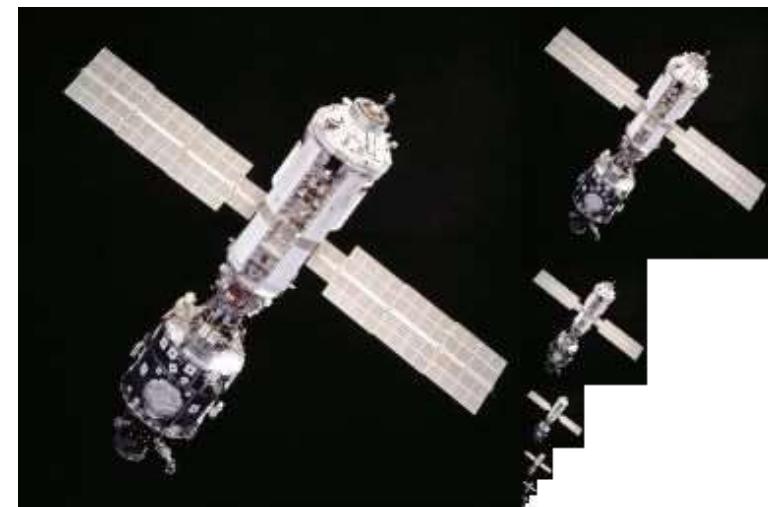
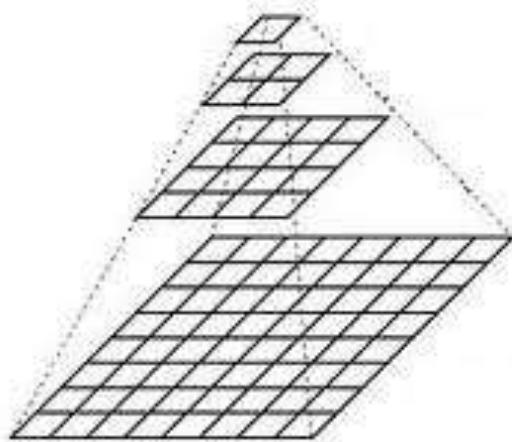


Nach Reduktion mit  
Vorfilterung

# Bildpyramiden

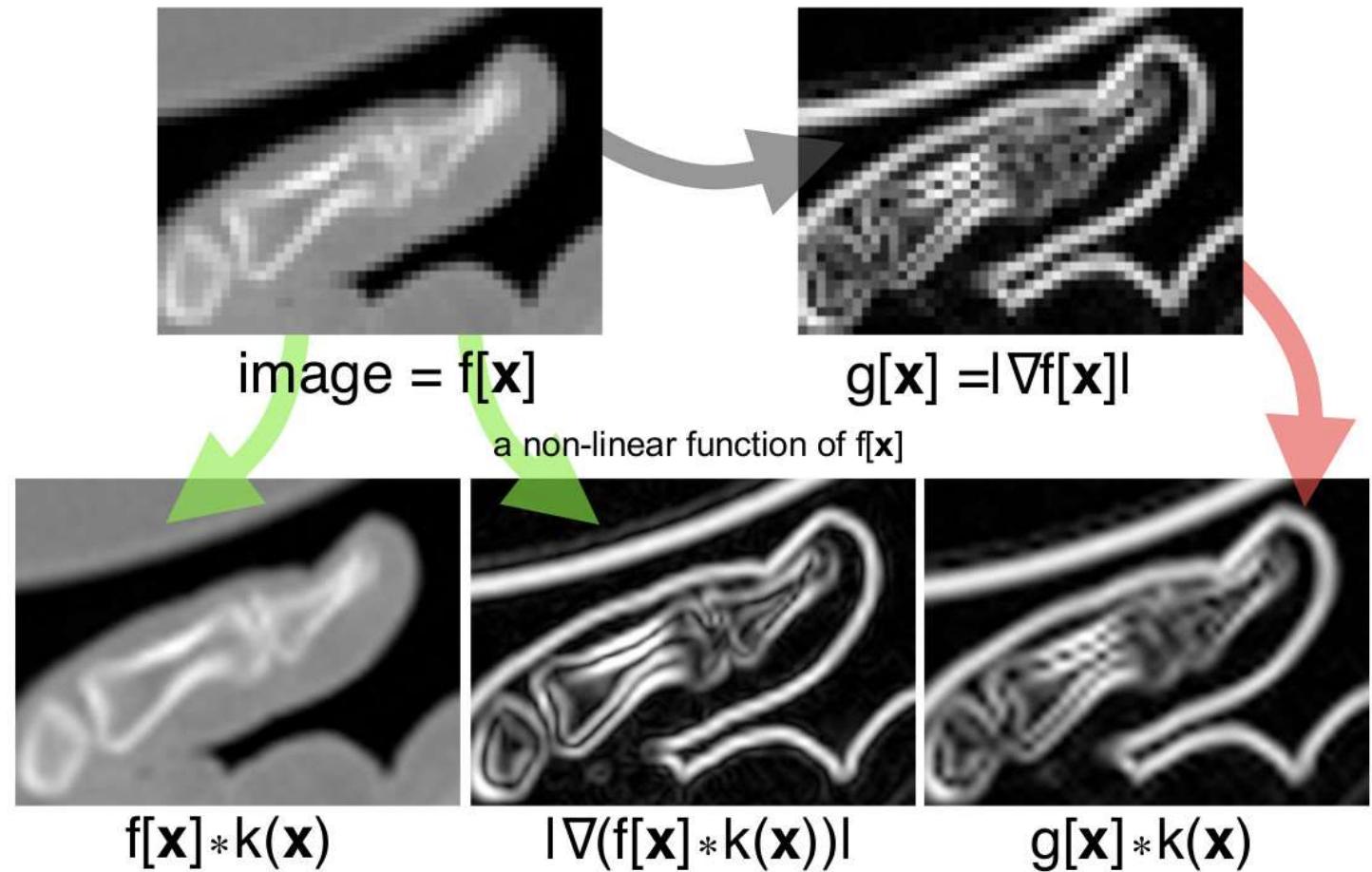
- **Bildpyramiden** werden dort eingesetzt, wo Bilder auf mehreren Skalen benötigt werden
  - Erzeugt durch wiederholtes Downsampling um den Faktor 2
  - Benötigt nur 33% zusätzlichen Speicherplatz:

$$\frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \dots = \frac{1}{3}$$



# Interpolation: Was gewinnen wir?

- Interpolation an sich zeigt Bilddetails im Prinzip nicht klarer
  - Information zu höheren Bildfrequenzen ist nicht vorhanden
- Vor nichtlinearen Transformationen ausgeführt kann sie trotzdem zu schärferen Ergebnissen führen
  - *Beispiel:* Kantenerkennung



# Superresolution / Super-Auflösung

Als **Superresolution** bezeichnet man Ansätze, die tatsächlich Information in höheren Frequenzbändern hinzufügen

- z.B. aus mehreren Bildern oder mittels Bilddatenbanken



Figure credit: Ledig et al, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network", arXiv 2016

# Zusammenfassung: Resampling

- **Faltungsbasierte Interpolation** ermöglicht die Rekonstruktion glatter Bilder
  - Flexibilität durch große Auswahl unterschiedlicher Kerne
  - Größe (Träger) des Kerns bestimmt wesentlich den Rechenaufwand
- **Lineare Interpolation** ist eine einfache und schnelle Alternative
  - Bi-/trilineare Interpolation in 2D- bzw. 3D-Bildern
- **Naives Downsampling** erzeugt Aliasing-Artefakte
  - Bessere Alternativen: Pooling oder Vorfilterung

## Zum Nach- und Weiterlesen

- W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery: *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 2<sup>nd</sup> edition, 1992
- R.E. Challis, R.I. Kitney: *Biomedical Signal Processing (in four parts).* Medical & Biological Engineering & Computing, 1990/91
- Andreas Maier et al. (Eds.): Medical Imaging Systems. Springer LNCS 11111, 2018 (*Open Access Book*)

# Quizfragen zu Kapitel 2

- Wie können wir anhand der Fourierkoeffizienten eines Signals erkennen, ob das Signal reellwertig ist?
- Erklären Sie anhand des Faltungssatzes, wie sich eine Glättung durch Faltung mit einem Gauß-Kern auf das Amplitudenspektrum des Signals auswirkt.
  - Wie wirkt es sich auf die Phasen der Fourierkoeffizienten aus?
- Warum können beim Downsampling von Signalen Aliasing-Artefakte auftreten?
  - Wie können wir das vermeiden?

## 3. Bildgebende Verfahren

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

4./11./18./25. November 2024

## **3.1 Röntgenbildgebung**

# Kurze Geschichte der Röntgenstrahlung

- Röntgenstrahlung wurde am 8. November 1895 von **Wilhelm Conrad Röntgen** in Würzburg entdeckt
  - Er selbst nannte sie „X-Strahlung“, englisch noch heute „X-ray“
  - 28.12.1895: „Über eine neue Art von Strahlen“
  - 1901: Röntgen erhält den ersten Nobelpreis in Physik überhaupt



Wilhelm Conrad Röntgen



Röntgenbild der Hand seiner Frau

# Klinische Anwendung: Radiographie

- Die **Radiographie** erzeugt mittels Röntgenstrahlung zweidimensionale Projektionsbilder



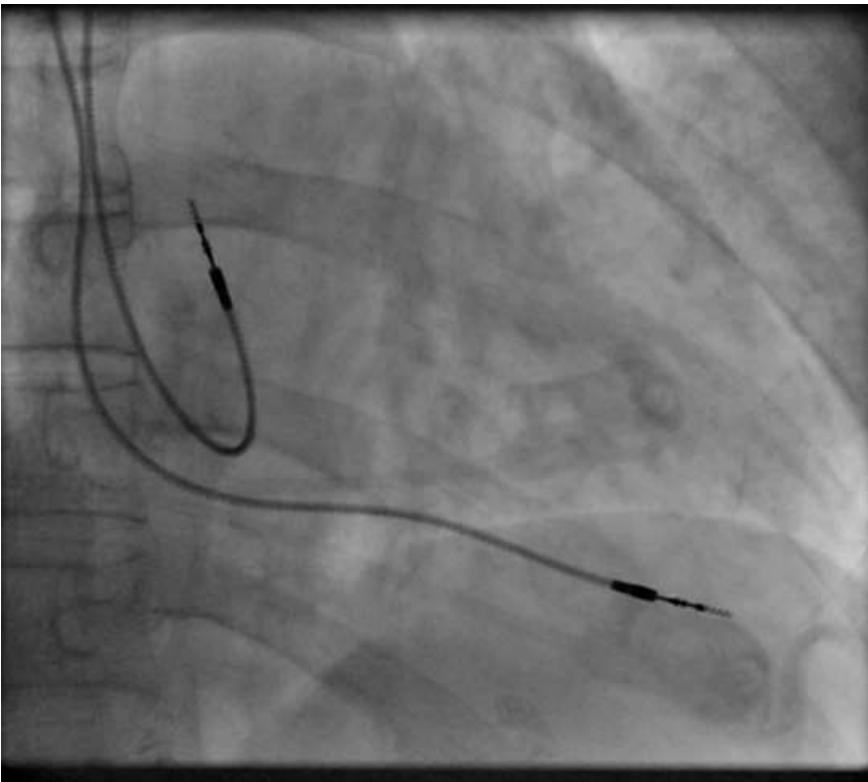
Diagnose einer Fraktur, Kontrolle des OP-Erfolgs



Diagnose einer Covid-Pneumonie

# Klinische Anwendung: Fluoroskopie

- Die **Fluoroskopie** (Durchleuchtung) beobachtet innere Vorgänge *kontinuierlich* mittels Röntgenstrahlung



Führungsdrähte zur Implantierung  
eines Herzschrittmachers



Barium-Breischluck-  
Untersuchung

# Klinische Anwendung: Subtraktionsangiographie

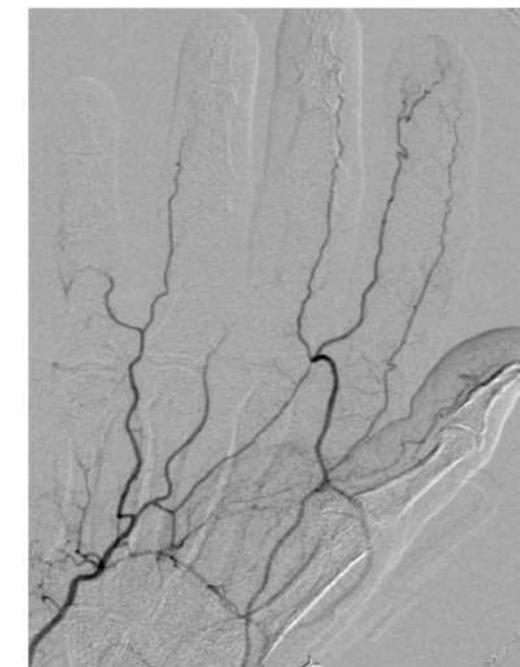
- Die **Angiographie** stellt Blutgefäße dar
  - Dies erfordert im Röntgen in der Regel ein **Kontrastmittel**
- Bei der **digitalen Subtraktionsangiographie** entsteht das Angiogramm durch Subtraktion einer **Leeraufnahme**



Leeraufnahme



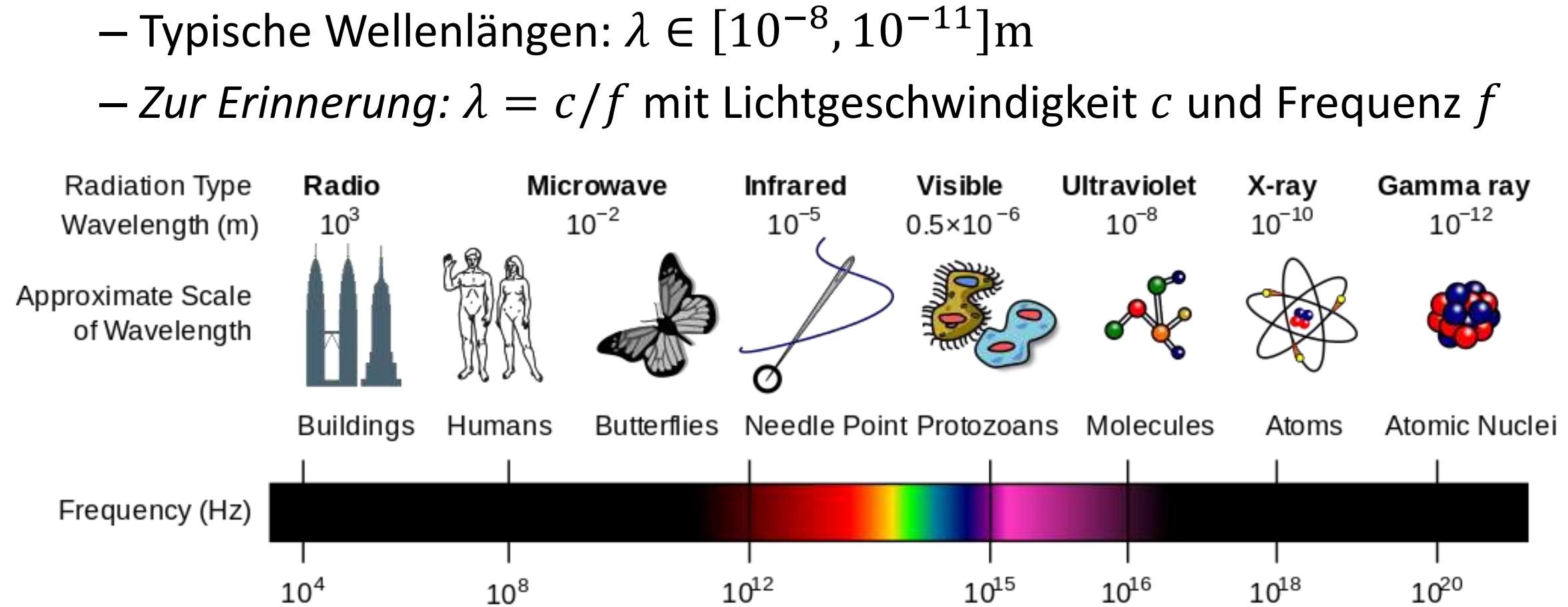
Mit Kontrastmittel



Differenzbild

# Was ist Röntgenstrahlung?

- Röntgenstrahlen sind **elektromagnetische Wellen** mit deutlich höherer Frequenz als sichtbares Licht
  - Typische Wellenlängen:  $\lambda \in [10^{-8}, 10^{-11}] \text{m}$
  - *Zur Erinnerung:  $\lambda = c/f$  mit Lichtgeschwindigkeit  $c$  und Frequenz  $f$*



# Energie der Röntgenstrahlung

- Statt der Frequenz oder Wellenlänge gibt man häufig die **Photoenergie** von Röntgenstrahlung an:

$$E = \frac{hc}{\lambda} = hf$$

- $h \approx 6,626 \times 10^{-34}$  Js ist das Plancksche Wirkungsquantum
- Die gebräuchliche Einheit dieser Energie ist **Elektronvolt (eV)**
  - $1\text{eV} \approx 1,602 \times 10^{-19}$  J ist die kinetische Energie, die ein Elektron beim Durchlaufen einer Beschleunigungsspannung von 1V gewinnt
  - Typische Energien im diagnostischen Bereich: [30,150] keV

# Gefahren der Röntgenstrahlung

- Röntgenstrahlung wirkt **ionisierend**
- Zu lange oder intensive Bestrahlung führt zu akuten Hautschäden (**Radiodermatitis**)
- Auch geringere Dosen können das Erbgut schädigen und langfristig die Entwicklung von **Krebs** begünstigen

**Fun fact:** Pedoskope sind in Deutschland seit 1973 verboten und wurden von *Time* zu den 100 dümmsten Ideen des 20. Jhd. gezählt



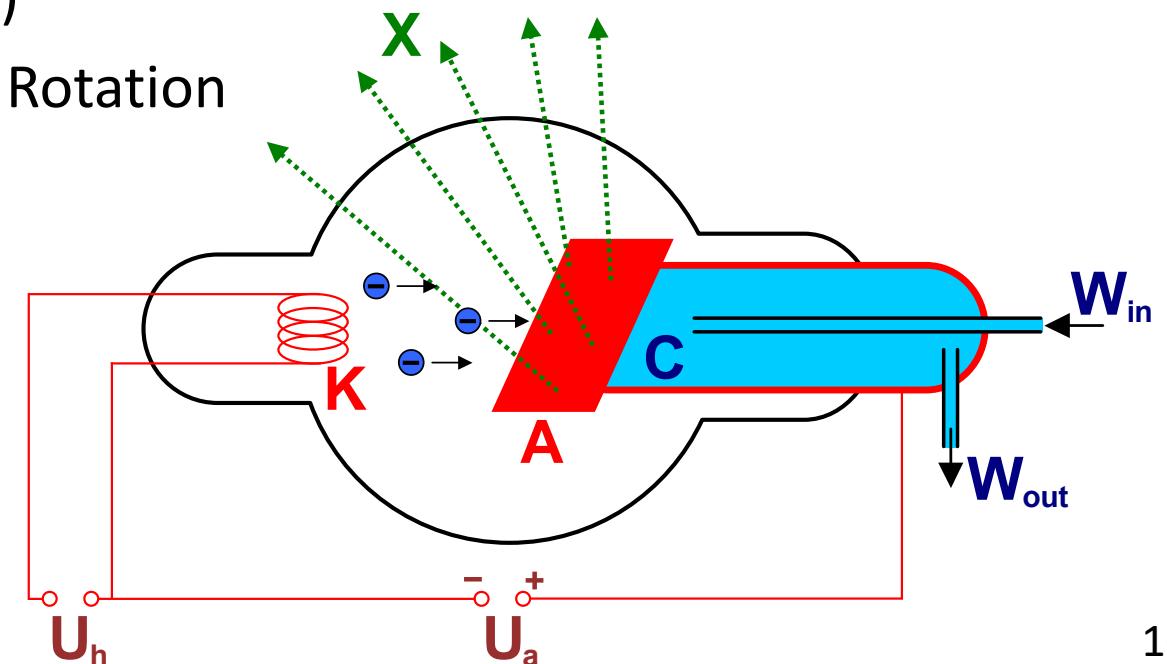
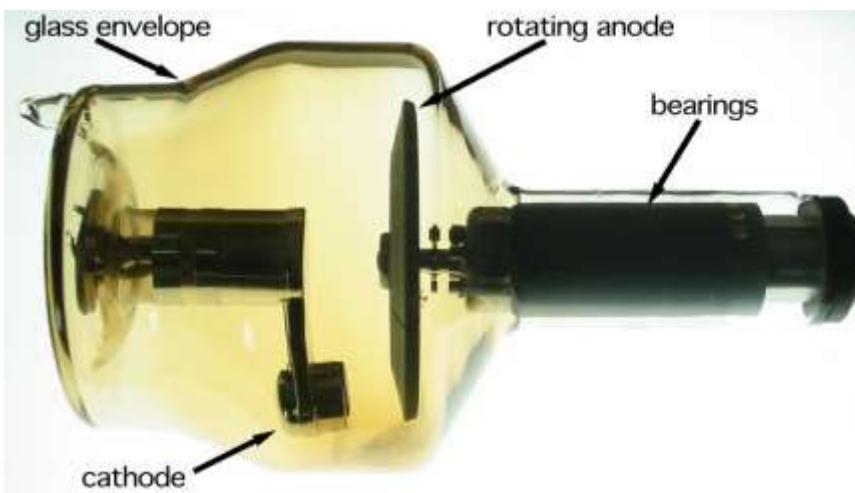
Symptome einer Radiodermatitis



Pedoskop der 1930er Jahre

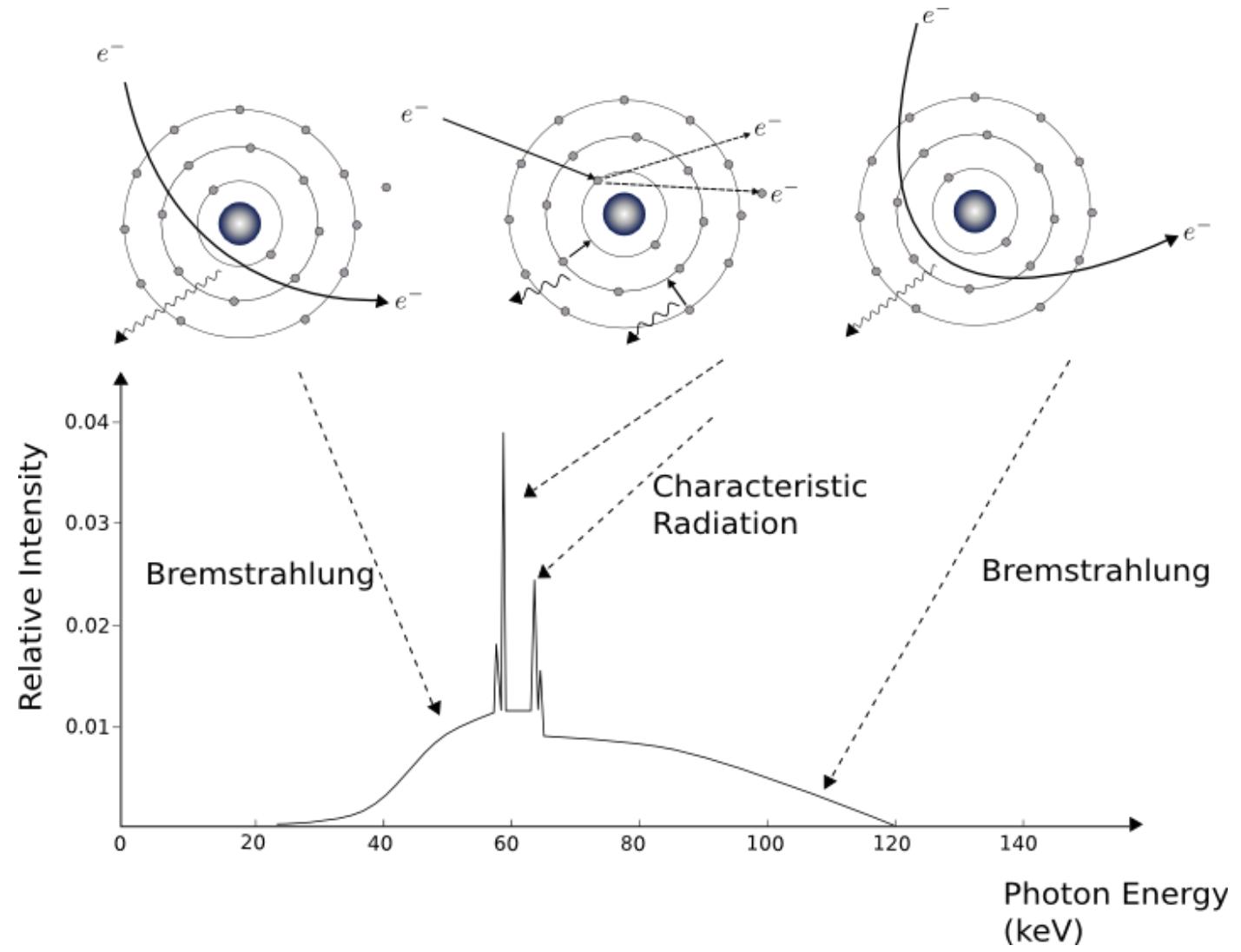
# Erzeugung von Röntgenstrahlung

- Röntgenstrahlung wird mittels **Röntgenröhren** erzeugt
  - Aus einer Kathode K in einer Vakuum-Röhre treten Elektronen aus
  - Diese werden mit Hochspannung  $U_a$  auf eine Anode A beschleunigt
  - Dort wird die kinetische Energie der Elektronen umgewandelt in
    - elektromagnetische Strahlung (<1%)
    - Wärme (>99%) – Abhilfe: Kühlung / Rotation



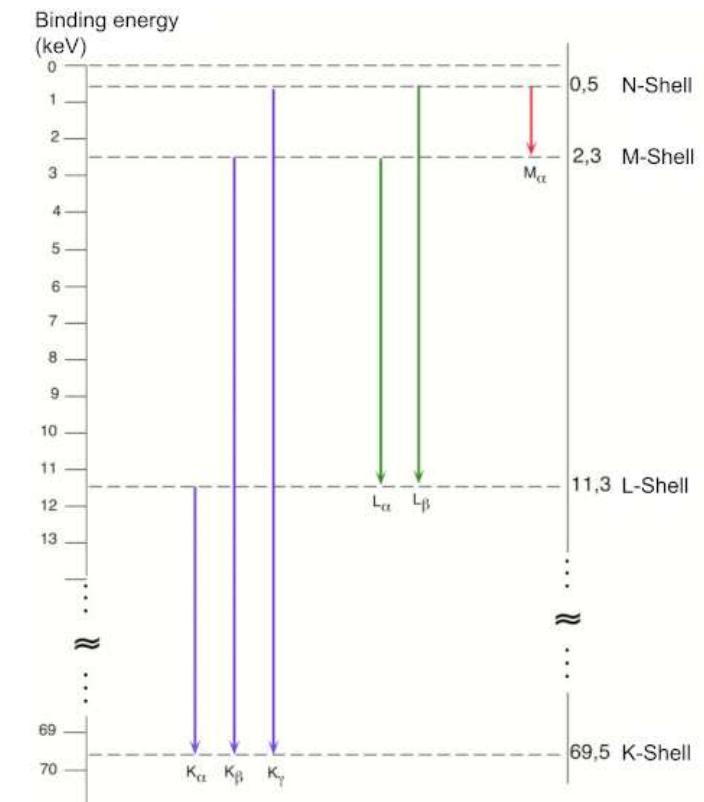
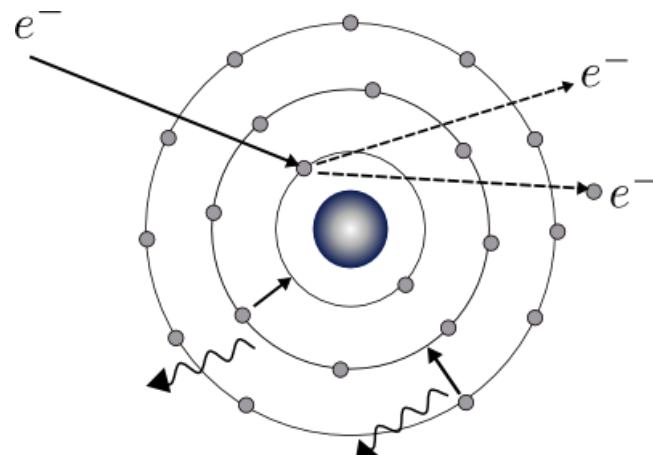
# Spektrum der Röntgenstrahlung

- Im Spektrum der Röntgenstrahlung überlagern sich
  - Das kontinuierliche Spektrum der Bremsstrahlung
  - Das diskrete Spektrum der charakteristischen Strahlung
- Zu geringe Energien werden herausgefiltert
  - z.B. mit einer dünnen Schicht Aluminium



# Charakteristische Strahlung

- Die **charakteristische Strahlung** entsteht, wenn ein gebundenes Elektron des Anodenmaterials aus einer inneren Schale herausgeschlagen wird und eines aus einer höheren Schale seinen Platz einnimmt
  - Die möglichen Energiedifferenzen haben feste Werte, die vom Material der Anode (z.B. Wolfram) abhängen



# Abschwächung durch Interaktion mit Materie

- Röntgenstrahlung interagiert mit der durchstrahlten Materie.
- Ihre Ursprungsintensität  $I_0$  wird dabei über den durch  $x$  parametrisierten Weg mit dem **Schwächungskoeffizienten**  $\mu$  abgeschwächt:

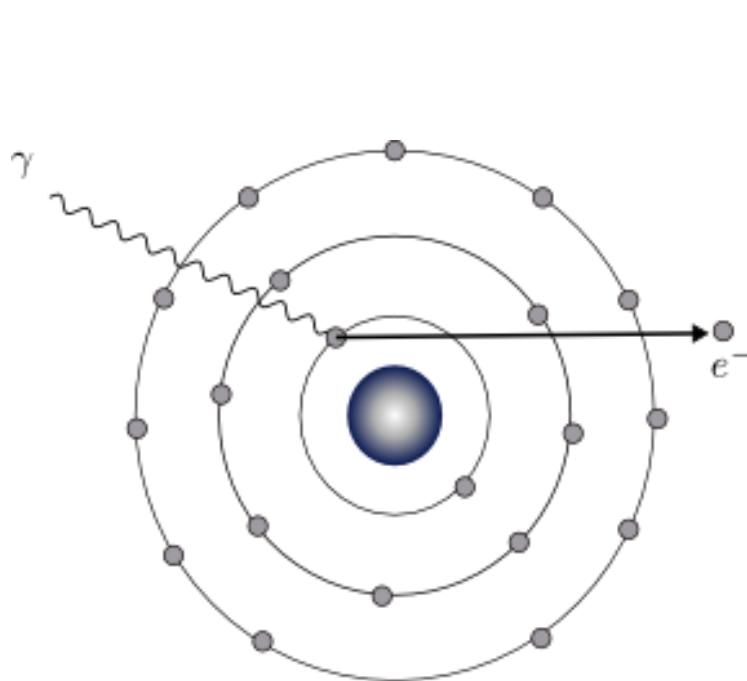
$$I = I_0 e^{- \int \mu(x) dx}$$

- $\mu$  hängt vom Material und seiner Dichte ab (z.B. Knochen, Muskeln). Dies erzeugt den diagnostisch relevanten **Bildkontrast**.
- Berücksichtigt man darüber hinaus die Abhängigkeit von der Energie  $E$  der Strahlung, ergibt sich:

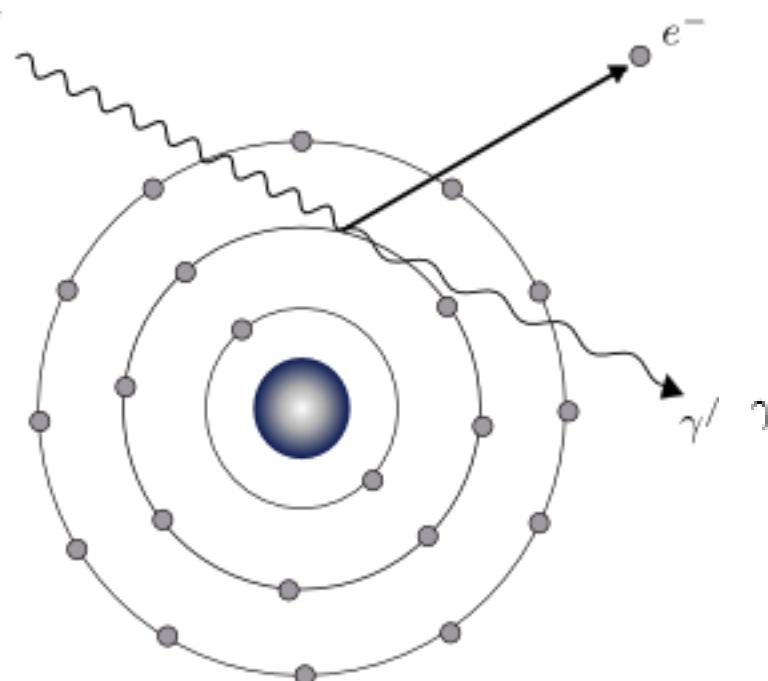
$$I = \int_0^{E_{\max}} I_0(E) e^{- \int \mu(x,E) dx} dE$$

# Arten der Interaktion mit Materie

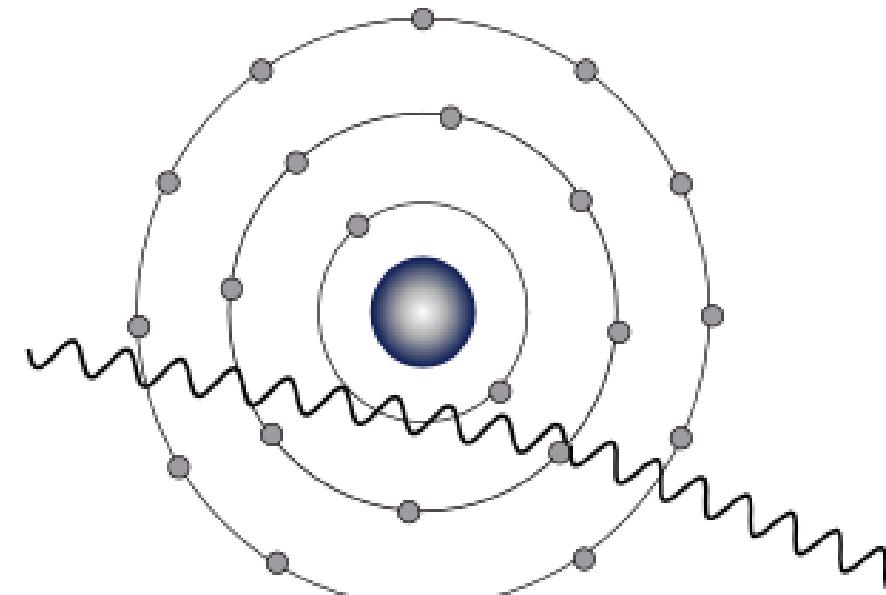
- Im diagnostischen Röntgen leisten folgende physikalische Effekte relevante Beiträge zum Schwächungskoeffizienten  $\mu$ :



Photoabsorption



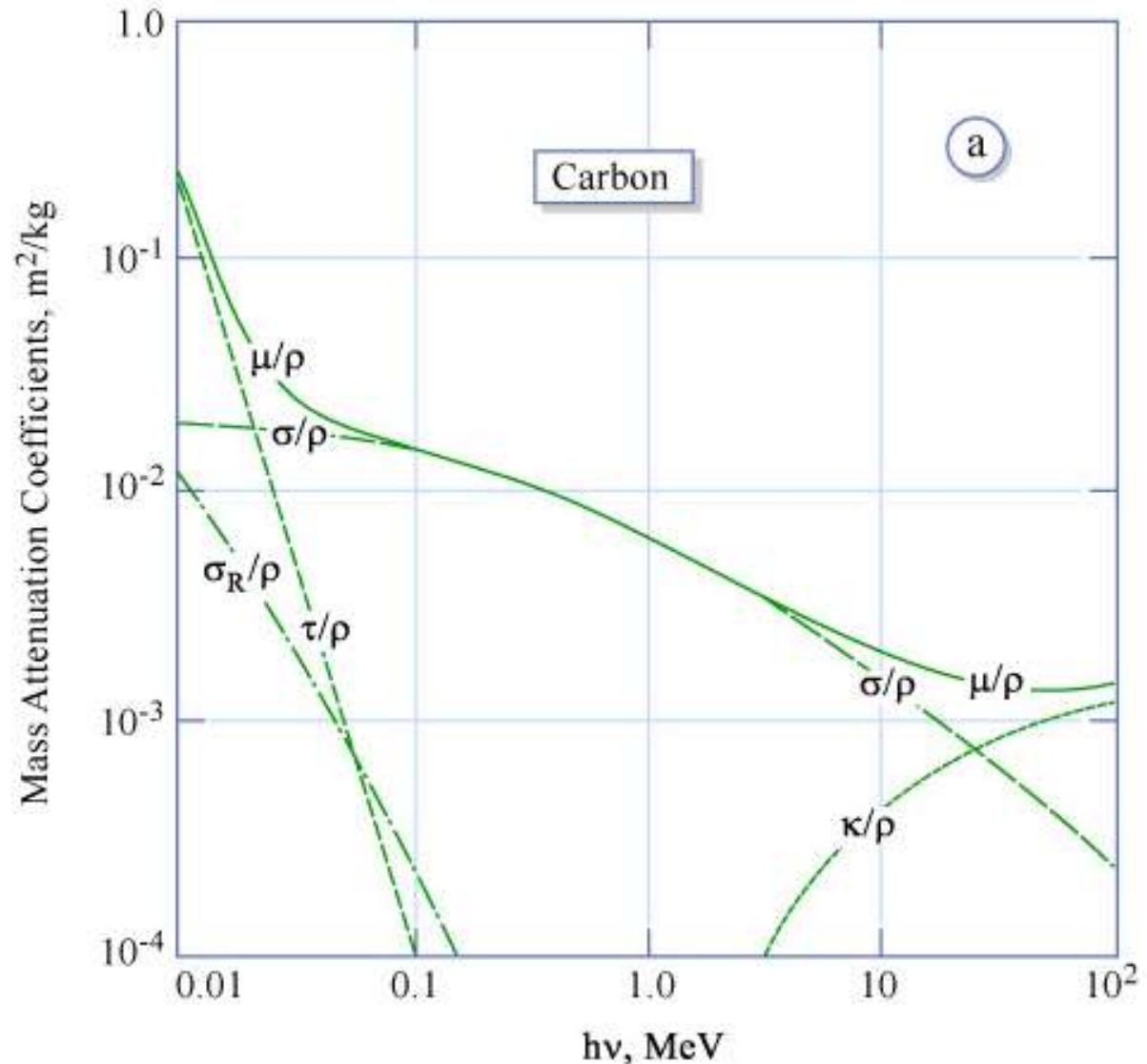
Compton-Streuung



Rayleigh-Streuung

# Beispiel: Schwächungsspektrum von Kohlenstoff

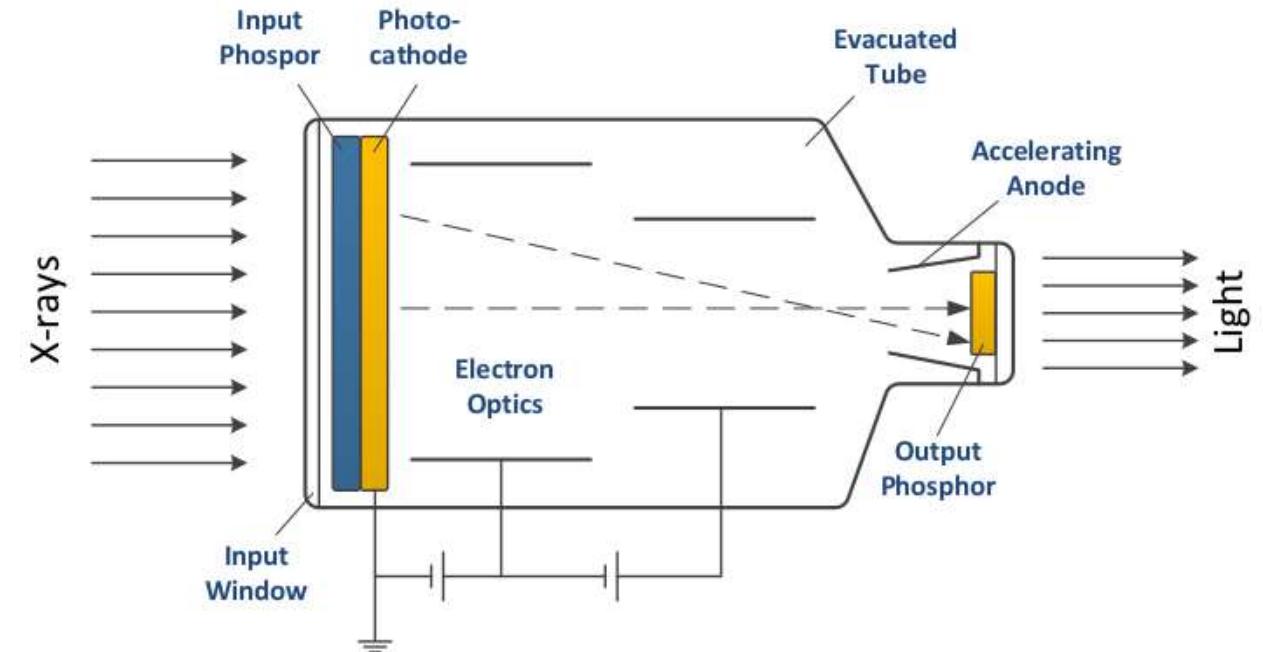
- Der **Massenschwächungskoeffizient**  $\mu/\rho$  berücksichtigt die Abhängigkeit des Schwächungskoeffizienten  $\mu$  von der Dichte  $\rho$  des Materials
  - $\tau$  Photoabsorption
  - $\sigma$  Compton-Effekt
  - $\sigma_R$  Rayleigh-Streuung
  - $\kappa$  Paarbildung
    - für uns hier irrelevant



# Röntgenbildverstärker

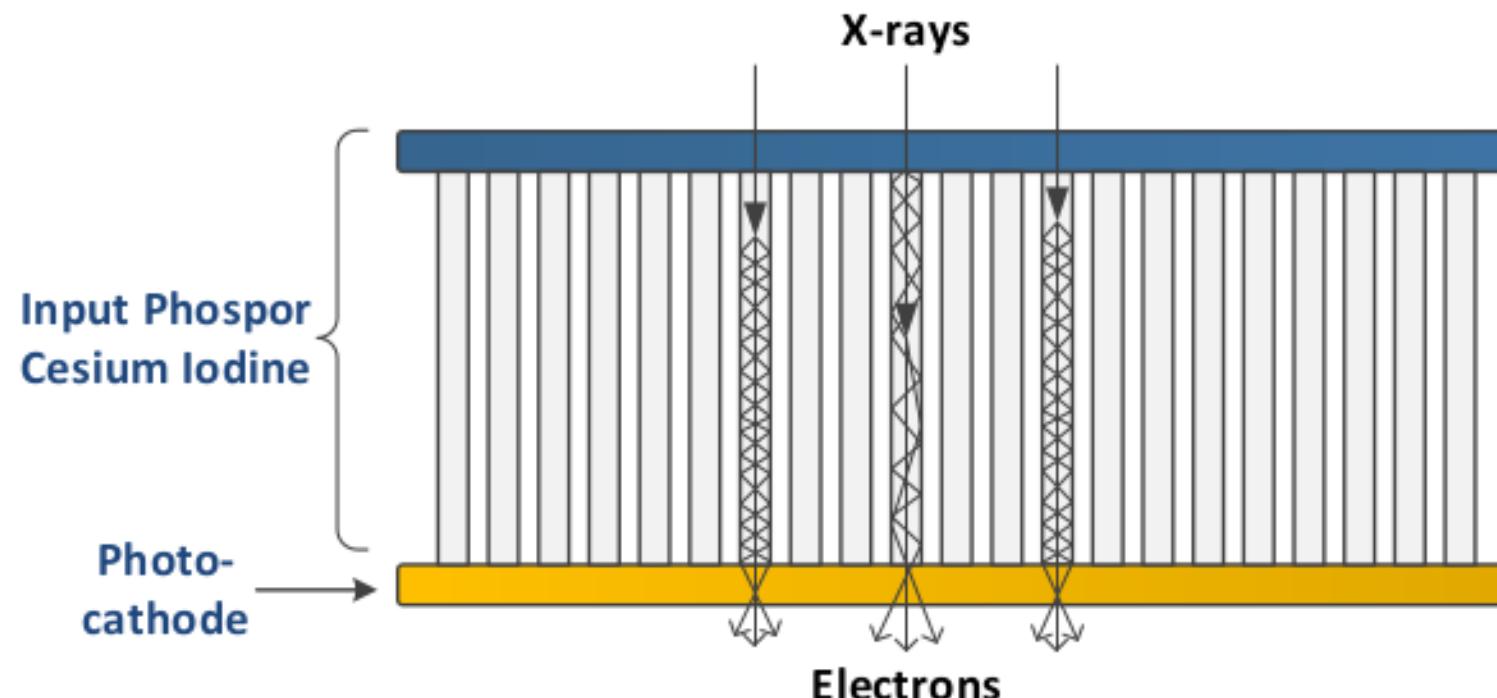
**Röntgenbildverstärker** reduzieren die benötigte Strahlendosis insbesondere bei kontinuierlicher Durchleuchtung

1. **Eingangsleuchtschirm** (Szintillator) wandelt Röntgenstrahlung in sichtbares Licht um
2. Dieses löst in **Photokathode** Elektronen aus, die im Vakuum mit 25-35kV beschleunigt werden
3. **Ausgangsleuchtschirm** erzeugt das finale Bild



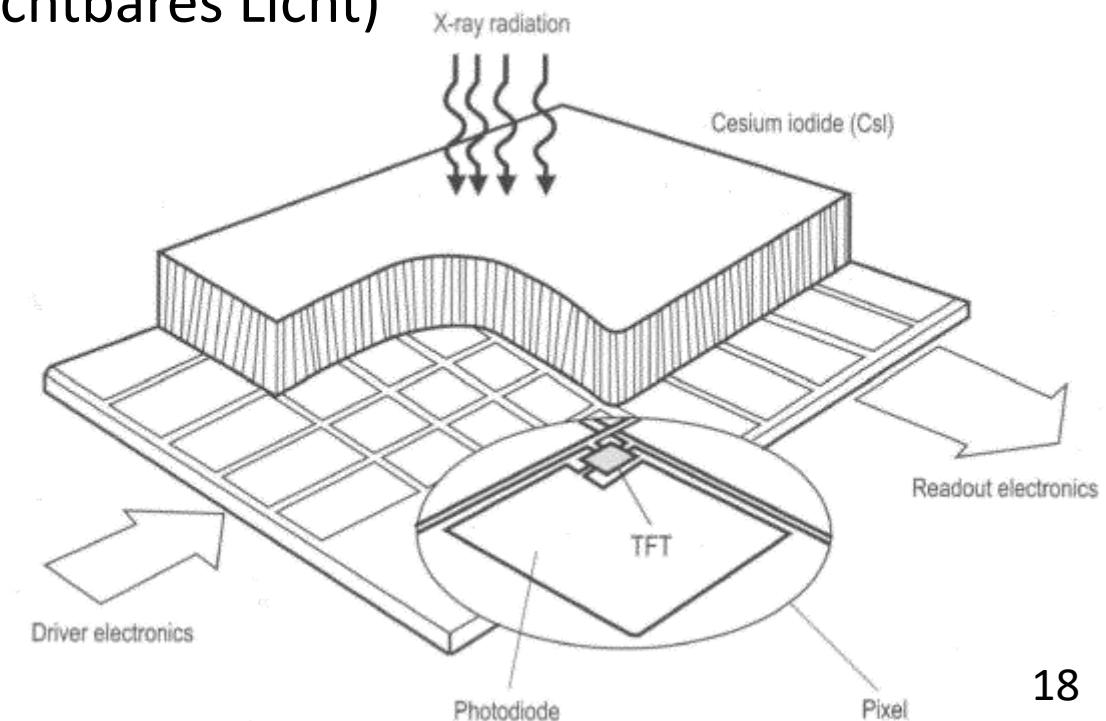
# Säulenstruktur von CsI-Szintillatoren

- Szintillatoren werden meist aus Cäsium-Jodid (CsI) gefertigt
  - Hohe Röntgenabsorption, hoher Konversionsgrad
  - Ermöglicht Herstellung von Säulenstrukturen, die trotz relativ dicker Leuchtschichten eine gute Ortsauflösung erhalten



# Digitale Röntgenbildgebung

- Stand der Technik sind **Flachbilddetektoren**, die digitale Röntgenbilder erzeugen
- Grundprinzip ähnlich wie bei **Digitalkameras**:
  1. Umwandlung elektromagnetischer Strahlung in elektrische Ladung
    - Direkt oder indirekt (Zwischenstufe: sichtbares Licht)
  2. Akkumulation dieser Ladung während der Belichtungszeit
  3. Auslesen, Verstärkung und Digitalisierung der Ladungen
    - Übliche Quantisierung: 12-16 Bit
    - „Binning“ beschleunigt das Auslesen bei reduzierter Bildauflösung



# Bildrauschen und -artefakte

- Die Bildqualität wird beeinträchtigt durch
  - **Rauschen**: Probabilistische Variabilität zwischen Aufnahmen
  - **Artefakte**: Systematische Fehler (z.B. Vignettierung = Abschwächung des Bildes am Rand)
- **Quellen von Bildrauschen** in der Röntgenbildgebung sind stochastische Prozesse bei der
  - Erzeugung von Röntgenstrahlung
  - Abschwächung von Röntgenstrahlung
  - Detektion von Röntgenstrahlung

# Zusammenfassung: Röntgen-Bildgebung

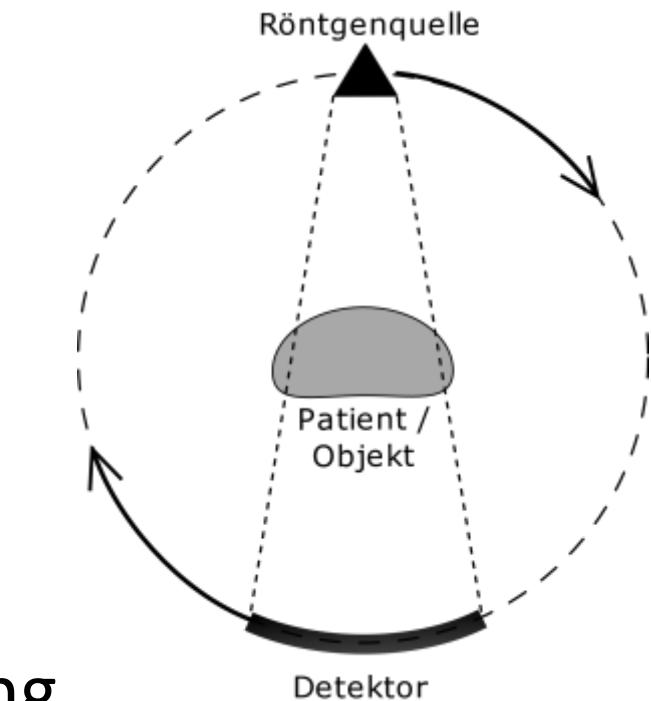
- **Röntgenbildgebung** ist eine klassische Möglichkeit Strukturen und Vorgänge im Körperinneren darzustellen
- **Röntgenstrahlen** sind besonders energiereiche elektromagnetische Wellen, die in **Röntgenröhren** erzeugt werden
  - Sie durchdringen den Körper, werden von verschiedenen Materialien unterschiedlich stark abgeschwächt (**Bildkontrast**)
- Die Abbildung erfolgt
  - historisch durch Filme, fluoreszierende Schirme, **Röntgenbildverstärker**
  - heute in der Regel durch digitale **Flachbilddetektoren**
- Stochastische Prozesse verursachen ein **Bildrauschen**

## **3.2 Computertomographie**

# Grundidee der Computertomographie

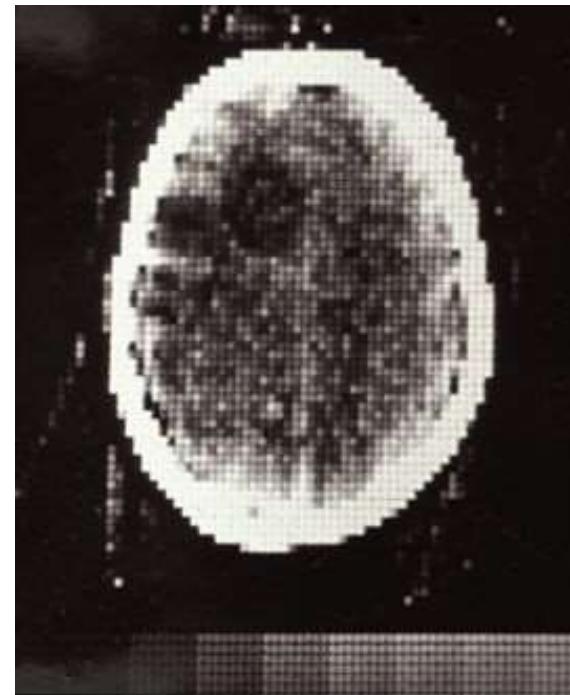
Grundidee der **Computertomographie**:  
Überlagerungsfreie Schicht- statt Schattenbilder

- Röntgen ermittelt Linienintegrale des Schwächungskoeffizienten (siehe 3.1)
- Aus Aufnahmen aus verschiedenen Richtungen können wir 2D-Schichtbilder  $f(x, y)$  rekonstruieren (inverse Radon-Transformation)
- Mehrere Schichtbilder ergeben eine 3D-Darstellung



# Geschichte der Computertomography (CT/CAT)

- 1957-63: **Allan Cormack** beschreibt die Grundlagen der Computertomographie
- 1969: **Sir Godfrey Hounsfield** entwickelt den ersten Prototypen
- 1971: Erste CT-Aufnahme eines Menschen
- 1972: Erster kommerzieller CT-Scanner
- 1979: Cormack und Hounsfield erhalten gemeinsam den **Nobelpreis für Physiologie und Medizin**



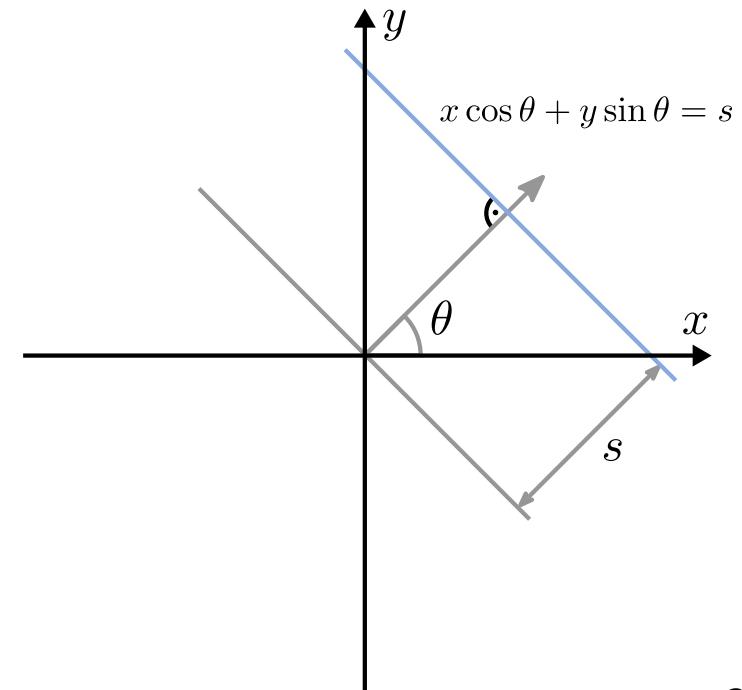
Erster klinischer CT-Scan



Modernes CT-Gerät

# Radon-Transformation

- **Johann Radon:** „Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten“ (1917)
  - Jede integrierbare Funktion  $f(x, y)$  wird eindeutig durch alle geraden Linienintegrale  $p(\theta, s)$  über ihr Definitionsgebiet beschrieben
  - Parametrisierung aller Geraden über
    - Winkel  $\theta \in [0^\circ, 180^\circ]$
    - Abstand  $s \in [-\infty, \infty]$  vom Ursprung

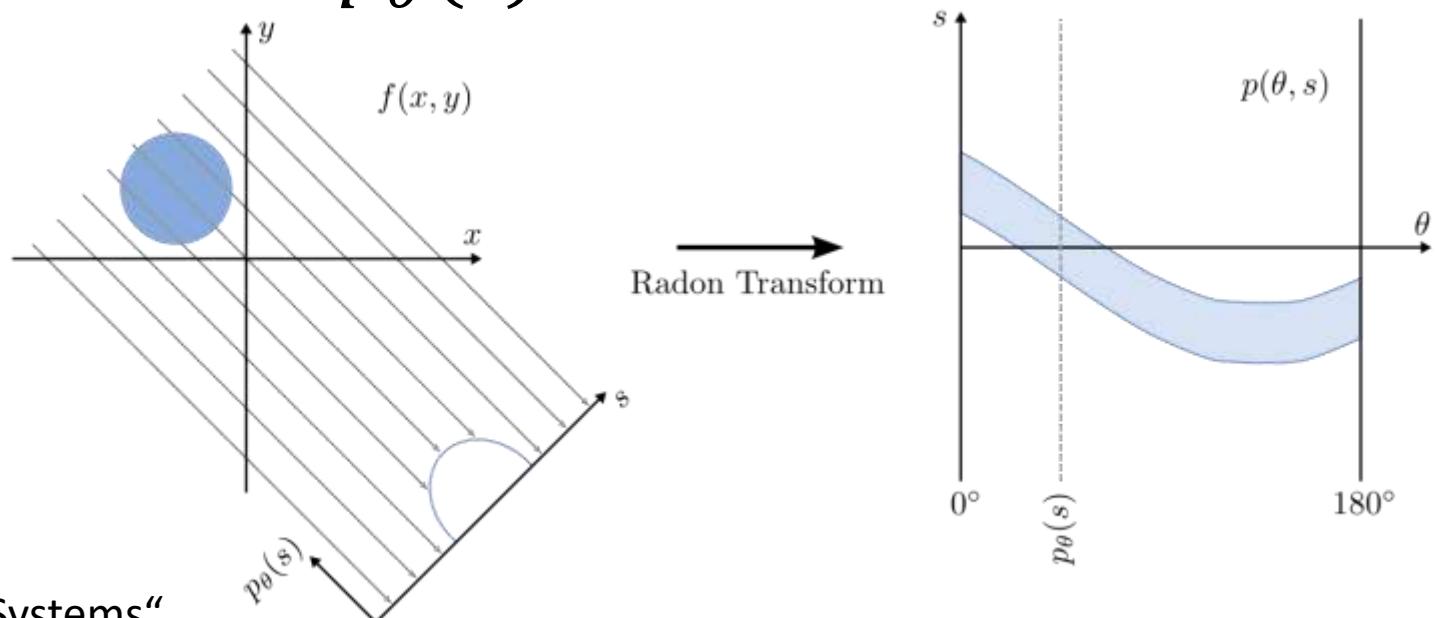


# Radon-Transformation: Formalisierung

- Formalisierung der **Radon-Transformation**  $p(\theta, s)$  von  $f(x, y)$ :

$$p(\theta, s) = \iint_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy$$

- Die graphische Darstellung von  $p(\theta, s)$  bezeichnet man als **Sinogramm**, einen Ausschnitt  $p_\theta(s)$  mit festem Winkel als **Projektion**

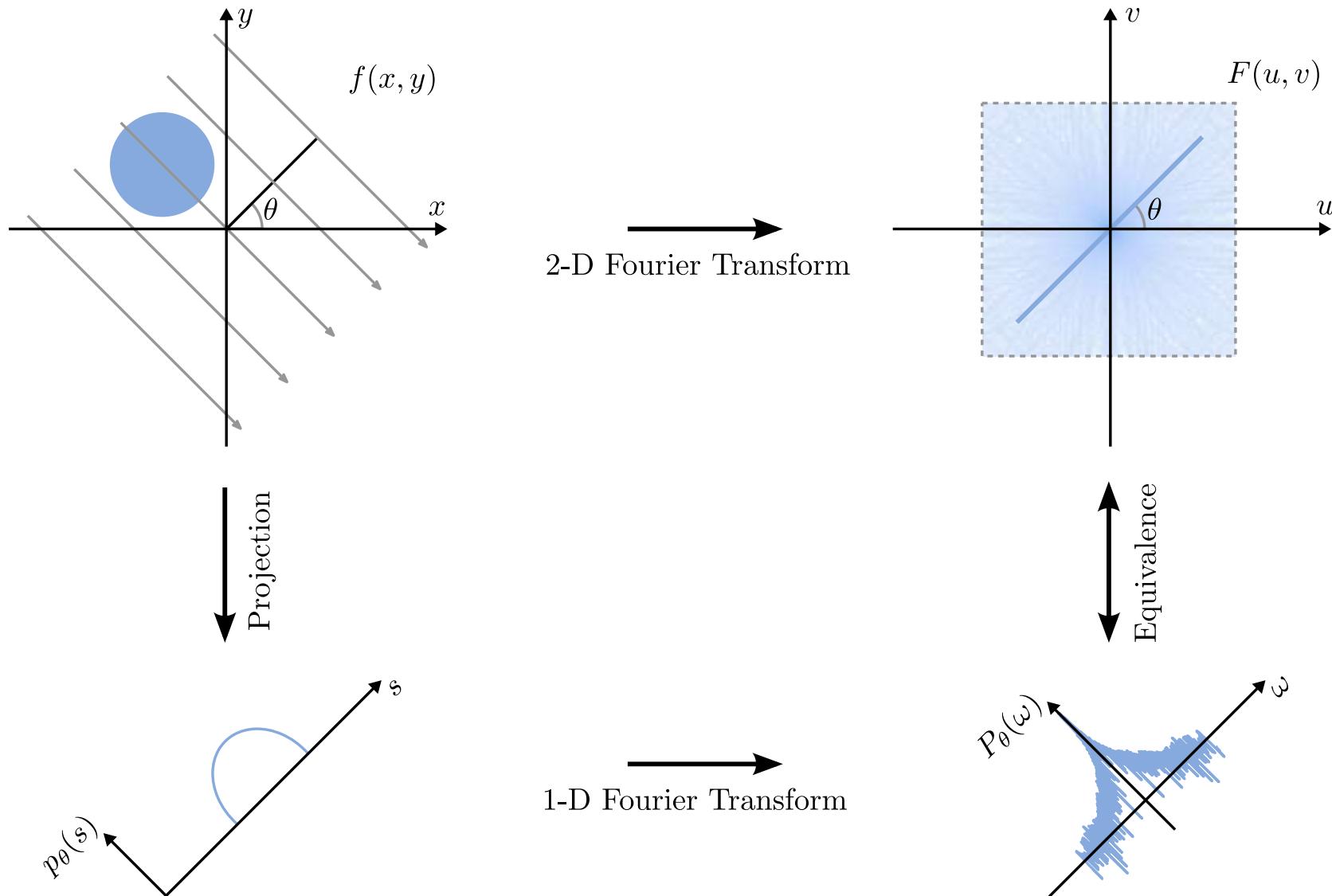


# Fourier-Scheiben-Theorem: Idee

- **Grundidee der CT-Rekonstruktion:** Aus den Projektionen können wir die Fouriertransformation  $F(u, v)$  des Schichtbilds bestimmen und diese dann invers transformieren
- Dabei hilft das **Fourier-Scheiben-Theorem** (*auch Fourier-Schnitt-Theorem*). Es beschreibt das Verhältnis zwischen
  - 2D-Fouriertransformation des Bildes  $F(u, v) = \mathcal{F}\{f(x, y)\}$
  - 1D-Fouriertransformation einer Projektion  $P_\theta(\omega) = \mathcal{F}\{p_\theta(s)\}$

$P_\theta(\omega)$  entspricht genau einer “Scheibe” von  $F(u, v)$  entlang einer Geraden durch den Ursprung mit Winkel  $\theta$

# Fourier-Scheiben-Theorem: Illustration



# Fourier-Scheiben-Theorem: Herleitung

- Wir überprüfen das FST zunächst durch Einsetzen der Projektion  $p_0(s)$  mit Winkel  $\theta = 0$  (auf die x-Achse)

$$p_0(s) = p_0(x) = \int_{-\infty}^{\infty} f(x, y) dy$$

in die 1D-Fouriertransformation:

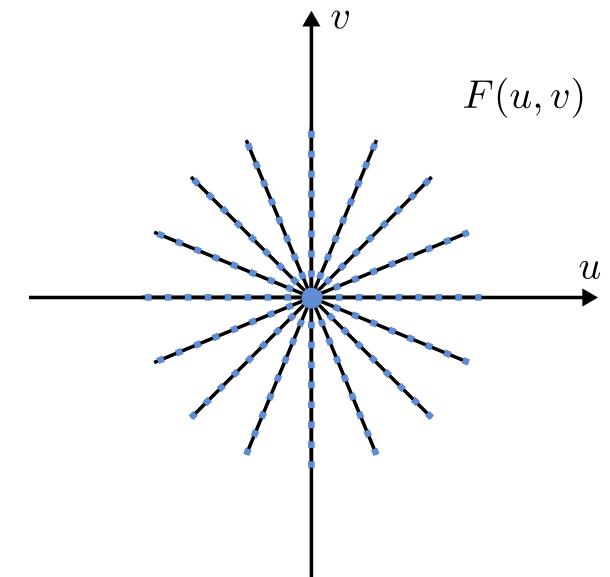
$$\begin{aligned} P_0(u) &= \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} f(x, y) dy \right] e^{-2\pi i ux} dx \\ &= \iint f(x, y) e^{-2\pi i(ux+0y)} dx dy = F(u, 0) \end{aligned}$$

- Für alle anderen Winkel  $\theta$  ergibt sich dieselbe Rechnung in einem entsprechend gedrehten Koordinatensystem

# Analytische CT-Rekonstruktion: Grundidee

- **Grundidee:** Die 1D-fouriertransformierten Projektionen  $P_\theta(\omega)$  lassen sich zu einer vollständigen Darstellung  $F_p(\omega, \theta)$  des Schichtbildes im Frequenzraum zusammensetzen
  - $\omega, \theta$  sind Polarkoordinaten im Frequenzraum
- Aus dem Frequenzraum können wir das Schichtbild durch eine inverse 2D-Fouriertransformation rekonstruieren:

$$f(x, y) = \iint F(u, v) e^{2\pi i(ux+vy)} du dv$$



# Inverse 2D-Fouriertransformation in Polarkoordinaten

- Beim Übergang von kartesischen in Polarkoordinaten  $u = \omega \cos \theta, v = \omega \sin \theta$  erhalten wir die Jacobi-Determinante

$$\det J = \det \begin{pmatrix} \frac{du}{d\omega} & \frac{du}{d\theta} \\ \frac{dv}{d\omega} & \frac{dv}{d\theta} \end{pmatrix} = \det \begin{pmatrix} \cos \theta & -\omega \sin \theta \\ \sin \theta & \omega \cos \theta \end{pmatrix}$$
$$= \omega \cos^2 \theta + \omega \sin^2 \theta = \omega$$

und somit das umgeformte Integral

$$f(x, y) = \int_0^{2\pi} \int_0^\infty F_p(\omega, \theta) \omega e^{2\pi i \omega(x \cos \theta + y \sin \theta)} d\omega d\theta$$

# Analytische CT-Rekonstruktion: Finale Form

$$f(x, y) = \int_0^{2\pi} \int_0^{\infty} F_p(\omega, \theta) \omega e^{2\pi i \omega(x \cos \theta + y \sin \theta)} d\omega d\theta$$

- Wir ersetzen nun  $F_p(\omega, \theta) = P_\theta(\omega)$  und  $x \cos \theta + y \sin \theta = s$  und passen die Integrationsgrenzen an unsere bisherige Konvention  $\theta \in [0, \pi), \omega \in (-\infty, \infty)$  an:

$$f(x, y) = \int_0^{\pi} \int_{-\infty}^{\infty} P_\theta(\omega) |\omega| e^{2\pi i \omega s} d\omega d\theta$$

# Interpretation als Gefilterte Rückprojektion

- Das innere Integral

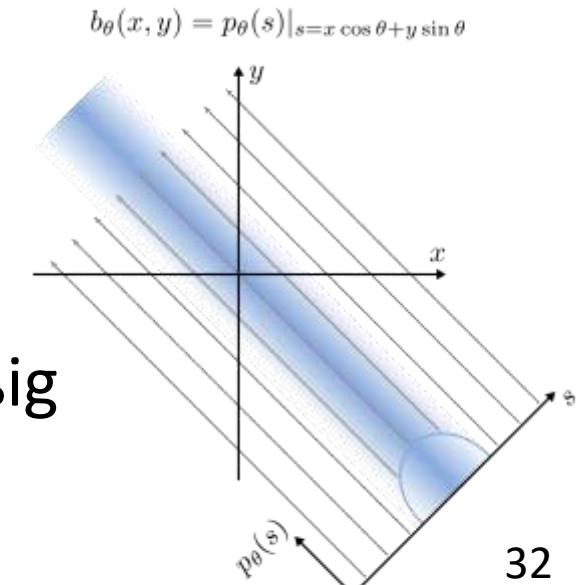
$$\tilde{p}_\theta(s) = \int_{-\infty}^{\infty} P_\theta(\omega) |\omega| e^{2\pi i \omega s} d\omega$$

können wir als **Filterung** der Projektion  $p_\theta(s)$  auffassen

- Multiplikation der Fourier-Transformierten  $P_\theta(\omega)$  mit  $|\omega|$  entspricht einer Faltung von  $p_\theta(s)$  mit dem Kern  $\mathcal{F}^{-1}\{|\omega|\}$  (Faltungssatz)

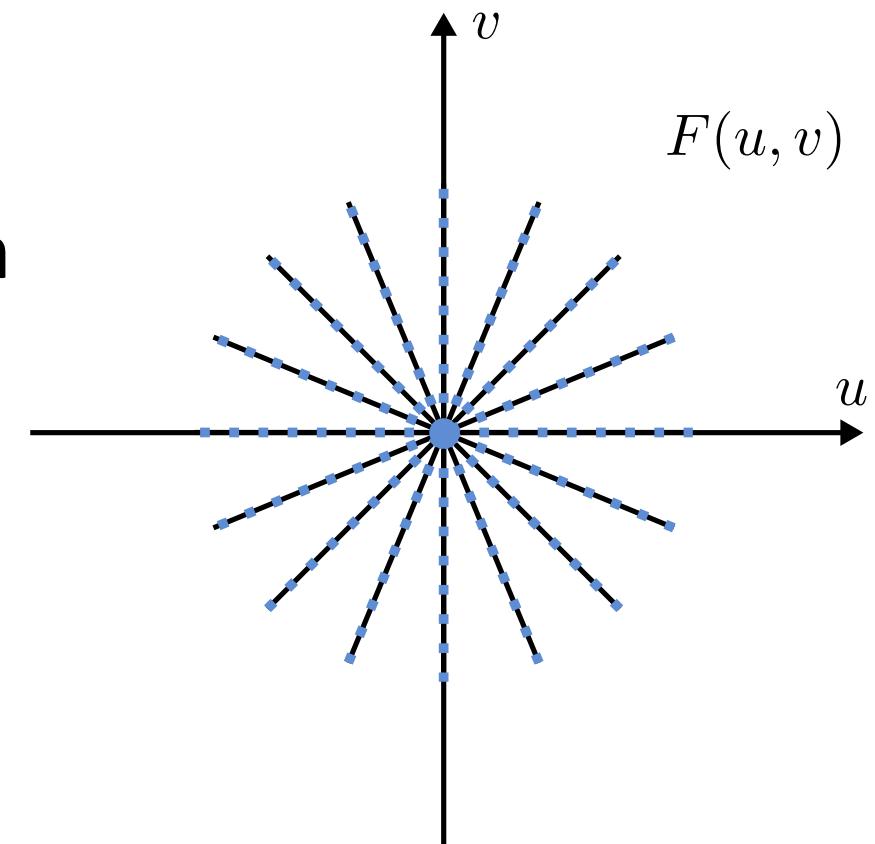
- Das Gesamtintegral  $f(x, y) = \int_0^\pi \tilde{p}_\theta(s) d\theta$  entspricht demnach einer Summe aller **gefilterter Rückprojektionen**

- *Vorstellung:* Verteile Werte jeder Projektion gleichmäßig auf alle Orte, die darin eingegangen sind



# Anschaauung: Notwendigkeit der Filterung

- Mathematisch folgt die Filterung aus der Jacobi-Determinante  $|\omega|$  die beim Übergang in Polarkoordinaten auftritt.
- Intuitiv gleicht sie die unterschiedliche Dichte aus, die sich aus der Abtastung des Frequenzraums in Polarkoordinaten ergibt:



# Filterung in der Praxis

- In der Praxis ergeben sich zwei Einschränkungen:
  1. Der Detektorabstand  $\Delta s$  begrenzt gemäß Abtasttheorem die maximale Frequenz in  $p_\theta(s)$  auf  $\omega_{\max} = \frac{1}{2\Delta s}$
  2. Eine Filterung mit  $|\omega|$  verstärkt hohe Raumfrequenzen und somit das Bildrauschen
- Wir ersetzen daher  $|\omega|$  durch einen bandbegrenzten Filter  $H(|\omega|)$ . Dabei müssen wir abwägen zwischen
  - **Bildauflösung:** Optimal bei harter Begrenzung auf  $\omega_{\max}$
  - **Rauschdämpfung:** Schleichende Abschwächung bereits  $< \omega_{\max}$

# Zwei beliebte Filter

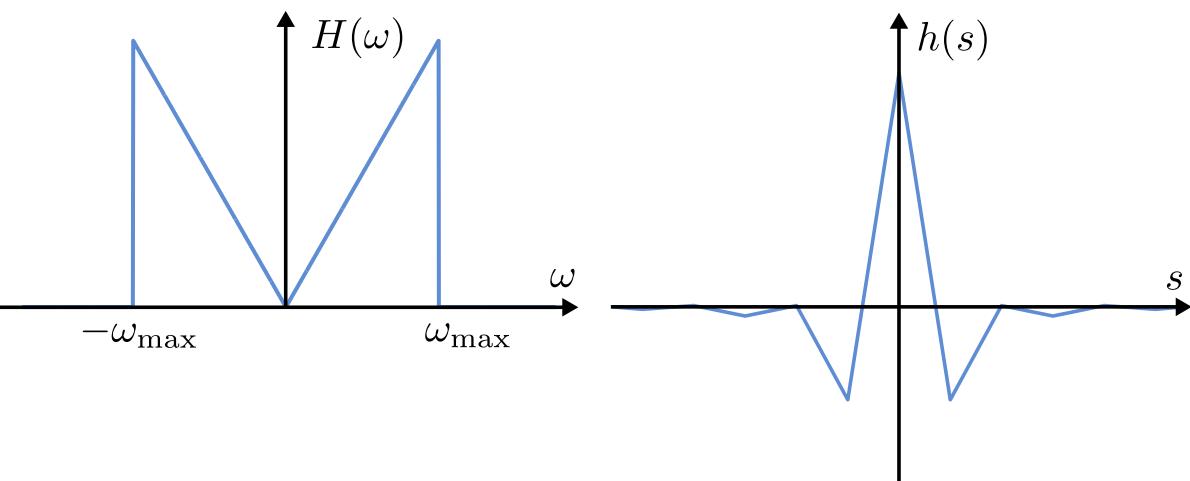
- **Ram-Lak** (Ramachandran/Lakshminarayanan):

$$H(|\omega|) := |\omega| \times \text{rect}(\omega/\omega_{\max})$$

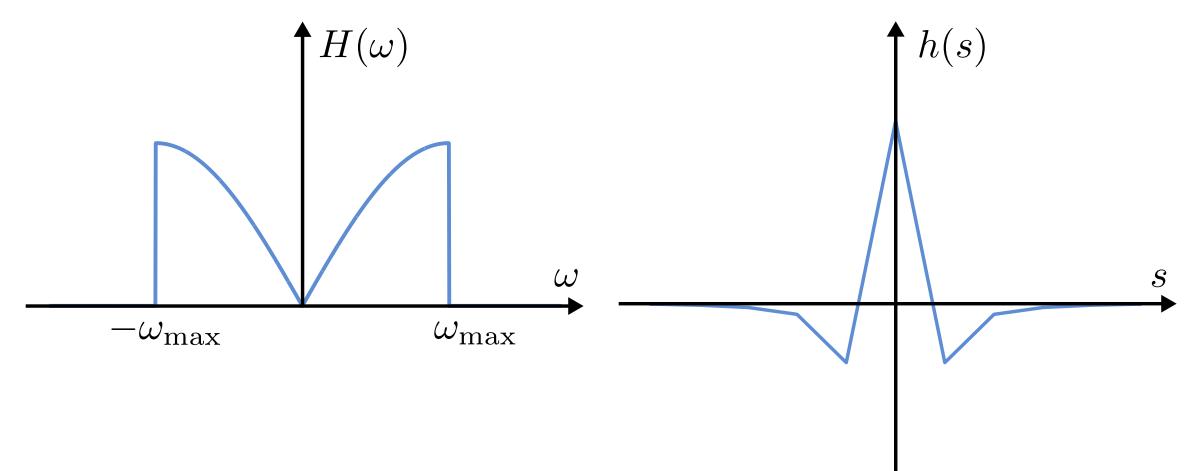
- **Shepp-Logan:**

$$H(|\omega|) := |\omega| \times \text{sinc}(\pi\omega/2\omega_{\max}) \times \text{rect}(\omega/\omega_{\max})$$

**Ram-Lak**



**Shepp-Logan**



# Gefilterte Rückprojektion: Implementierung

- Aufgrund des Faltungstheorems erhalten wir

$$\tilde{p}_\theta(s) = \int_{-\infty}^{\infty} h(s') p_\theta(s - s') ds'$$

- Aufgrund der Diskretisierung von  $s$  im Detektor wird daraus

$$\tilde{p}_{\theta,s} = \sum_{s'} h_{s'} p_{\theta,s-s'} \Delta s$$

- Die Rückprojektion von  $N$  gefilterten Projektionen ergibt

$$f(x, y) = \frac{\pi}{N} \sum_i \tilde{p}_{\theta_i}(s) \text{ mit } s = x \cos \theta_i + y \sin \theta_i$$

- Auswertung von  $\tilde{p}_{\theta_i}(s)$  durch Interpolation von  $\tilde{p}_{\theta,s}$
- Iteration über Pixel  $(x, y)$ , keine Interpolation im Frequenzraum nötig!

# Hounsfield-Einheiten

CT berechnet aus einfallender Röntgenintensität  $I_0$  und ausgehender Intensität  $I$  zunächst Schwächungskoeffizienten  $\mu$ :

$$\ln\left(\frac{I_0}{I}\right) = \int \mu(x) \, dx$$

Diese werden dann in **Hounsfield-Einheiten** (*engl.* Hounsfield Unit, HU) umgerechnet:

$$HU_x := 1000 \times \frac{\mu_x - \mu_{\text{Wasser}}}{\mu_{\text{Wasser}}}$$

- Kalibrierung auf Luft:=-1000 und Wasser:=0
- Ganzzahlige Diskretisierung mit 12 Bit: -1024 bis +3071 HU

# Hounsfield-Einheiten: Beispielhafte Werte

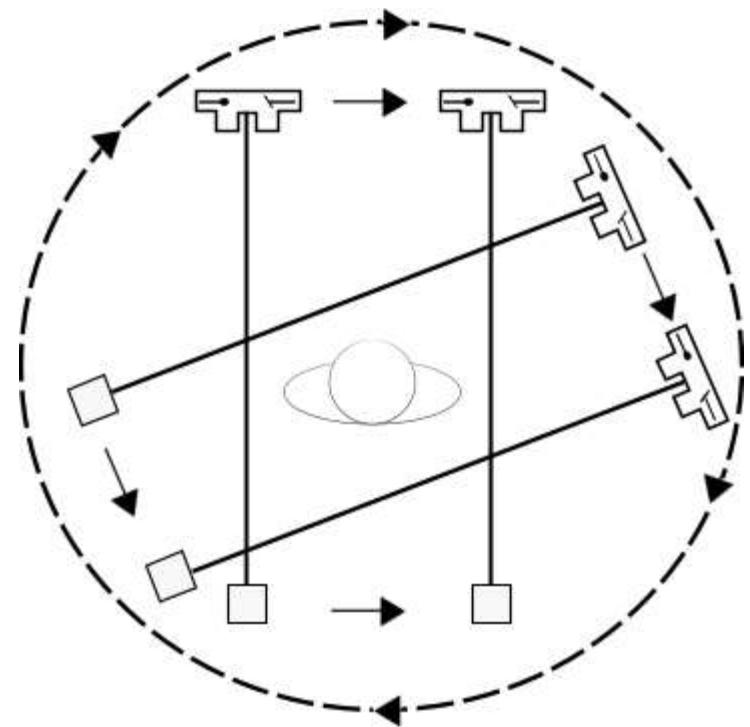
Material	HU (Bereich)
<i>Luft</i>	-1000
Lunge	≈ -600 to -400
Fettgewebe	≈ -100 to -20
<i>Wasser</i>	0
Muskel / Weichgewebe	≈ +20 to +80
Knochen	> +500

*Hinweis:* HU anderer Materialien als Luft und Wasser weisen eine gewisse Abhangigkeit u.a. von der Harfe der Rontgenstrahlung auf

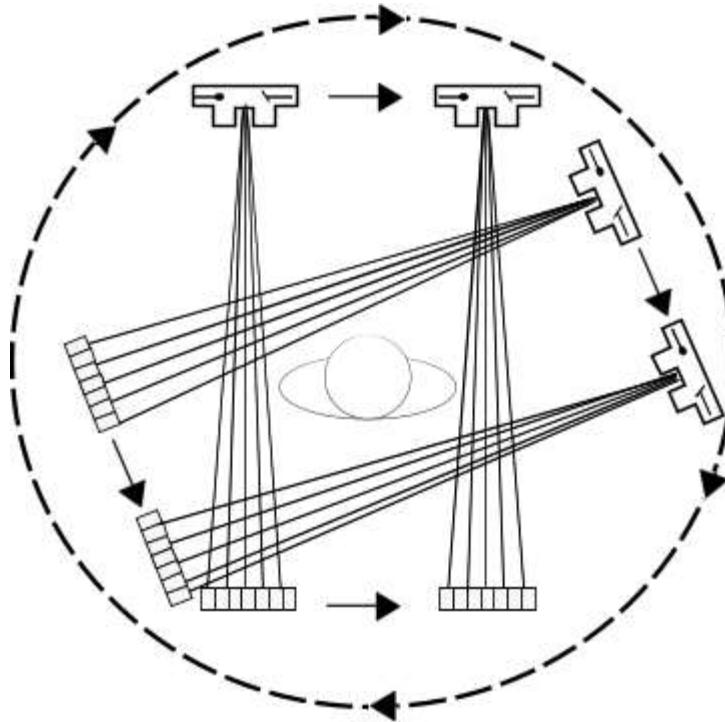
# Aufbau von CT-Scannern: Erste/Zweite Generation

CT-Geräte verschiedener Generationen erfassen  $p_\theta(s)$  wie folgt:

1. Generation: Einzelner Röntgenstrahl misst alle  $\theta$  und  $s$  nacheinander

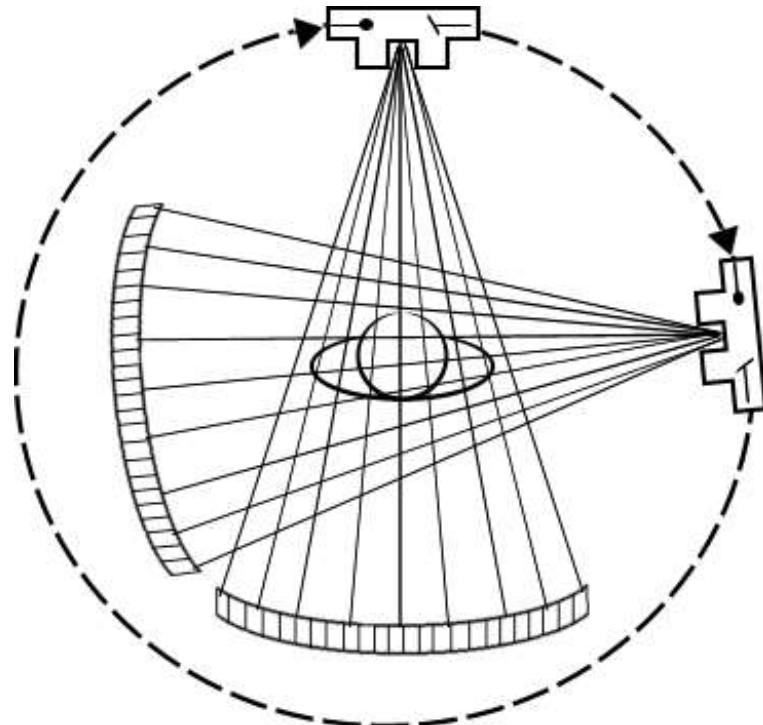


2. Generation: Strahlenfächer und Detektor-Array messen mehrere  $(\theta, s)$  gleichzeitig



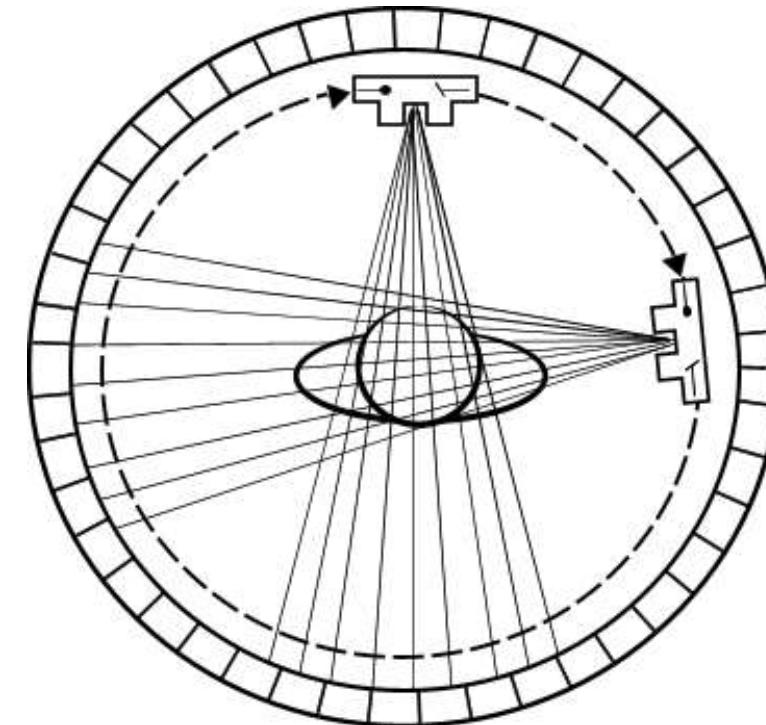
# Aufbau von CT-Scannern: Dritte/Vierte Generation

3. Generation: Strahlenfächer deckt den kompletten Körper ab, keine Translation mehr nötig



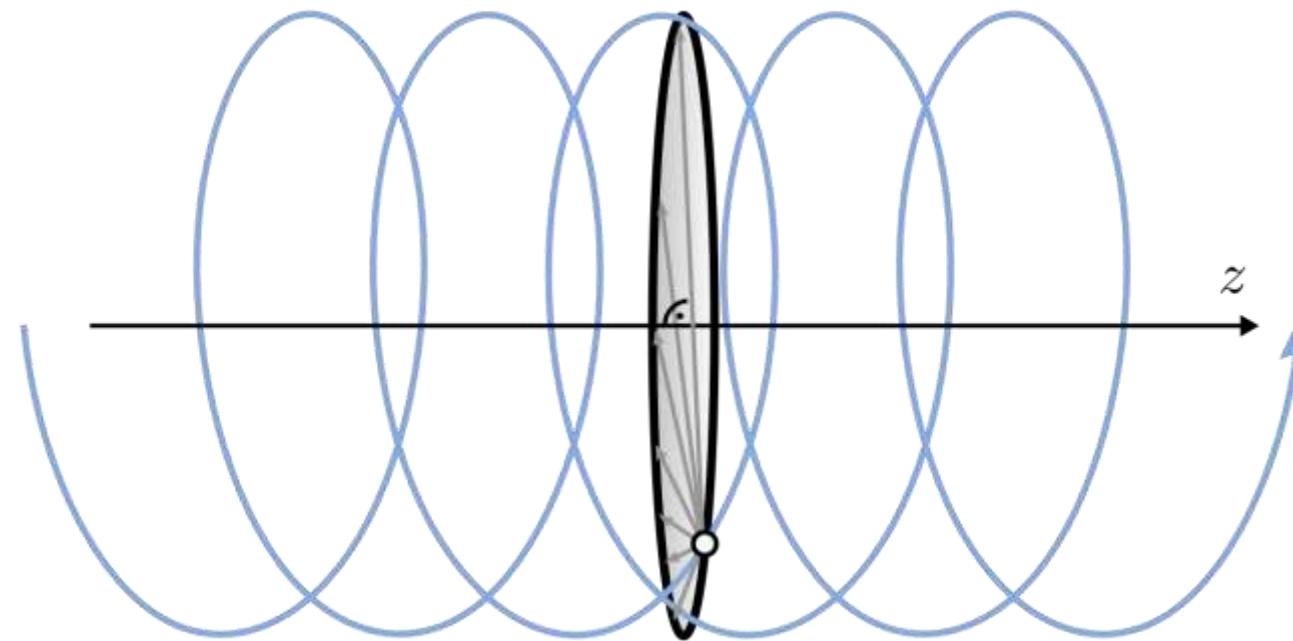
4. Generation: Ein Detektorring umgibt den Körper, nur die Röntgenröhre rotiert

- Idee konnte sich nicht durchsetzen



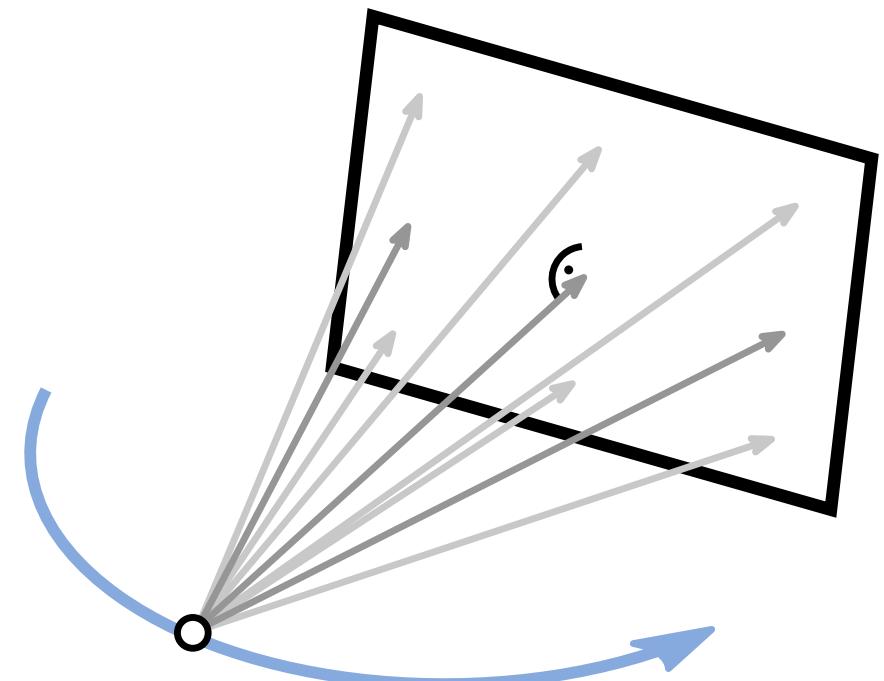
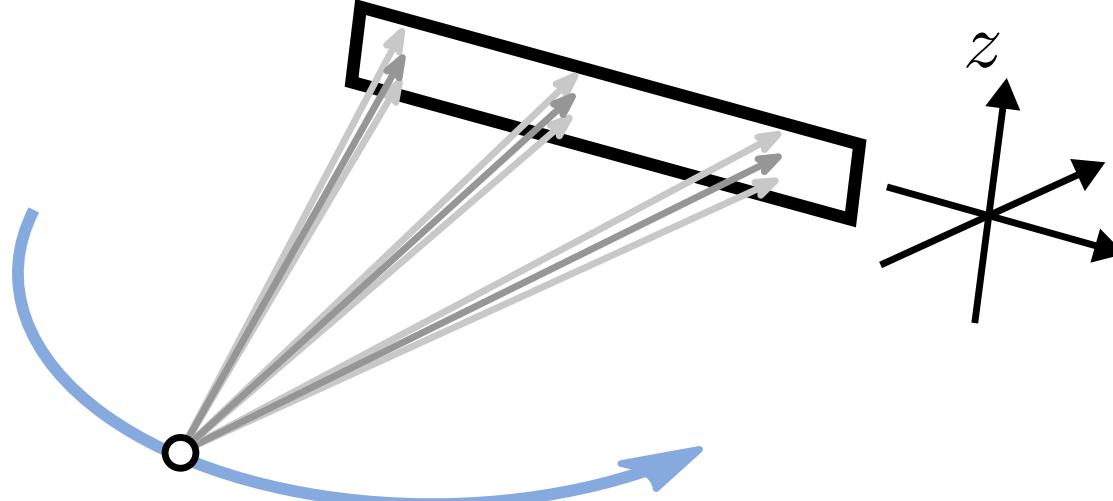
# Spiral-/Helix-CTs

- **Spiral-CTs** (seit 1989) schieben den Patiententisch *während* der Rotation kontinuierlich vor
  - Schichten sind nicht mehr durch die Aufnahme definiert, sondern werden durch Interpolation rekonstruiert



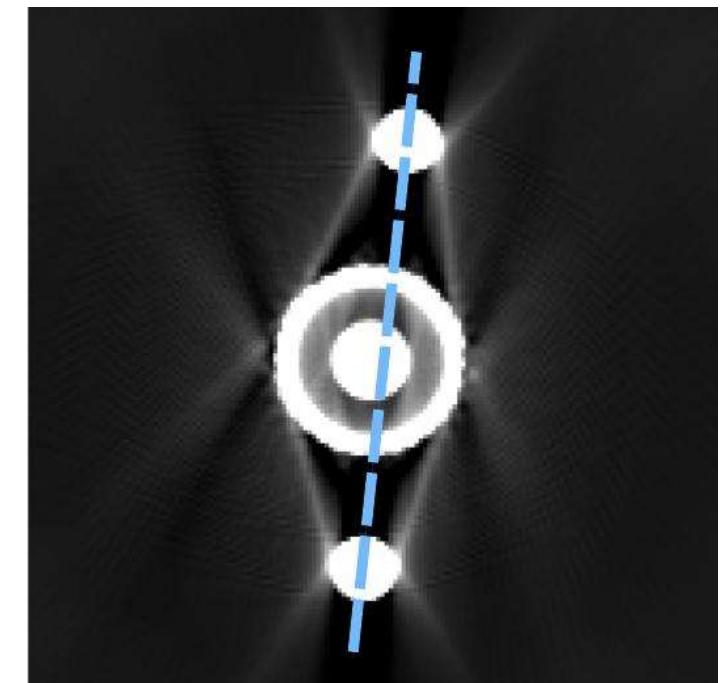
# Mehrschicht-CTs

- **Mehrschicht-CT-Geräte** (seit 1993) nehmen durch ein 2D-Detektor-Array mehrere Schichten gleichzeitig auf
  - Aber: Nicht mehr alle Strahlen liegen in der Rotationsebene!



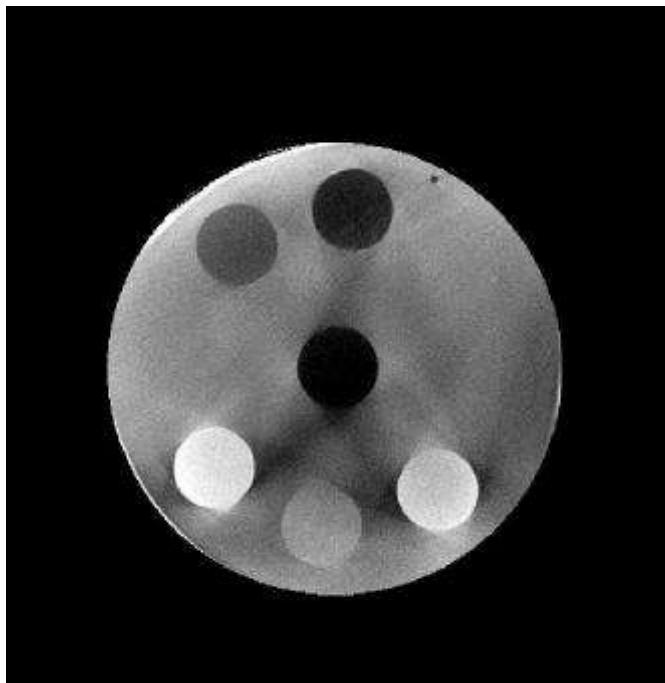
# Bildartefakte im CT: Beispiel Strahllaufhärtung

- Die CT-Rekonstruktion geht davon aus, dass ein Volumenelement  $(x, y)$  in jeder Projektion denselben Schwächungskoeffizienten  $\mu(x, y)$  besitzt
- In der Praxis ist das u.a. durch **Strahllaufhärtung** verletzt:
  - Der Schwächungskoeffizient hängt vom Röntgenspektrum ab
  - Das Spektrum wird zwischen Röntgenröhre und Punkt  $(x, y)$  durch stärkere Absorption “weicher” Anteile “härter”
  - $\mu(x, y)$  hängt vom Material auf diesem Weg ab
- Dies führt zu Streifenartefakten im Bild

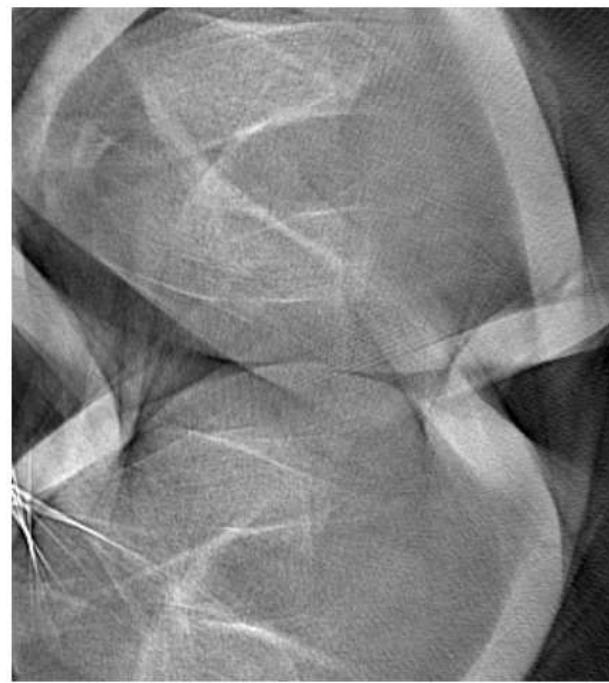


# Weitere Bildartefakte im CT

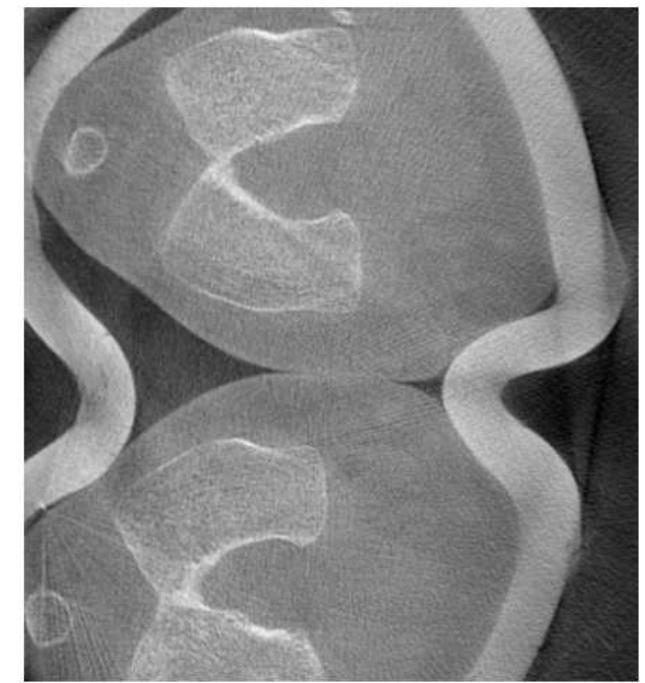
- Auch die Streuung von Röntgenstrahlung sowie Bewegung während der Aufnahme (z.B. durch Herzschlag und Atmung) verletzen die Annahmen der CT-Rekonstruktion
  - z.T. kann man diese Faktoren jedoch algorithmisch berücksichtigen



Streuungsartefakte



Bewegungsartefakt



Korrigierte Rekonstruktion 44

# Zusammenfassung: Computertomographie

- **Computertomographie** (*engl. computed tomography, CT*)
  - basiert auf Röntgenaufnahmen aus verschiedenen Richtungen
  - rekonstruiert aus Radon-Transformation 2D-Schichtbilder, diese werden zu einem 3D-Volumen zusammengesetzt
- **Gefilterte Rückprojektion**
  - ist ein verbreiteter Algorithmus zur CT-Rekonstruktion
  - basiert auf Fourier-Scheiben-Theorem
- Schwächungskoeffizienten werden im CT üblicherweise in **Hounsfield-Einheiten** umgerechnet
- **Artefakte** entstehen u.a. durch Strahlaufhärtung, Streuung, Bewegung

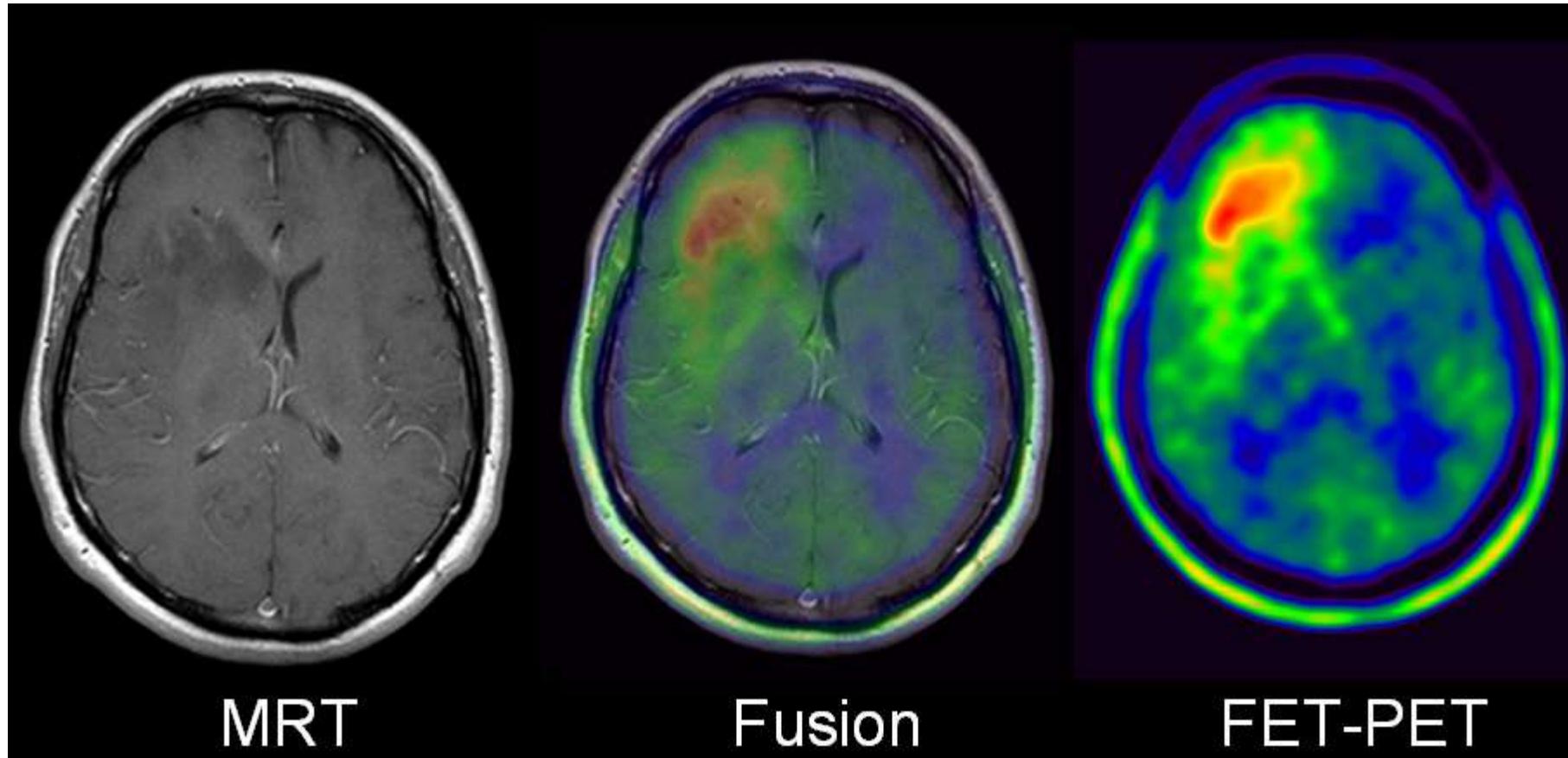
### **3.3 Nuklearmedizinische Bildgebung**

# Zielsetzung und Grundidee

- Röntgen und CT stellen **Strukturen** im Körperinneren dar
- **Funktionelle** Bildgebung zeigt dagegen, wo bestimmte biologische Prozesse stattfinden
  - Das **Tracer-Prinzip** nutzt Stoffe, die am Stoffwechsel teilnehmen und gleichzeitig ein zur Bildgebung nutzbares Signal liefern
  - Die **Nuklearmedizin** setzt radioaktive Substanzen zu diagnostischen und therapeutischen Zwecken ein, insbesondere auch als Tracer, der gespritzt, geschluckt oder inhaliert wird
  - **Ziel:** Darstellung der raumzeitlichen Verteilung des Tracers im Körper

# Beispiel: FET-PET zur Darstellung von Hirntumoren

- Hirntumore nehmen vermehrt Tyrosin auf. F-18-Ethyltyrosin (FET) dient daher der Tumordiagnostik.



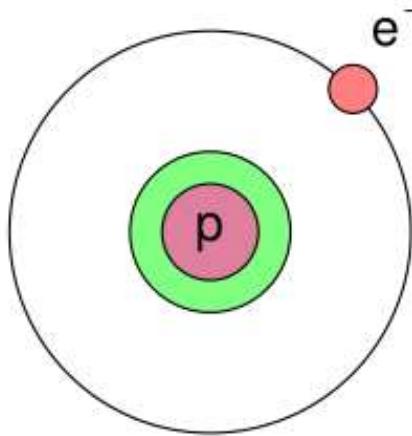
# Kurze Geschichte Nuklearmedizinischer Bildgebung

- **György Hevesy** (1885-1966) gilt als Pionier des Tracer-Prinzips
  - 1923: Untersuchung der Aufnahme von Blei durch Ackerbohnen mittels radioaktivem  $^{212}\text{Pb}$  in nicht toxischer Konzentration
  - 1935: Untersuchung des Phosphat-Metabolismus von Ratten durch mit  $^{32}\text{P}$  versetztem Futter
    - Führt insbesondere zum Nachweis eines kontinuierlichen Austauschs von Phosphor im Knochen
  - 1943: Nobelpreis für Chemie
- **Hal Anger** erfindet 1957 die Gamma-Kamera
- Erste SPECT-Scanner in den 1960er Jahren
- Erste (moderne) PET-Scanner in den 1970er Jahren

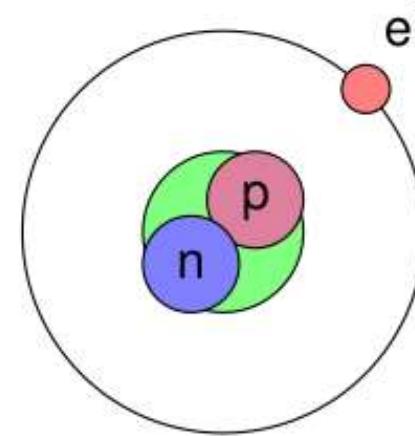


# Physikalische Grundlage: Isotope

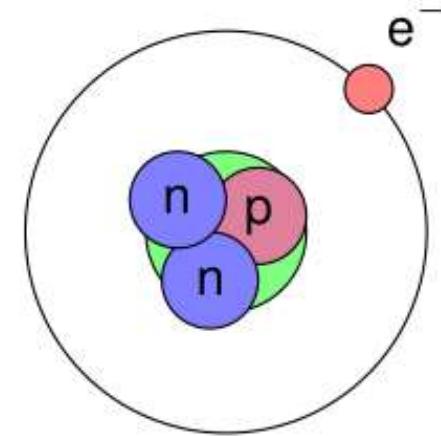
- **Chemische Elemente** unterscheiden sich durch die Zahl der Protonen im Kern (Ordnungszahl Z)
- **Nuklide** sind durch Ordnungszahl, Massenzahl  $A=Z+N$  ( $N$ =Neutronenzahl) und Energiezustand charakterisiert
- **Isotope** sind Nuklide derselben Ordnungszahl (selbes Element)



$^1_1H$



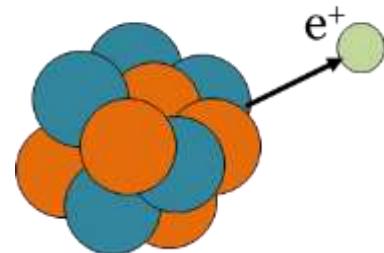
$^2_1H$



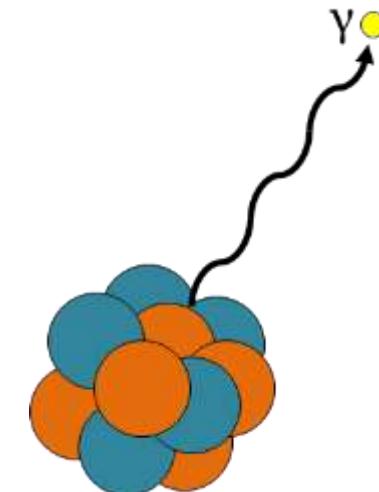
$^3_1H$

# Physikalische Grundlage: Radioaktiver Zerfall

- Instabile Nuklide wandeln sich spontan in andere um
  - Alpha-Zerfall: Aussendung eines  ${}^4\text{He}$ -Kerns
  - Beta-Zerfall: Aussendung eines Elektrons ( $\beta^-$ ) oder Positrons ( $\beta^+$ )
  - Elektroneneinfang: Proton im Kern wird zu Neutron
  - Isomerer Übergang: Aussendung von Gamma-Strahlung von angeregten („metastabilen“) Nukliden



$\beta^+$ -Zerfall



Isomerer Übergang

# Halbwertszeiten

- Radioaktiver Zerfall erfolgt exponentiell:

$$N(t) = N_0 e^{-\lambda t}$$

- $N(t)$ : Zahl der Nuklide zur Zeit  $t$
  - $N_0$ : Zahl der Nuklide zu Beginn ( $t = 0$ )
  - $\lambda$ : Zerfallskonstante
- 
- Statt der Zerfallskonstante wird häufig eine **Halbwertszeit** angegeben, nach der noch die Hälfte der Nuklide verbleibt:

$$T_{1/2} = \frac{\ln 2}{\lambda}$$

# Aktivität

- Die **Aktivität** einer Probe ist die Zahl der Zerfälle pro Zeit

$$A(t) = -\frac{dN}{dt} = \lambda N_0 e^{-\lambda t} = A_0 e^{-\lambda t}$$

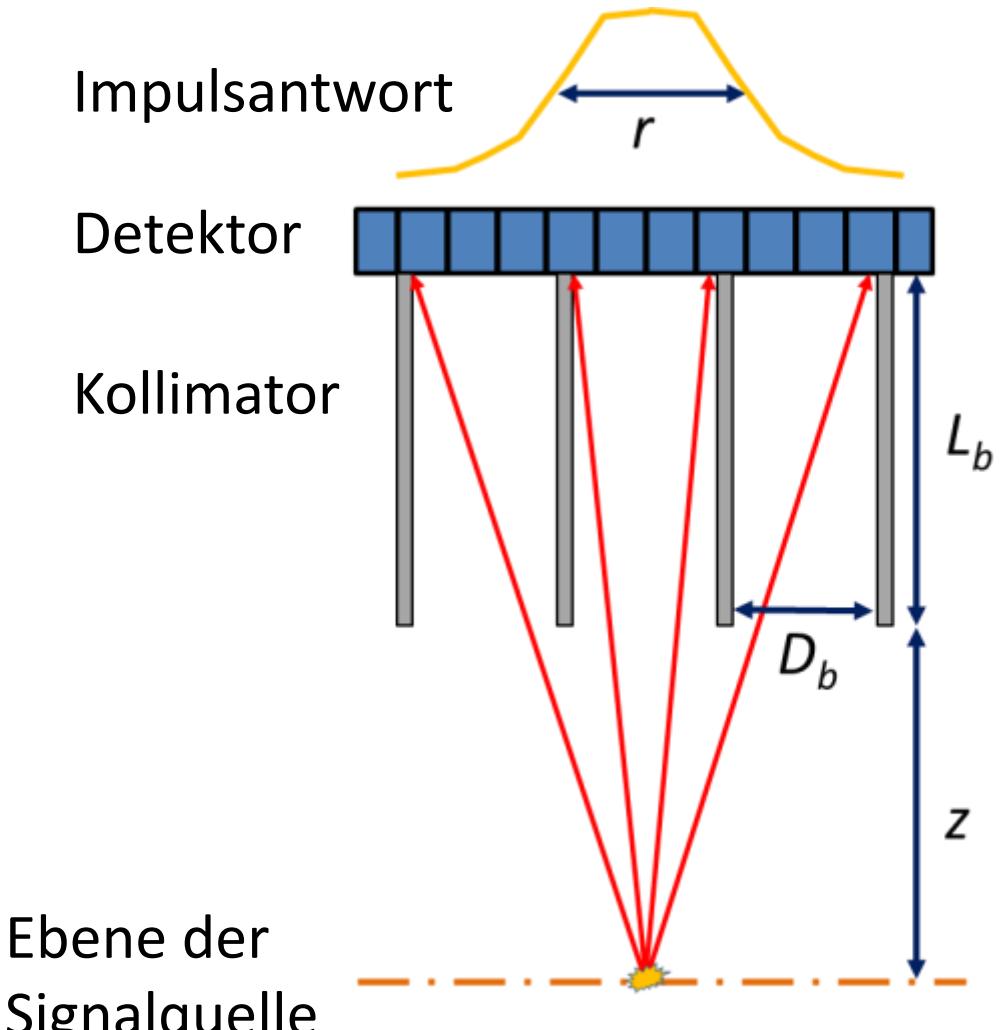
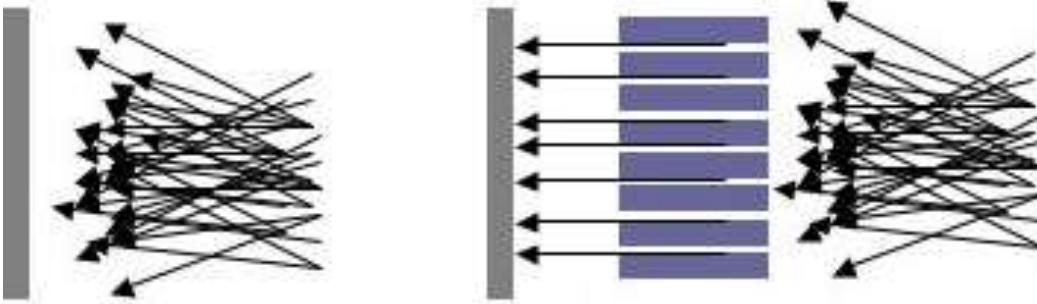
- *Einheit:* Becquerel (Bq) = Zerfälle pro Sekunde
- Kurze Halbwertszeit bedeutet hohe anfängliche Aktivität, die jedoch rasch wieder abnimmt. Bei Tracern muss die Halbwertszeit
  - mindestens einige Sekunden betragen, um sie anwenden zu können
  - höchstens einige Stunden betragen, um hinreichende Aktivität zu erzeugen
- Durch Tracer eingebrachte Aktivitäten liegen üblicherweise im Bereich 100 MBq bis 1000 MBq

# Nuklearmedizinische Messtechnik

- Die **Detektion von Gammastrahlung** erfolgt üblicherweise durch
  - **Szintillations-Kristalle**, die Gamma-Quanten in sichtbare Photonen umwandeln
  - **Photomultiplier**, die die zunächst relativ wenigen Quanten zu einem messbaren Signal verstärken
- Ein **Impulshöhenanalysator** zählt nur hinreichend starke / energiereiche Ereignisse
  - Sortiert gestreute Quanten aus

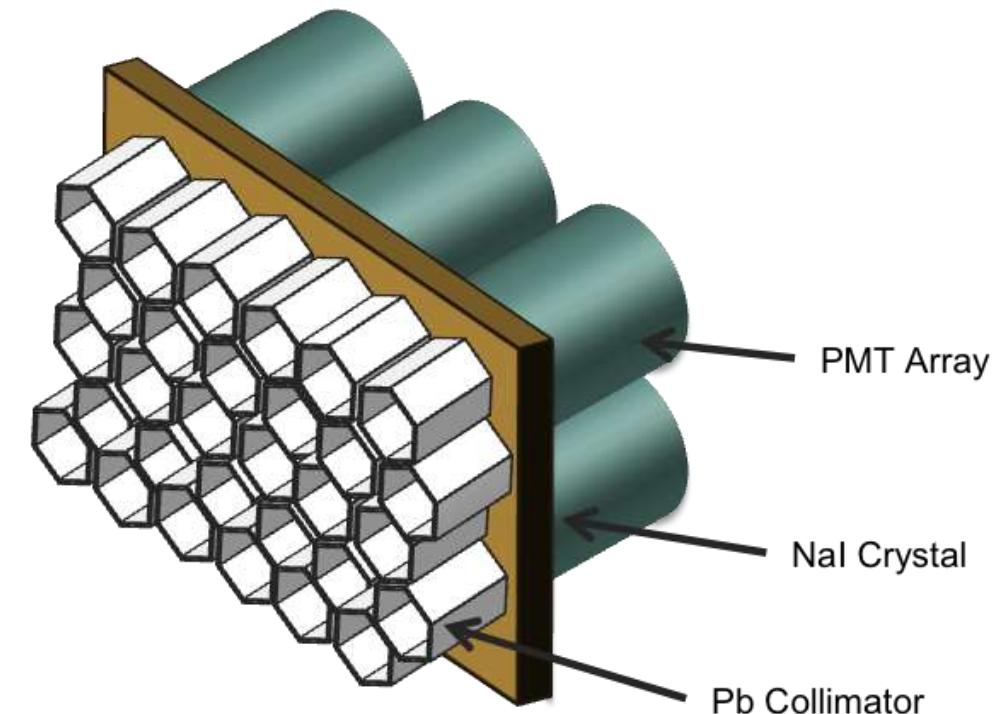
# Kollimatoren

- **Kollimatoren** ermöglichen ortsaufgelöste Messungen, indem sie nur Quanten an den Detektor lassen, die aus einem zylindrischen Bereich stammen
  - Verhältnis Durchmesser/Länge bestimmt Kompromiss zwischen Ortsauflösung und Effizienz (Anteil nachweisbarer Quanten)



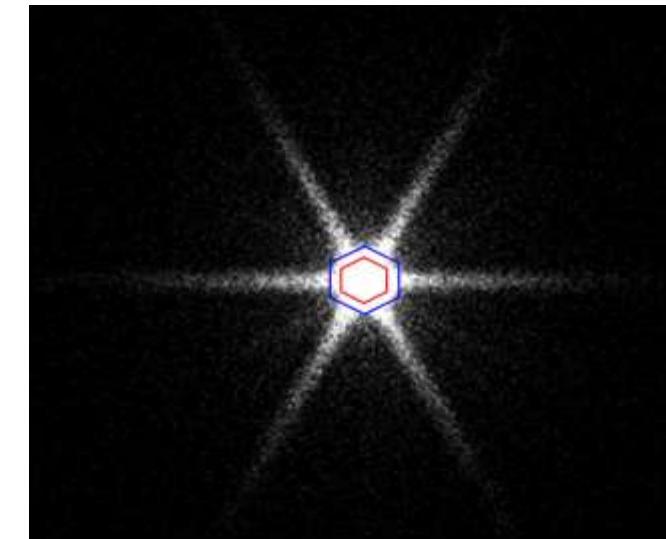
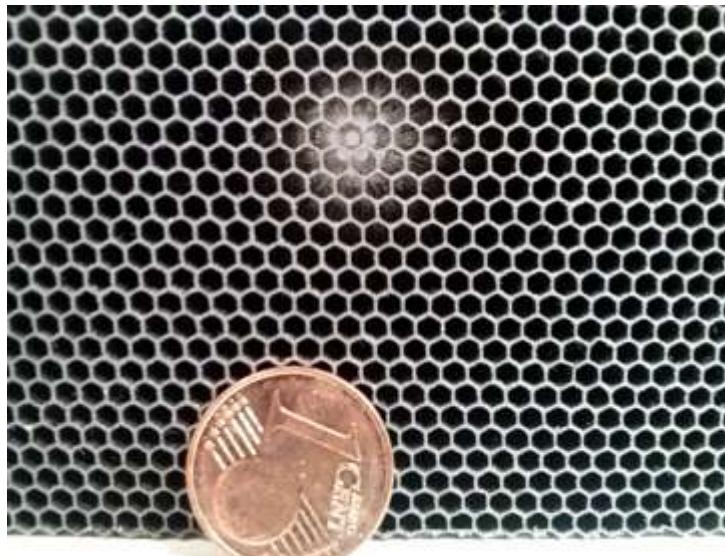
# Gamma-Kameras

- **Gamma-Kameras** erfassen gleichzeitig die Aktivitätsverteilung in einem größeren Bereich. Sie bestehen im Wesentlichen aus
  - Kollimator-Matrix
  - Szintillator-Kristall
  - Anordnung von 37-100 Photomultipliern
    - Licht eines jeden Gamma-Quants verteilt sich auf mehrere Photomultiplier
    - Schwerpunkt der Aktivierungen ergibt den genauen Ort eines Ereignis



# Punktspreizfunktion von Gamma-Kameras

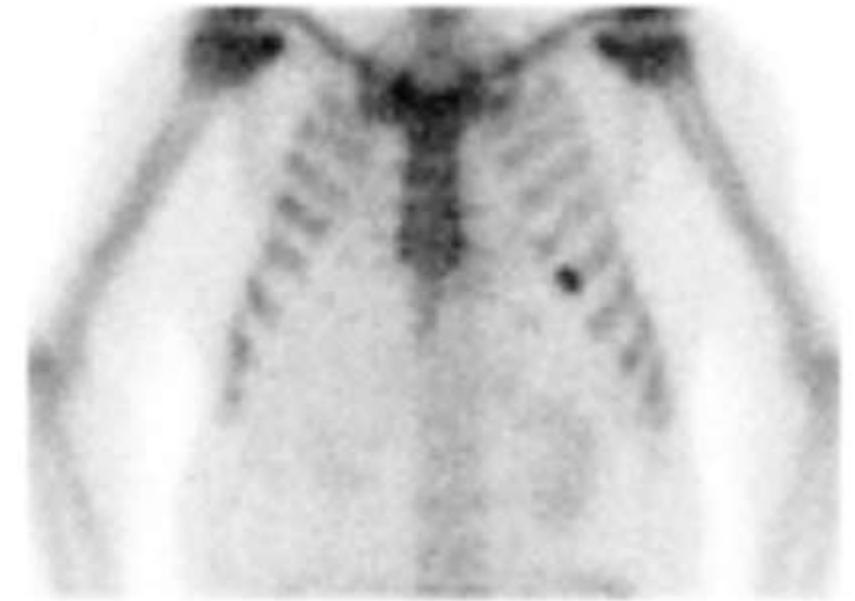
- Die **Punktspreizfunktion** (PSF, *engl.* point spread function) beschreibt das Bild einer punktförmigen Quelle
  - Ihre Breite begrenzt die maximal erreichbare Bildauflösung
- Die PSF von Gamma-Kameras wird wesentlich durch die Geometrie der Kollimatoren bestimmt:



Gemessene PSF einer  $^{99m}\text{Tc}$ -Punktquelle im Abstand von 10cm;  
Darstellung im Zentrum saturiert

# Planare Szintigraphie

- Bei der **planaren Szintigraphie** wird die Aktivität eines Tracers durch eine Gamma-Kamera aufgenommen
  - Ähnlichkeit zu Projektionsröntgen
  - Strahlenquelle befindet sich im Patienten
- Anwendungen sind u.a.
  - Überprüfung der Funktion von Herz, Lunge, Schilddrüse, Niere
  - Diagnose von Schilddrüsen- und Knochentumoren



# SPECT

- Analog zu CT rekonstruiert **Single Photon Emission Computed Tomography** (SPECT) Schnittbilder aus Linienintegralen:

$$\text{CT} \quad \ln\left(\frac{I_0}{I}\right) = \int \mu(x, y) dl \quad \text{vs.} \quad S = \int A(x, y) dl \quad \text{SPECT}$$

- SPECT misst mittels Gamma-Kamera mehrere Schnittebenen gleichzeitig, benötigt pro Position aber deutlich länger
- Statt gefilterter Rückprojektion kommen häufig algebraische Rekonstruktionsverfahren zum Einsatz

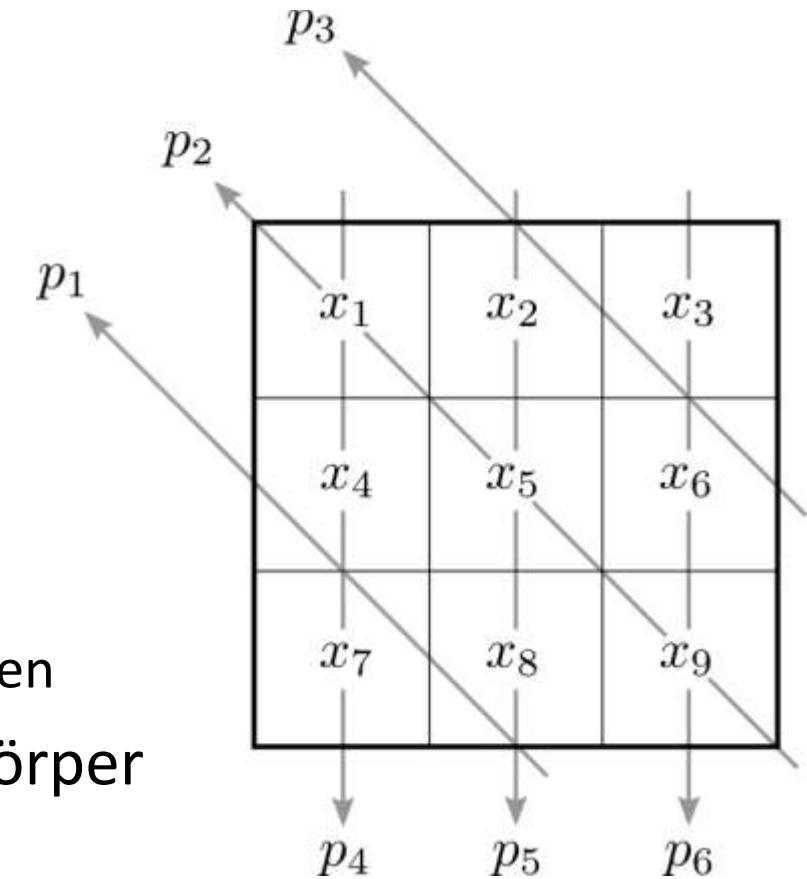


# Idee der Algebraischen Rekonstruktion

- Die **algebraische Rekonstruktion** basiert darauf, die Projektionen  $p$  als Linearkombination der Pixelwerte  $x_i$  im Schichtbild darzustellen:

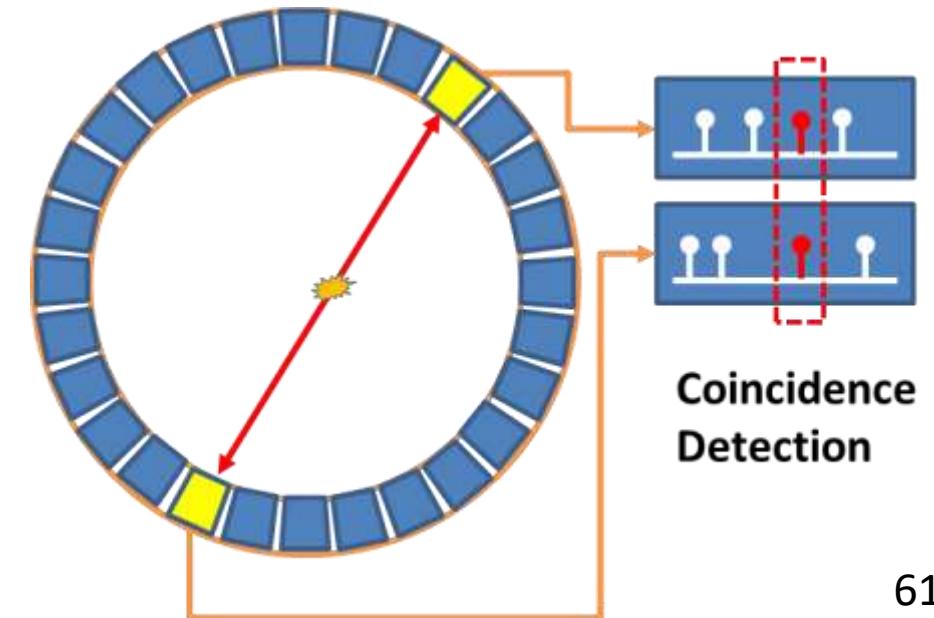
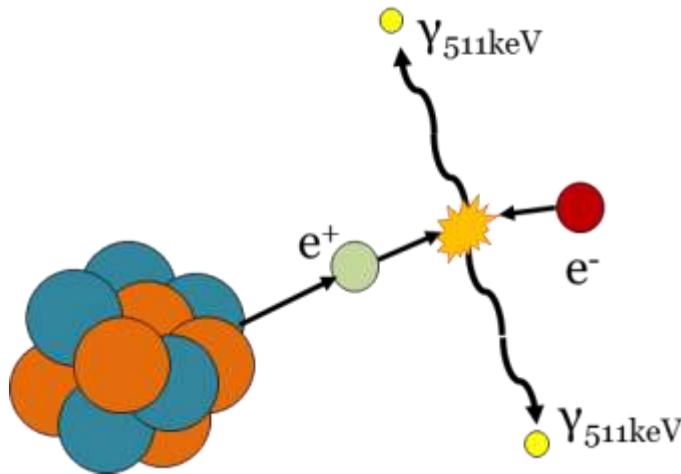
$$\mathbf{Ax} = \mathbf{p}$$

- Führt zu einem großen, dünn besetzten, schlecht konditionierten und i.d.R. überbestimmten Gleichungssystem
- Lösung mit iterativen numerischen Verfahren
  - Berücksichtigen Stochastizität radioaktiven Zerfalls
- Für CT unpraktikabel
  - SPECT hat geringere Auflösung und weniger Projektionen
- Ermöglicht Berücksichtigung der Absorption im Körper und von Abbildungsfehlern des Detektors (PSF)



# PET

- Die **Positronen-Emissions-Tomographie (PET)**
  - nutzt Positronen-Strahler statt Gamma-Strahlern
  - Annihilation von Positron und Elektron erzeugt zwei Gamma-Quanten im Winkel von ca. 180°
  - Deren gleichzeitiges Auftreffen auf einen Detektorring zeigt die Linie an, entlang der sie entstanden sind
    - max. 10-20 ns Unterschied



# PET: Vor- und Nachteile

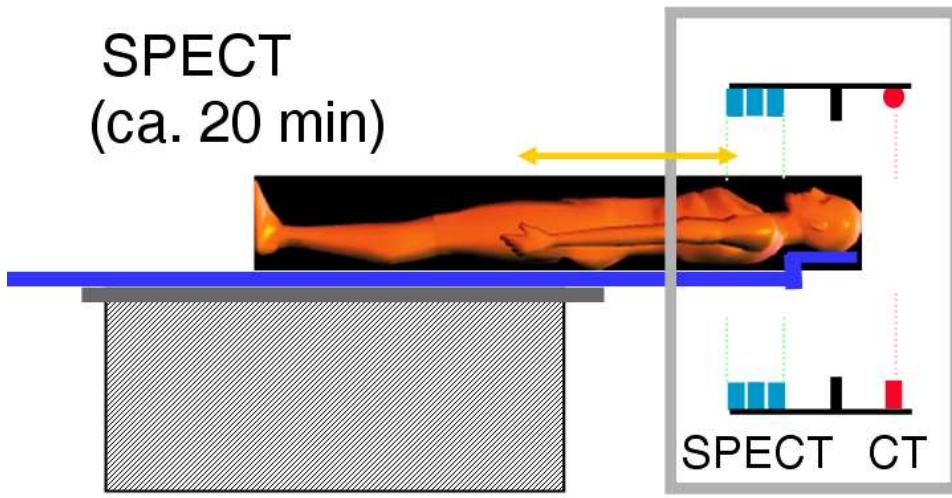
- **Vorteile von PET:**
  - Kollimatoren sind überflüssig, deutlich höhere Ausbeute
  - Gleichzeitige Messung aller Richtungen
  - Positronen-Strahler sind relativ kurzlebig, nach der Untersuchung verbleibt kaum aktives Radionuklid
- **Nachteile von PET:**
  - Kurze Halbwertszeiten erfordern Herstellung kurz vor Verabreichung
  - Wegstrecke bis zur Annihilation (einige mm) reduziert Ortsauflösung
  - Szintillatoren und Detektor-Elektronik sind kostspielig

# Hybride Bildgebung

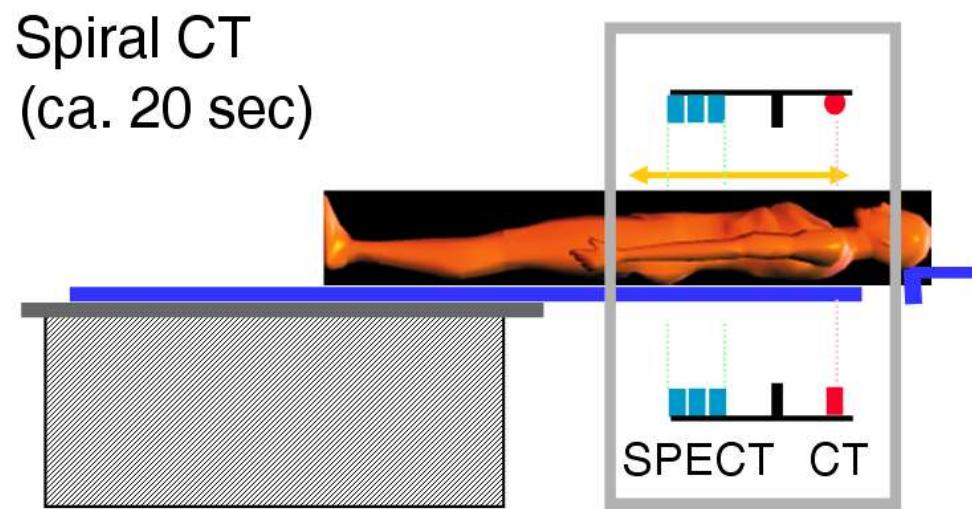
- Kombinierte **SPECT/CT-** und **PET/CT-Scanner** ermöglichen Überlagerung funktioneller und struktureller Bilder
  - CT-Bilder können auch zur Absorptionskorrektur genutzt werden
    - *Aber:* Schwächungskoeffizienten müssen aufgrund unterschiedlicher Quantenenergien geschätzt werden
- Inzwischen gibt es auch **PET/MR-Geräte**
  - Besserer Kontrast im Gehirn (siehe 3.4)
  - Photomultiplier werden durch Photodioden ersetzt
  - Absorptionskorrektur noch indirekter



# Illustration: SPECT-CT-Aufnahme der Schilddrüse

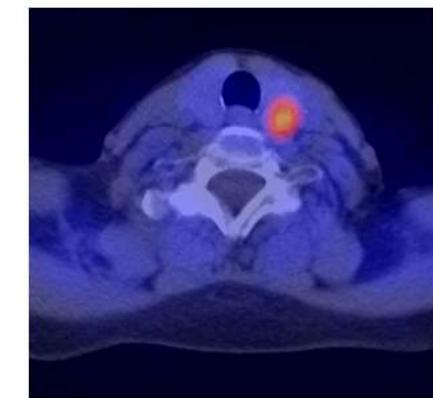
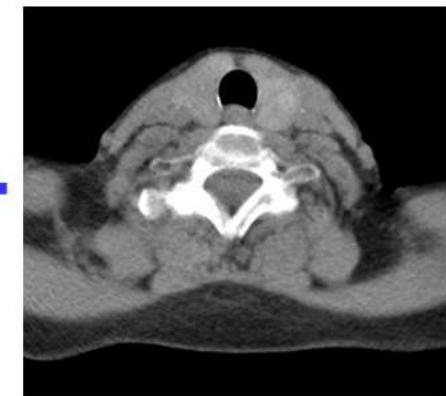


SPECT/CT



SPECT

CT



# Zusammenfassung: Nuklearmedizinische Bildgebung

- Nuklearmedizinische Bildgebung rekonstruiert die Aktivitätsverteilung **radioaktiver Tracer** im Körper
  - Ermöglicht **funktionelle Bildgebung**
- Gamma-Strahler und **Gamma-Kameras** ermöglichen
  - planare Szintigraphie (Projektionsbilder)
  - **SPECT** (Schichtbilder)
- **PET** nutzt Positronen-Strahler und Koinzidenz-Erkennung in Detektorringen
- Iterative **algebraische Verfahren** zur genaueren Bildrekonstruktion
- **Hybride Bildgebung** zur Überlagerung von Struktur und Funktion

## **3.4 Magnet-Resonanz-Tomographie**

# Grundprinzipien der Magnetresonanz-Tomographie

- Basiert auf der **Kernspinresonanz**
  - Physikalischer Effekt, bei dem bestimmte Atomkerne elektromagnetische Wellen bestimmter Frequenzen absorbieren
- Erfordert ein **starkes permanentes Magnetfeld** und bestimmte Abfolgen von **Impulsen hochfrequenter elektromagnetischer Wellen**
- Die genutzte Strahlung ist *nicht* ionisierend
- Bildgebung auch bei gesunden Probanden ethisch vertretbar



# Geschichte der MRT

- **Felix Bloch** und **Edward Purcell** beschreiben 1945/46 die magnetische Resonanz
  - Erhalten 1952 den Nobelpreis für Physik
- **Erik Odeblad** beschreibt 1955 unterschiedliche Relaxationszeiten verschiedener Gewebetypen
- **Raymond Damadian** patentiert 1974 einen MR-basierten Apparat zur (bildlosen) Krebserkennung
- **Paul Lauterbur** publiziert 1973 erste MRT-Bilder, **Sir Peter Mansfield** entwickelt mathematischen Formalismus und effiziente Messverfahren
  - Erhalten 2003 den Nobelpreis für Physiologie oder Medizin
- **John Mallard** entwickelt ersten Ganzkörperscanner, erste klinische Nutzung 1980



Felix Bloch  
(1905-1983)



Edward Purcell



Paul Lauterbur and Sir Peter Mansfield

# Motivation: MRT vs. CT

- CT stellt röntgendichte Materialien wie z.B. **Knochen** besonders kontrastreich dar
- MRT ermöglicht verschiedene Kontraste, insbesondere für **Weichgewebe** und **Organe**
- CT ist **schneller** (Sekunden statt Minuten), **kostengünstiger** und **breiter verfügbar**
- CT ermöglicht **höhere Auflösung**
  - ca. 0.4mm; MRT: 0.5-3mm
- CT ist **quantitativ** (Hounsfield-Einheiten), Intensität in MRT-Bildern ist nur relativ zu interpretieren



CT-Scan



MRT-Scan

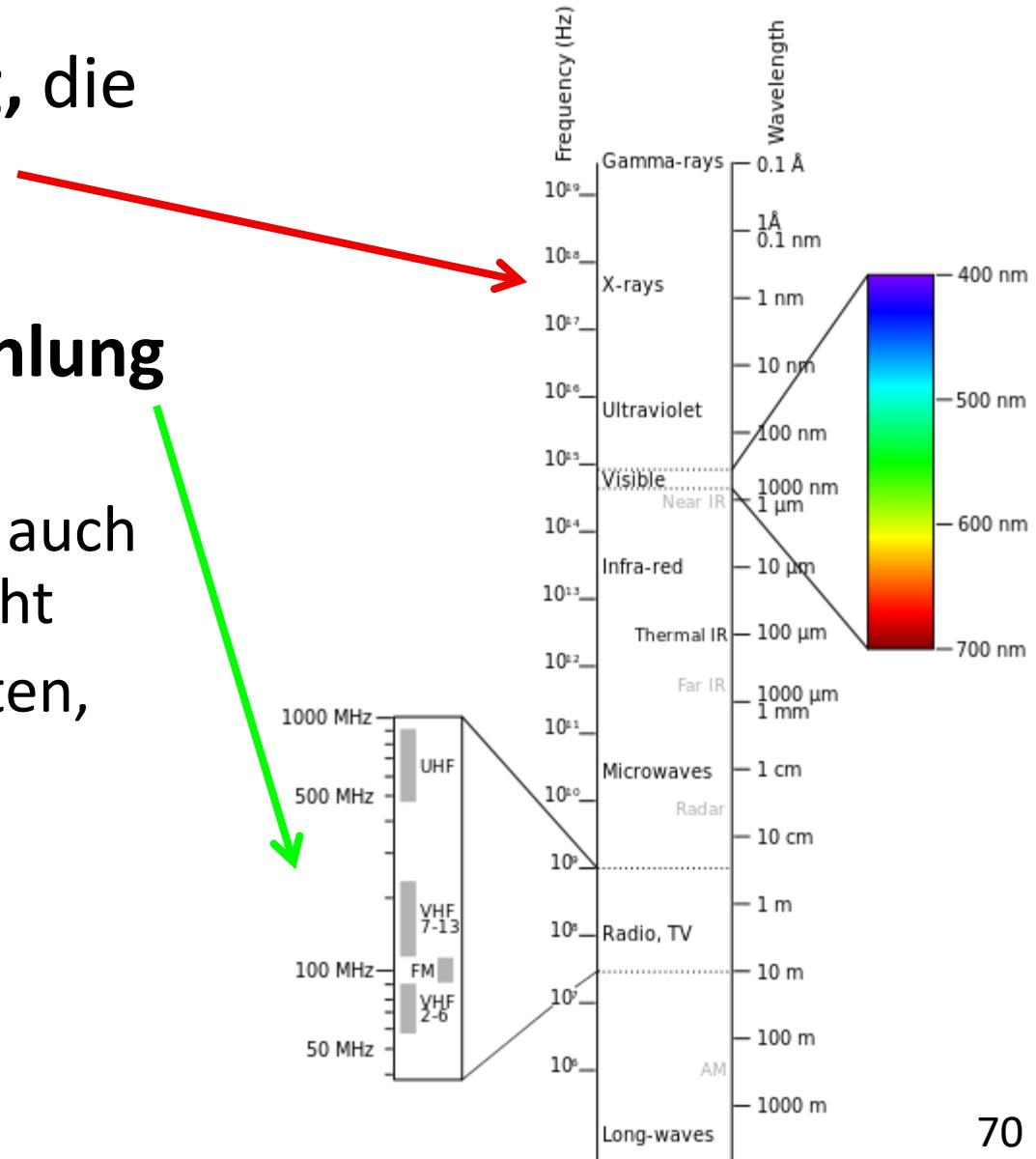
# Sicherheit von MRT und CT

- CT nutzt **ionisieren Röntgenstrahlung**, die biologischem Gewebe schaden kann
- Die starken **Magnetfelder** und die **langwellige elektromagnetische Strahlung** im MRT gelten als unbedenklich
  - MRT-Aufnahmen werden ohne weiteres auch aus wissenschaftlichem Interesse gemacht
  - *Aber:* Vorsicht bei bestimmten Implantaten, Tätowierungen



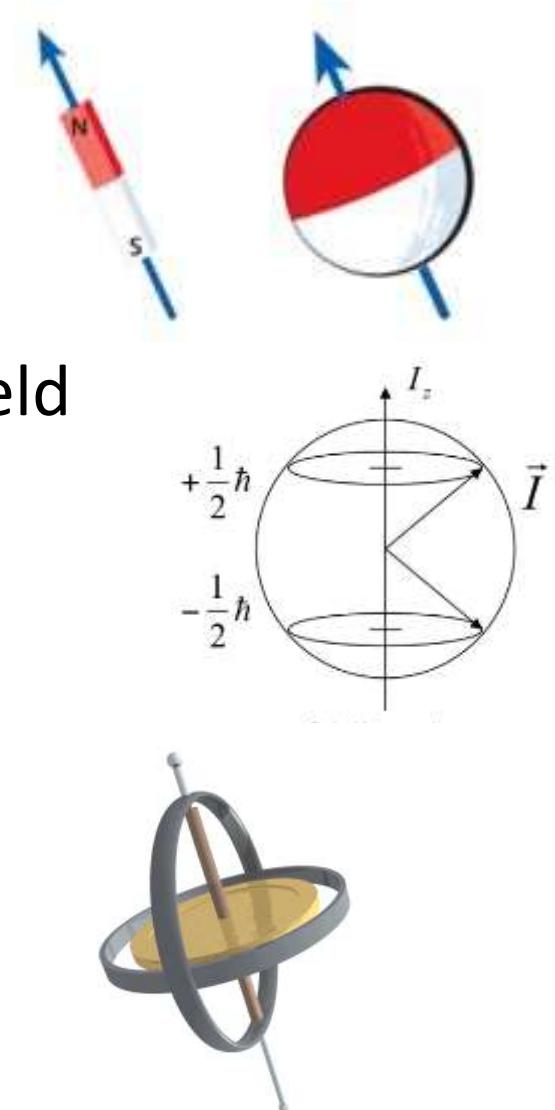
...und Bürostühlen

Bildquellen: Victor Blacus / [www.impactednurse.com](http://www.impactednurse.com)



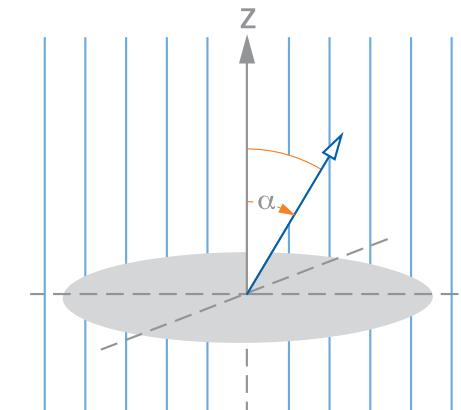
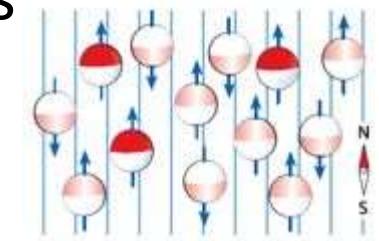
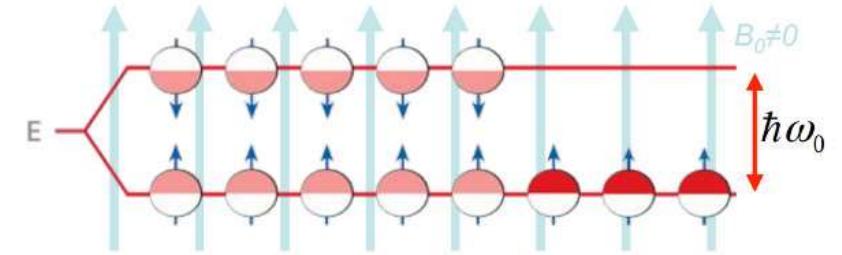
# Magnetisches Dipolmoment

- Der menschliche Körper besteht zu  $\approx 70\%$  aus H<sub>2</sub>O
- Die Atomkerne von Wasserstoff besitzen ein **magnetisches Dipolmoment  $\mu$** 
  - **Polarisierung:** In einem statischen externen Magnetfeld  $B_0$  richtet sich  $\mu$  in einem festen Winkel zu  $B_0$  aus.
  - Die Richtung von  $B_0$  bezeichnen wir als z-Achse, die Transversalebene wird durch x und y aufgespannt
  - Aufgrund des Winkels präzidiert  $\mu$  um  $B_0$ 
    - **Larmor-Frequenz**  $\omega_0 = \gamma B_0$
    - $\gamma$  = gyromagnetisches Verhältnis
    - Für Wasserstoff-Kerne ist  $\gamma = 267.513 \cdot 10^6$  rad/(s·T)

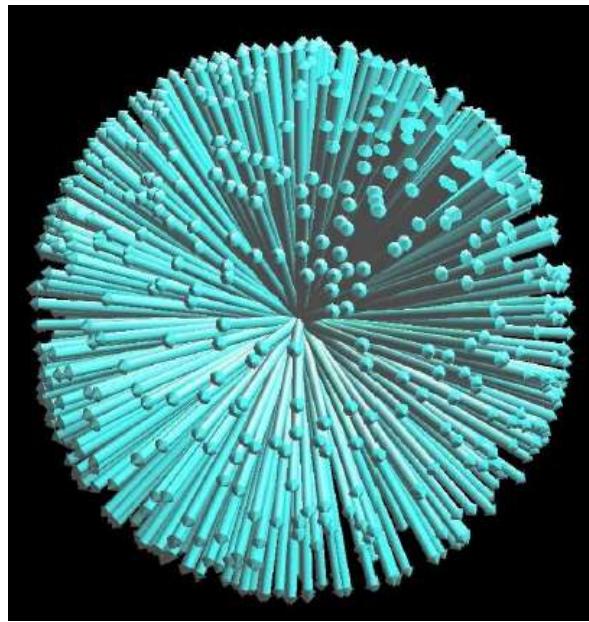


# Kernspinresonanz

- Die z-Komponente von  $\mu$  richtet sich parallel oder antiparallel zu  $\mathbf{B}_0$  aus
  - Entspricht niedrigem / hohem Energieniveau
  - Im thermischen Gleichgewicht herrscht ein leichter Überschuss von Kernen im niedrigen Energieniveau
  - Deren magnetische Dipolmomente summieren sich zu einer **makroskopischen Magnetisierung**  $\mathbf{M}_0 \sim \mathbf{B}_0$
- **Anregung: Kernspinresonanz**
  - Elektromagnetischer Puls mit der Resonanzfrequenz  $\omega_0$  hebt Kerne *kohärent* in das höhere Energieniveau
  - Magnetisierung  $\mathbf{M}_0$  steht nun im Winkel  $\alpha$  zu  $\mathbf{B}_0$ 
    - $\alpha$  ist proportional zu Stärke und Dauer des Hochfrequenz (HF)-Pulses
    - Nun präzidiert auch  $\mathbf{M}_0$  mit der Larmor-Frequenz

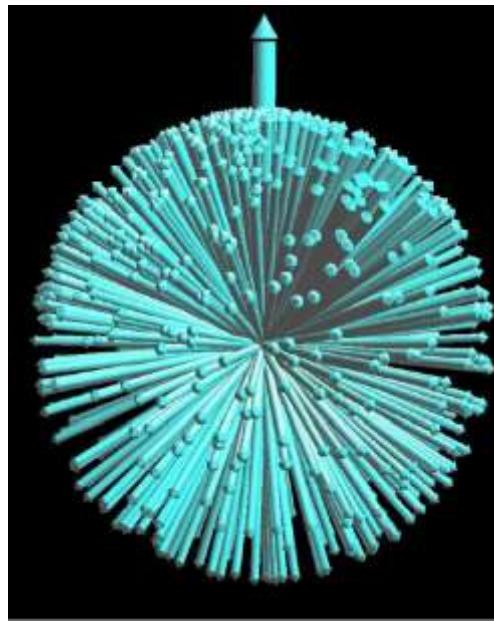


# Graphische Zusammenfassung



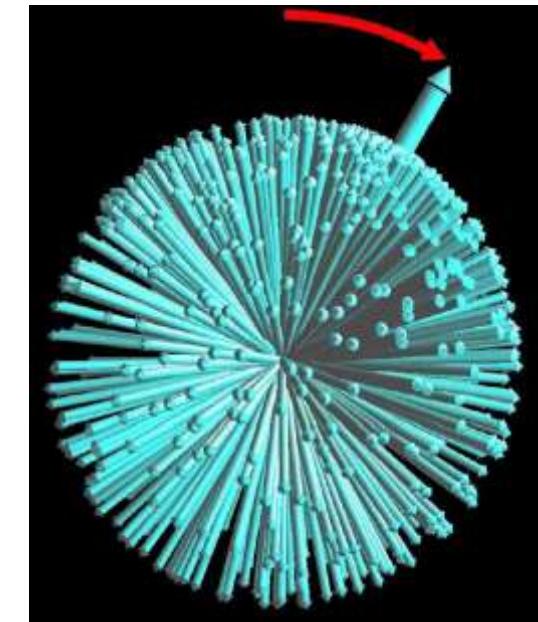
**Ohne externes  
Magnetfeld:**

- Ausrichtung der Spins völlig zufällig
- Keine makroskopische Magnetisierung



**Polarisierung mit starkem  
Magnetfeld ( $B_0$ ):**

- Verteilung bevorzugt Ausrichtung parallel zum Magnetfeld
- Phasen bleiben zufällig

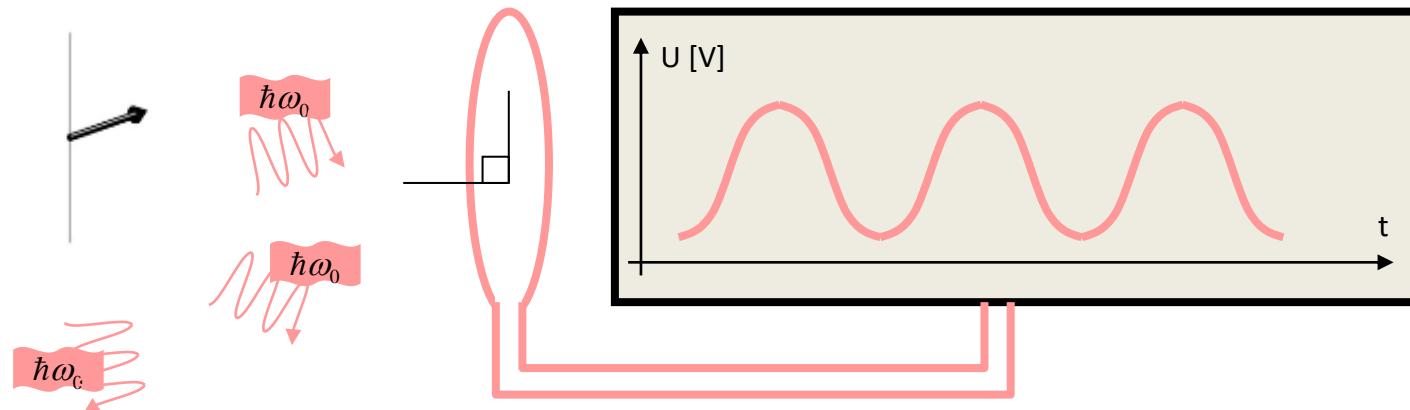


**Anregung mit  
resonantem HF-Puls ( $B_1$ ):**

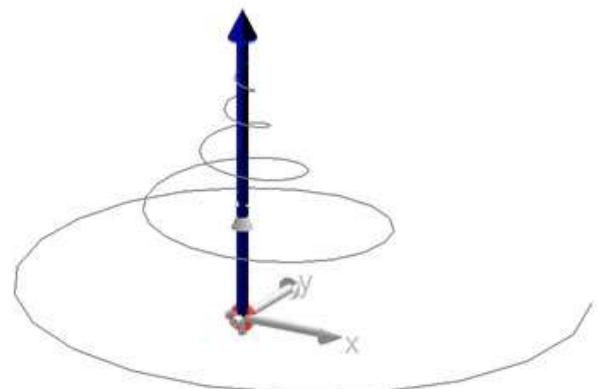
- Kohärente Rotation der Spin-Verteilung
- Auslenkung der Magnetisierung

# Das MR-Signal

- Die rotierende Magnetisierung **induziert** in einer Spule senkrecht zur Transversalebene eine **elektrische Spannung**
  - Hierauf basieren Messungen des MR-Signals
  - Gleiches Grundprinzip wie beim Fahrrad-Dynamo



# Relaxation



- Nach der Anregung kehrt die Magnetisierung exponentiell in ihren Gleichgewichtszustand zurück
  - *Longitudinale* Magnetisierung ( $M_z$ ) kehrt mit Zeitkonstante  $T_1$  zum Ausgangswert  $M_0$  zurück
  - *Quermagnetisierung* ( $M_{xy}$ ) zerfällt mit Zeitkonstante  $T_2$
- Nach einer  $90^\circ$ -Anregung:

$$M_z = M_0 \left(1 - e^{-\frac{t}{T_1}}\right)$$

$$M_{xy} = M_0 e^{-\frac{t}{T_2}}$$

# Typische T1- und T2-Zeiten im Gewebe

- Unterschiede in T1 und T2 ermöglichen Bildkontrast in der MRT:
- T1 (in ms):

	0.2 T	1.0 T	1.5 T
Muskel	370	730	863
Weisse Substanz	388	680	783
Graue Substanz	492	809	917
Liquor	1400	2500	3000

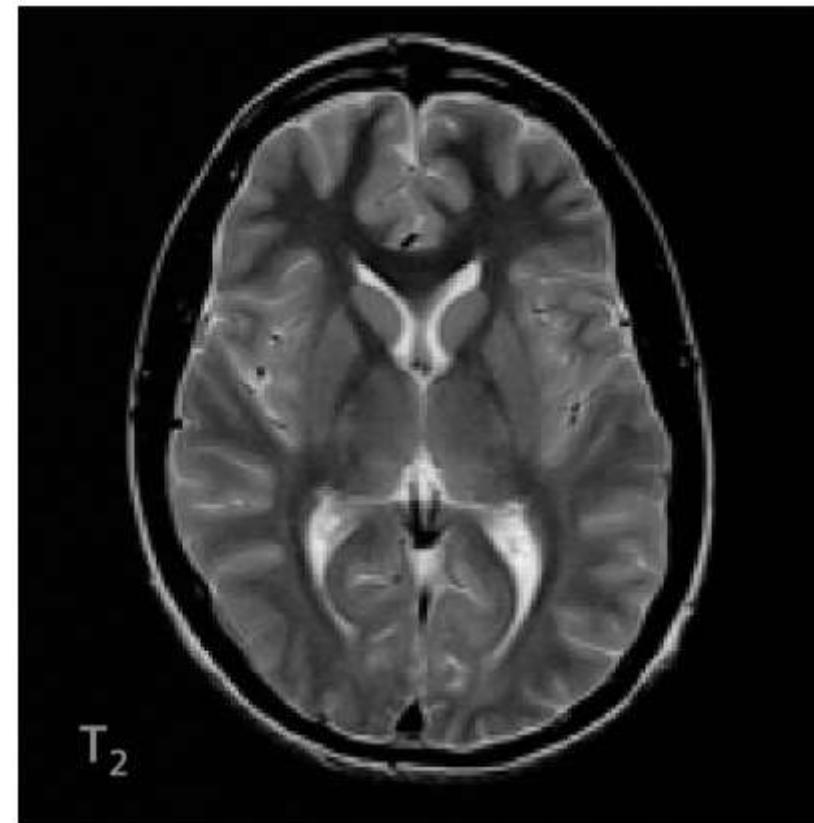
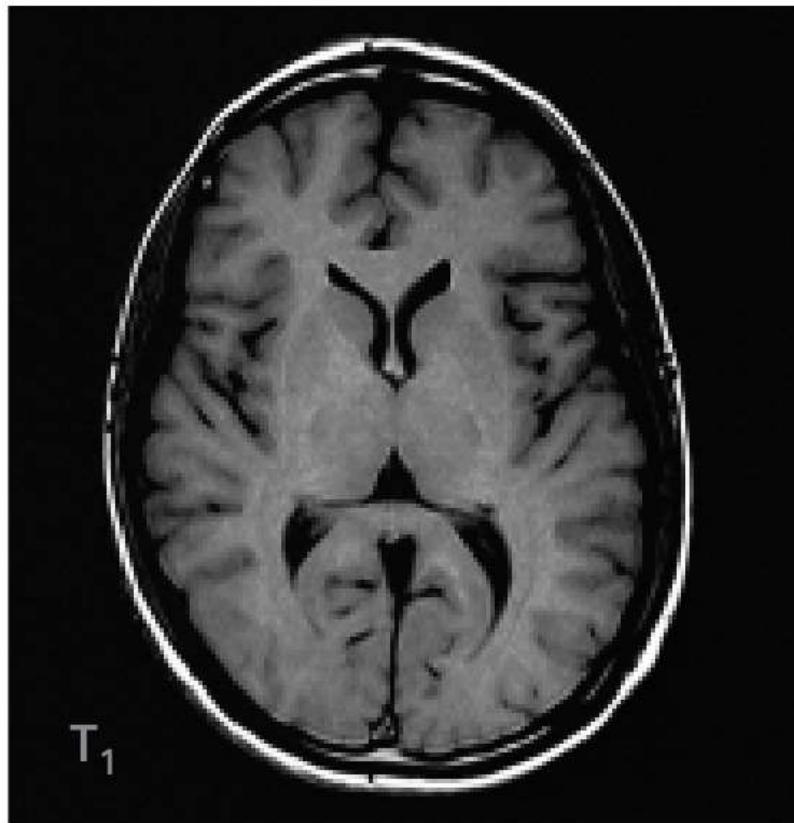
- T2 (in ms):

Muskel	47
Weisse Substanz	92
Graue Substanz	101
Liquor	1400

*Hinweis:* Inhomogenitäten des Magnetfelds beschleunigen die transversale Relaxation. Die effektive Zeitkonstante wird als T2\* bezeichnet.

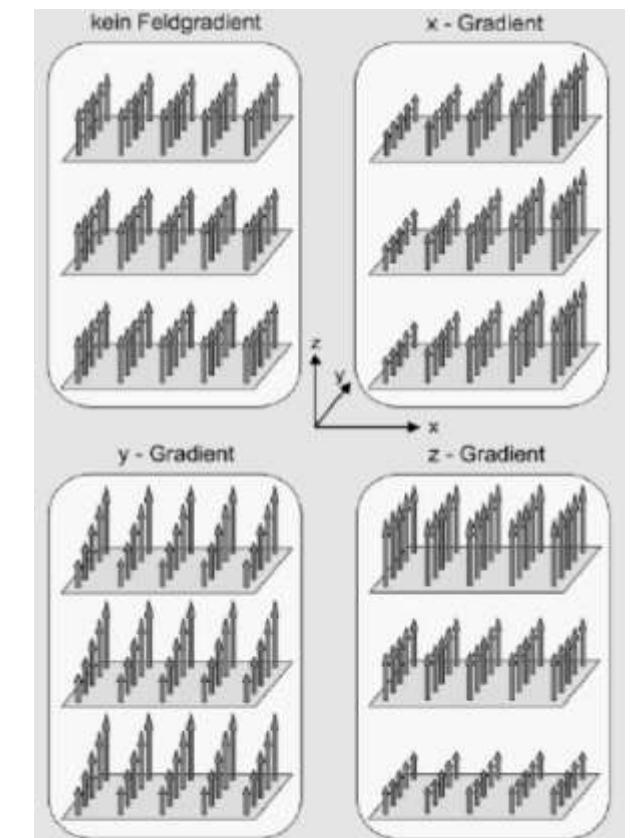
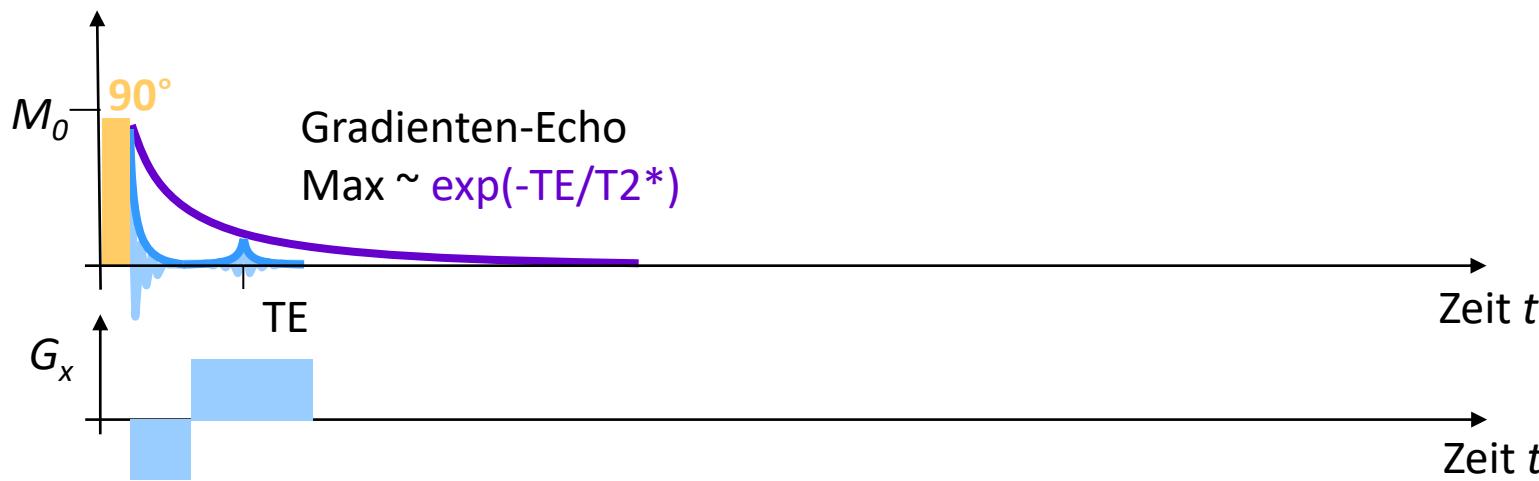
# Beispiele: T1/T2-Wichtung

- In T1-gewichteten Bildern erscheint langes T1 dunkel
- In T2-gewichteten Bildern erscheint langes T2 hell

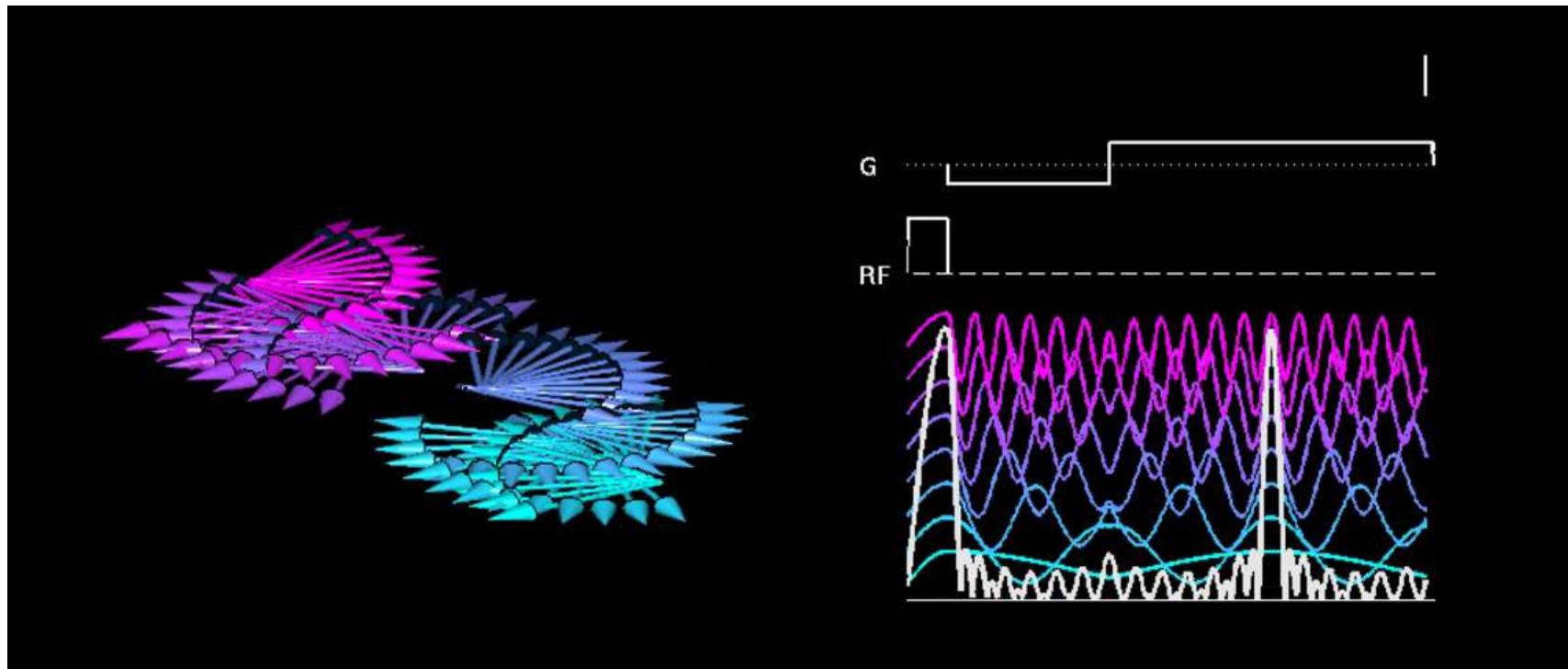


# Gradienten-Echos

- MR-Scanner messen idR nicht das MR-Signal direkt nach der Anregung („Free induction decay“), sondern ein **Echo**
  - Ermöglicht verschiedene Kontraste und vermeidet Störungen
  - Ermöglicht Bildgebung (sehen wir gleich)
- Beispiel **Gradienten-Echo**: Durch Feldgradienten können wir die Kerne gezielt dephasieren und anschließend wieder rephasieren



# Illustration: Gradienten-Echo

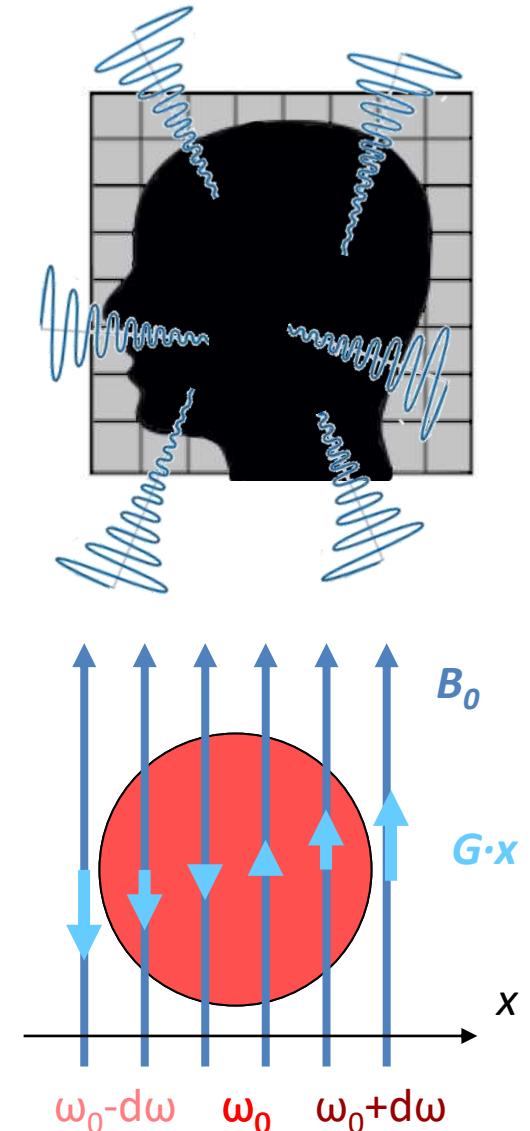


# Grundidee der MR-Tomographie

- Bisher tragen alle Wasserstoff-Kerne im Volumen auf dieselbe Weise zum Signal bei
- Zur Bildgebung (Tomographie) müssen wir das Signal in Anteile zerlegen, die von unterschiedlichen Positionen im Raum stammen
- Dazu nutzen wir verschiedene Varianten desselben Grundprinzips:

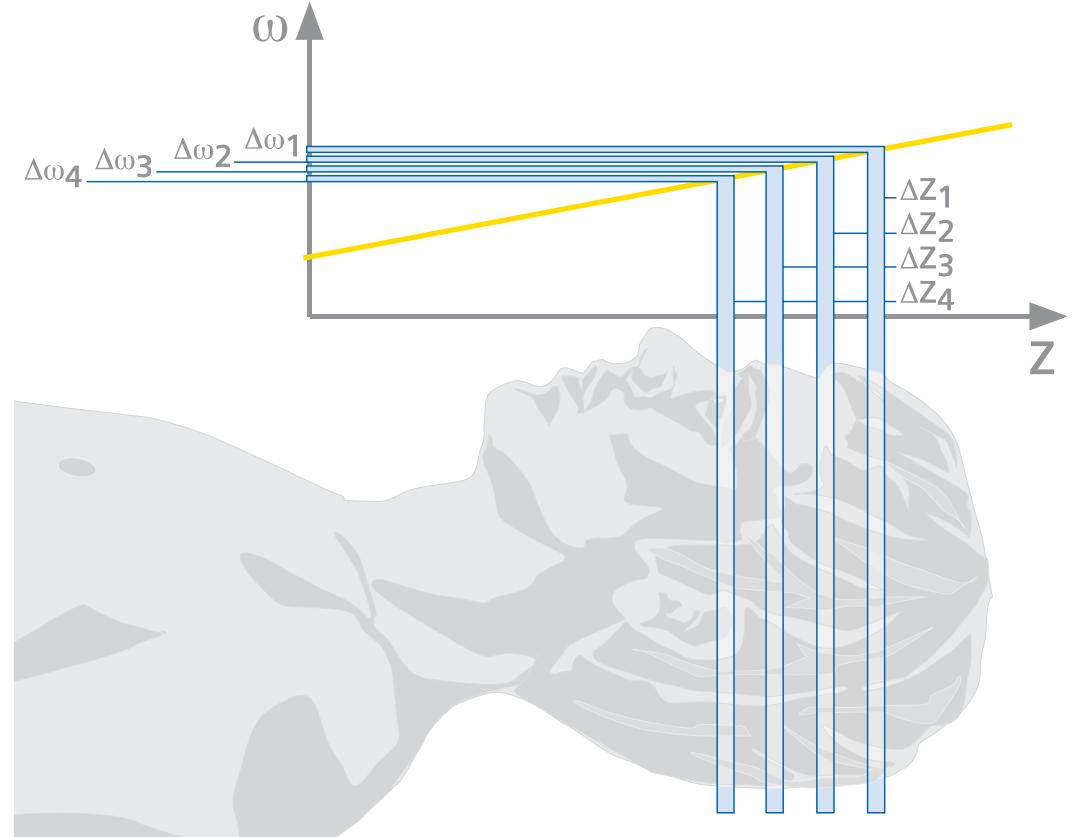
**Wir variieren die Larmor-Frequenz mittels Feldgradienten!**

- *Erinnerung:* Feldgradienten ändern die Stärke des Magnetfelds. Diese bestimmt die Larmor-Frequenz.



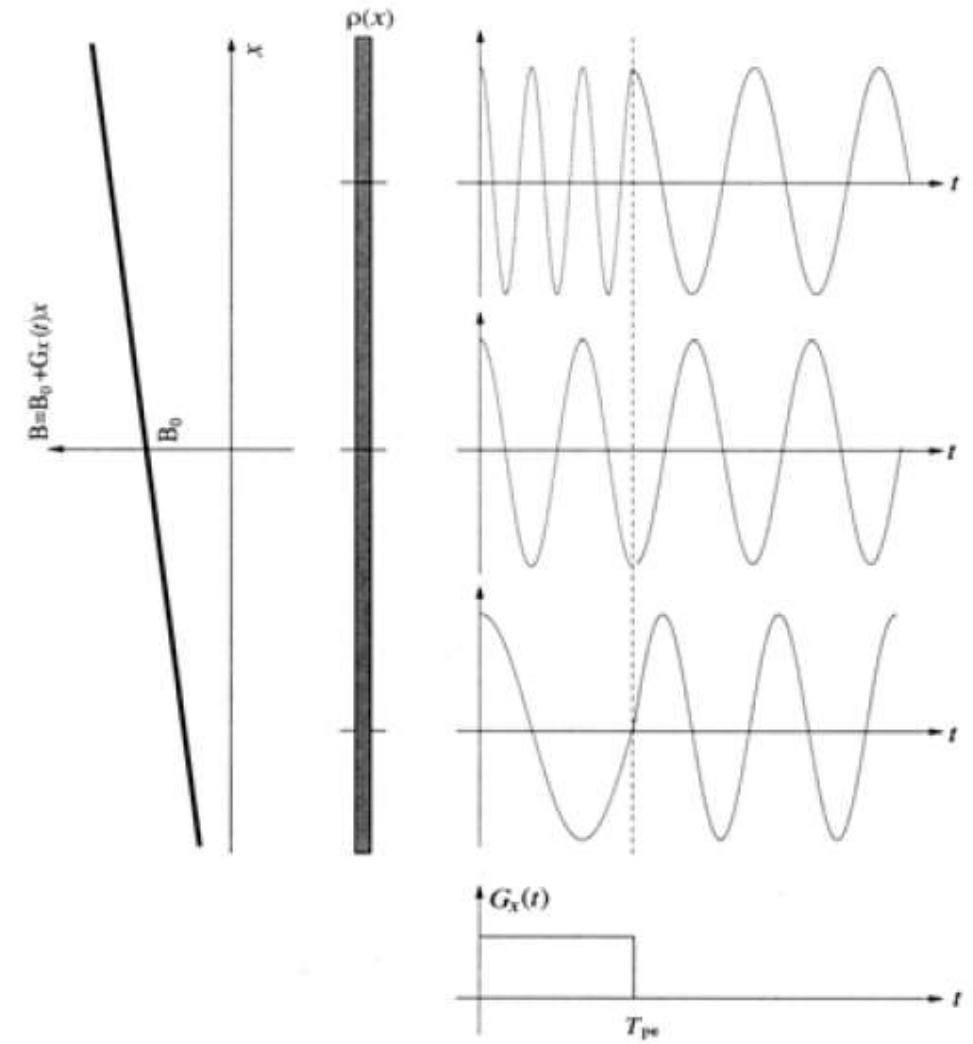
# 1. Selektive Anregung

- Die **selektive Anregung** erzeugt nur in einer einzelnen Schicht eine messbare Quermagnetisierung
  - Moduliert *während der Anregung* die Resonanzfrequenz durch einen Feldgradienten
  - HF-Pulse begrenzter Bandbreite regen daher nur Schichten begrenzter Dicke an



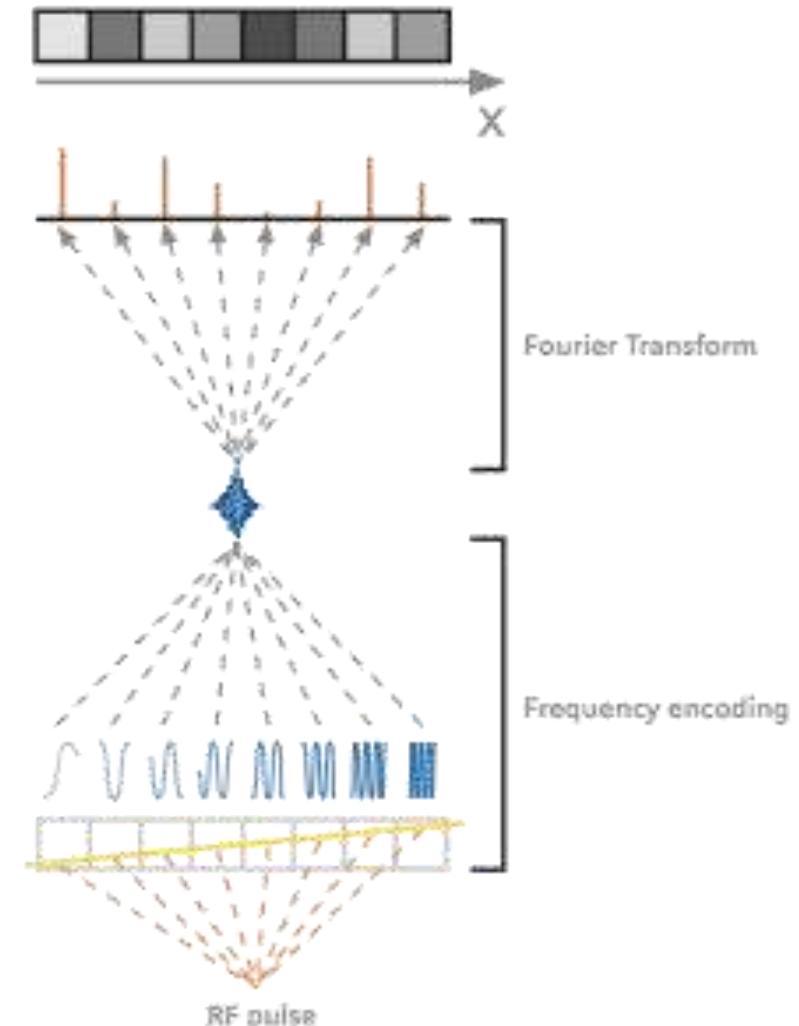
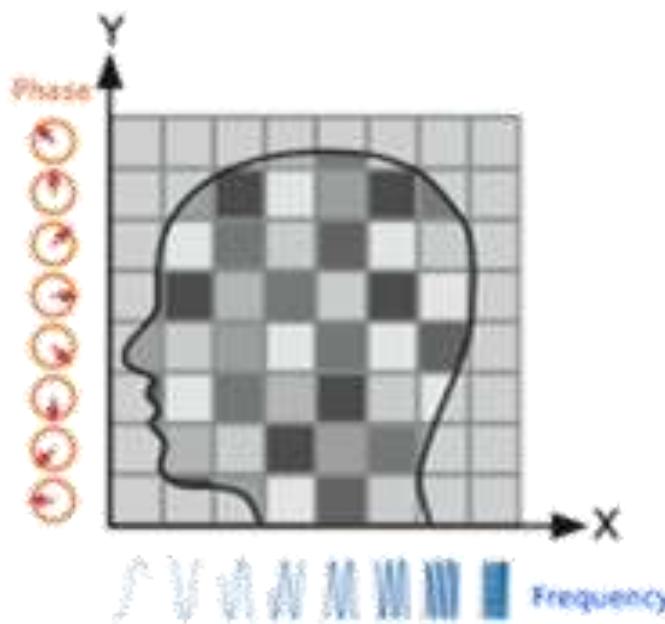
## 2. Phasenkodierung

- **Phasenkodierung** moduliert durch Feldgradienten die Lamorfrequenz *zwischen Anregung und Auslesen*
  - Signale von verschiedenen Positionen entlang der *Phasenkodierrichtung* unterscheiden sich durch ihre Phase
  - Wiederholte Messung mit verschieden starken Phasenkodiergradienten ermöglicht die Rekonstruktion der Signalanteile von verschiedenen Positionen

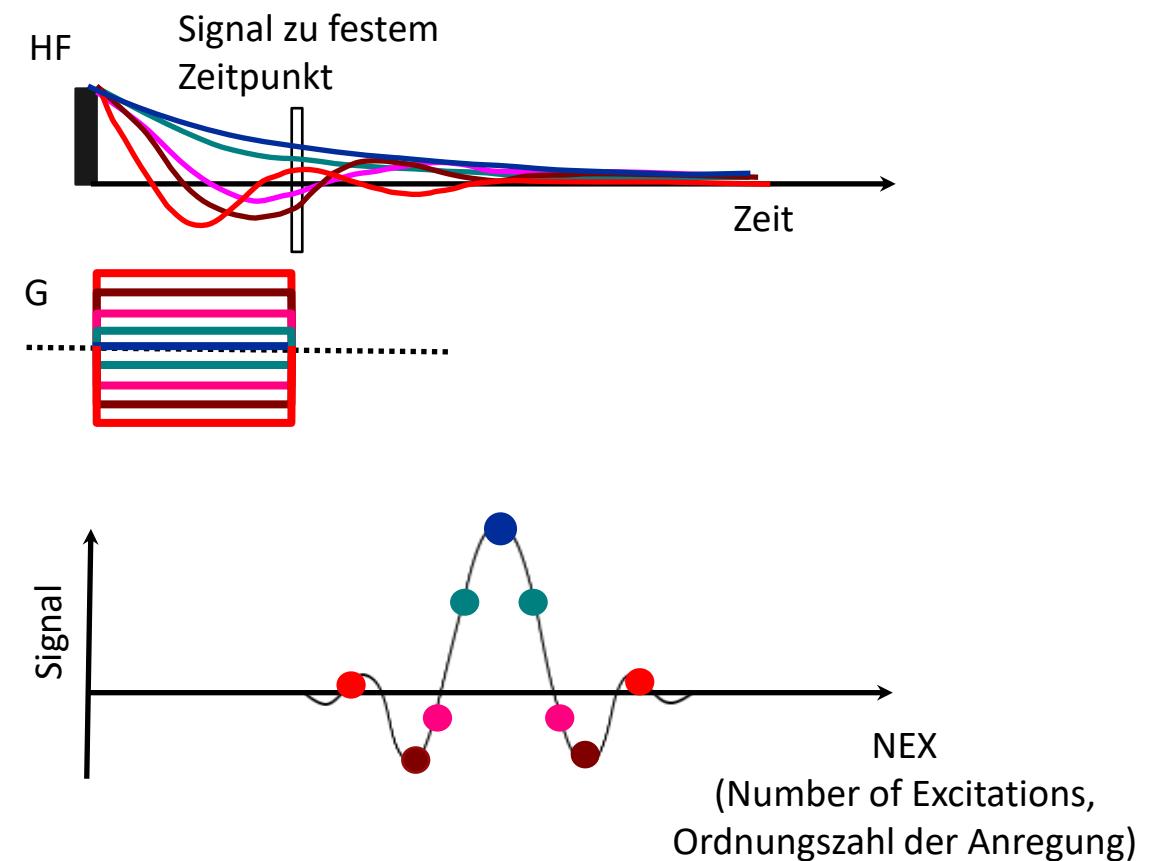
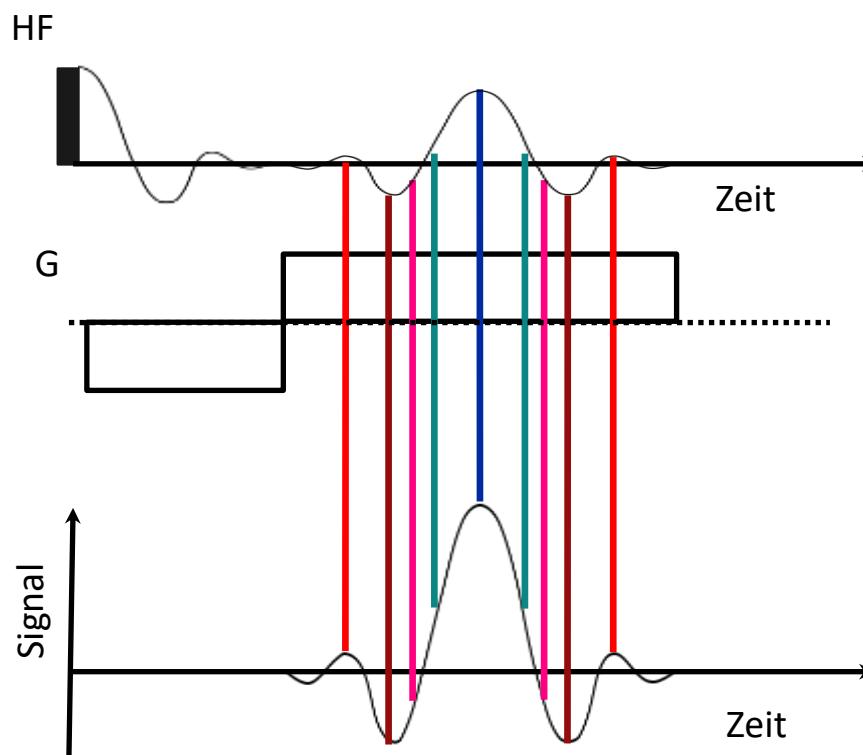


### 3. Frequenzkodierung

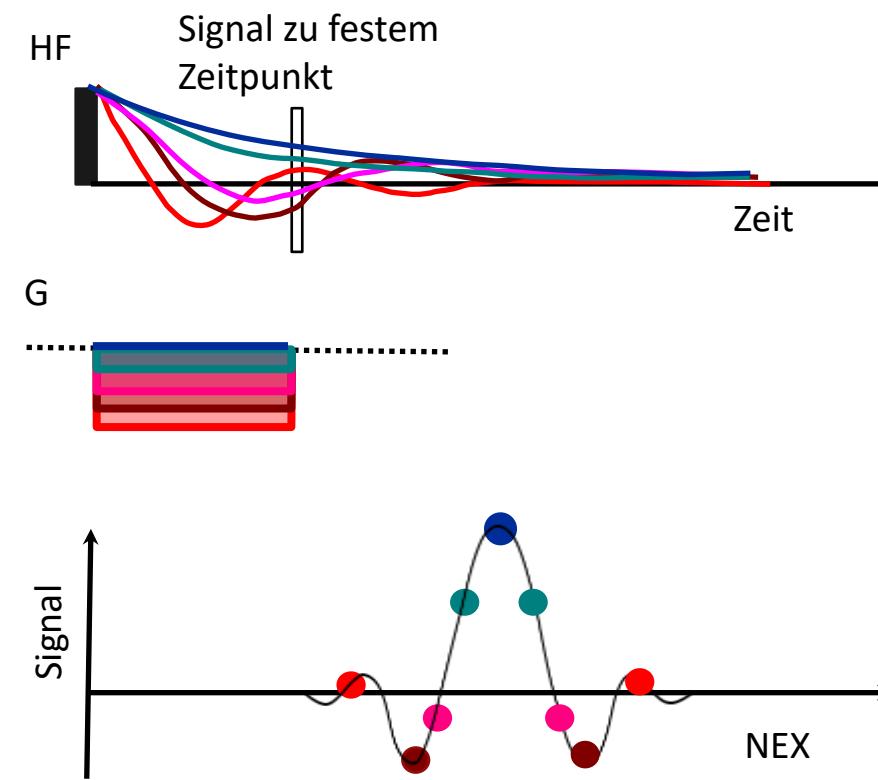
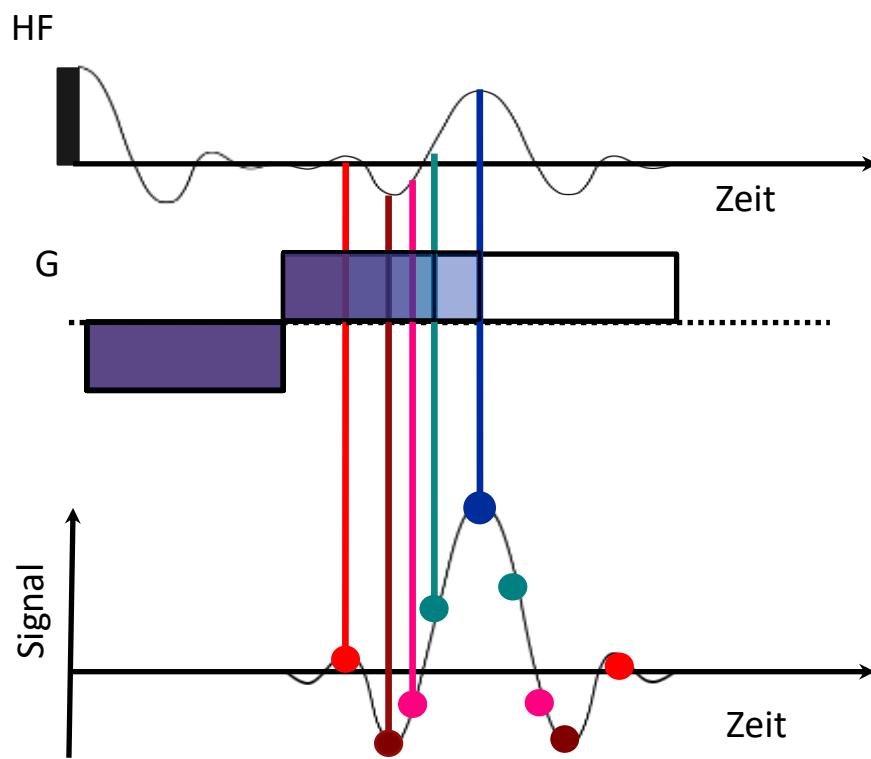
- **Frequenzkodierung** moduliert während des *Auslesens* die Larmorfrequenz durch einen Feldgradienten
  - Fourier-Transformation trennt das Gesamtsignal in Beiträge entlang der *Frequenzkodierrichtung* auf



# Illustration: Frequenz- vs. Phasenkodierung



# Illustration: Frequenz- vs. Phasenkodierung



# Signalgleichung mit Selektiver Anregung

- Die Signalgleichung nach einem  $90^\circ$ -Puls lautet:

$$s(t) = \iiint_{xyz} M_0(x, y, z) \exp\left(-\frac{t}{T_2^*}\right) \exp(-i\omega_0 t) dx dy dz$$

Vernachlässigt!

$\uparrow$        $\uparrow$        $\uparrow$

Gleichgewichtsmagnetisierung  
~ Protonendichte      Transversale Relaxation      Präzession des magnetischen Moments

- 2D-Integral nach selektiver Anregung:

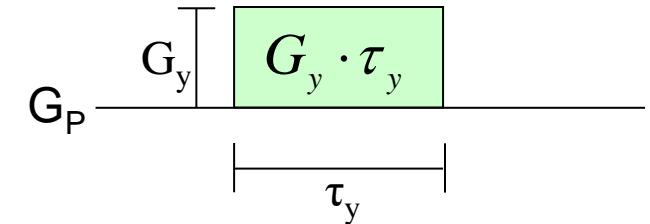
$$s(t) = \iint_{xy} m(x, y) \exp(-i\omega_0 t) dx dy$$

$z_0 + \frac{\Delta z}{2}$

mit  $m(x, y) = \int_{z_0 - \frac{\Delta z}{2}}^{z_0 + \frac{\Delta z}{2}} M_0(x, y, z) dz$

# Signalgleichung: Phasenkodierung

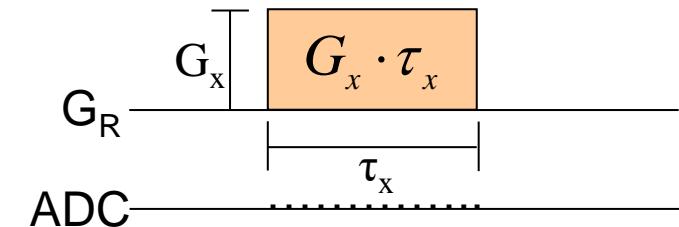
Ein linearer Feldgradient in y-Richtung zwischen Anregung und Auslesen ändert die Signalgleichung wie folgt:



$$s(G_y, \tau_y, t) = \iint_{xy} m(x, y) \exp(-i\omega_0 t) \exp(-i\gamma G_y \tau_y y) dx dy$$

# Signalgleichung: Frequenzkodierung

Ein linearer Feldgradient in x-Richtung während des Auslesens ändert die Signalgleichung wie folgt:



$$s(G_x, G_y, \tau_x, \tau_y, t) =$$

$$\iint_{xy} m(x, y) \exp(-i\omega_0 t) \exp(-i\gamma G_x \tau_x x) \exp(-i\gamma G_y \tau_y y) dx dy$$

# Signalgleichung: Demodulation

$$\iint_{xy} m(x, y) \exp(-i\omega_0 t) \exp(-i\gamma G_x \tau_x x) \exp(-i\gamma G_y \tau_y y) dxdy$$

||

Demodulation (mit Frequenz  $\omega_0$ )

(Wechsel in rotierendes Referenzsystem)

↓

$$\iint_{xy} m(x, y) \cancel{\exp(-i\omega_0 t)} \exp(-i\gamma G_x \tau_x x) \exp(-i\gamma G_y \tau_y y) dxdy$$

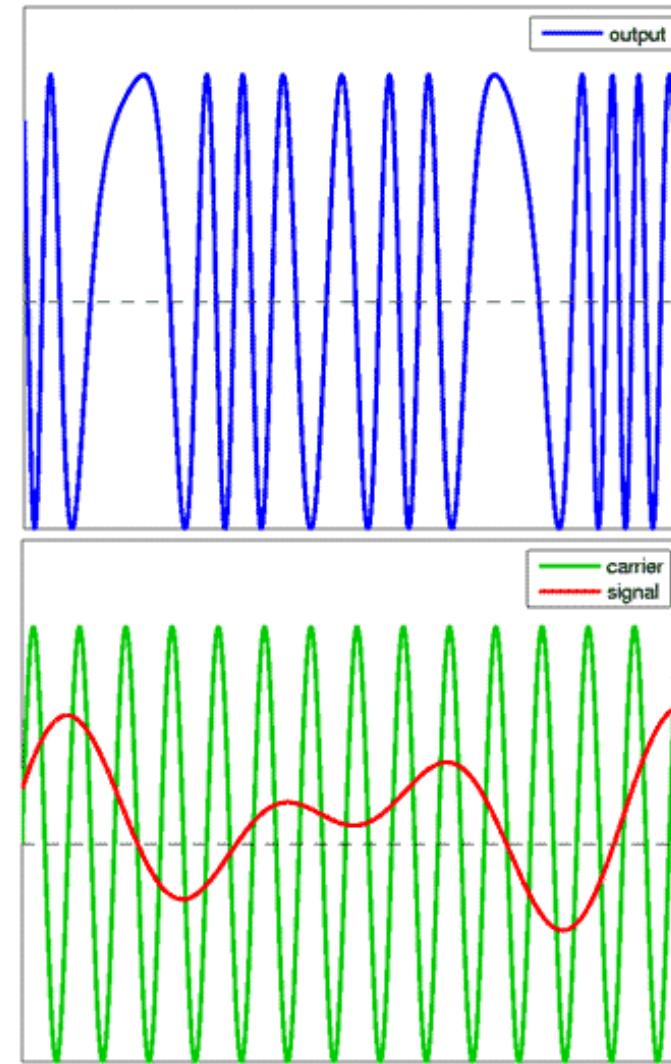


Figure from: [http://en.wikipedia.org/wiki/Phase\\_modulation](http://en.wikipedia.org/wiki/Phase_modulation)

# Signalgleichung: k-Raum-Formalismus

Das demodulierte Signal einer selektiv angeregten Schicht, die vor und während des Auslesens linearen Feldgradienten in  $y$ - und  $x$ -Richtung ausgesetzt war lautet demnach:

$$s(G_x, G_y, \tau_x, \tau_y) = \iint_{xy} m(x, y) \exp(-i\gamma G_x \tau_x x) \exp(-i\gamma G_y \tau_y y) dx dy$$

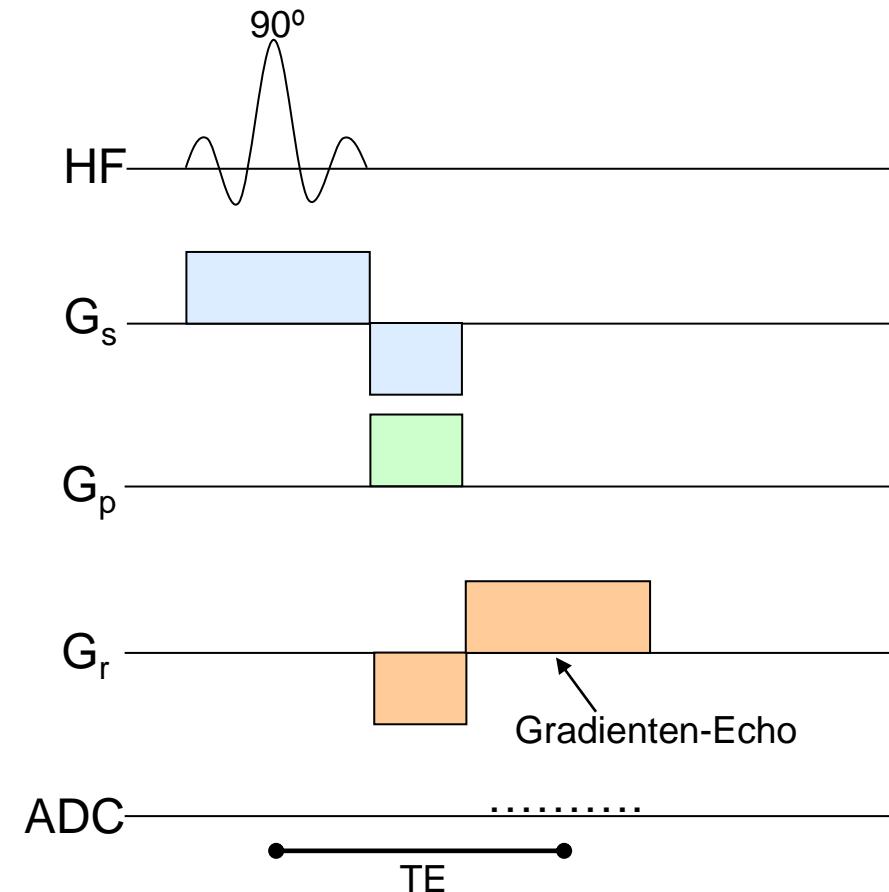
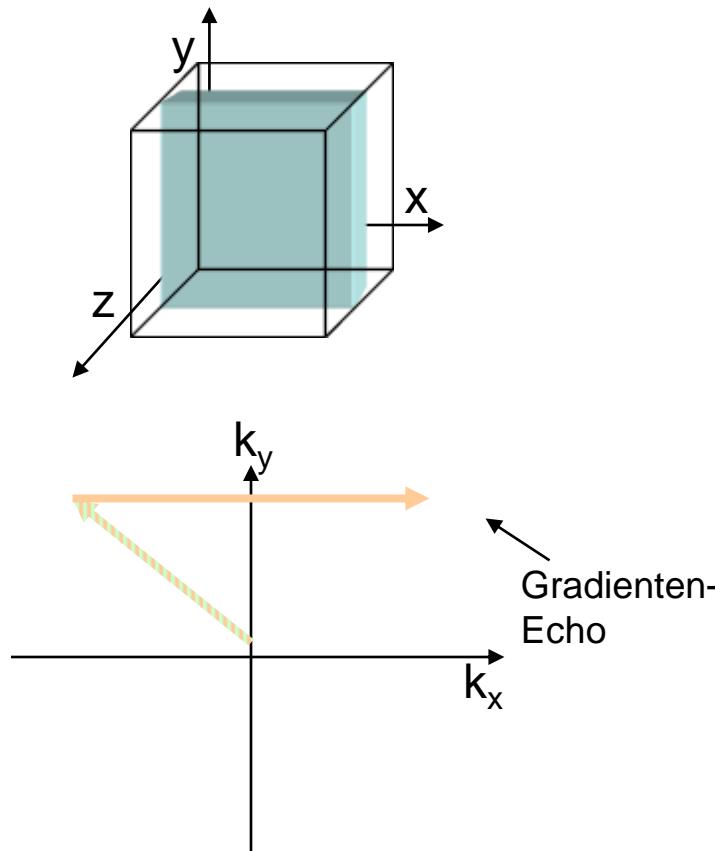
Das Signal ist die Fourier-Transformation der Magnetisierung!

Definition:  $k_x(G_x, \tau_x) = \frac{\gamma}{2\pi} G_x \tau_x, \quad k_y(G_y, \tau_y) = \frac{\gamma}{2\pi} G_y \tau_y$



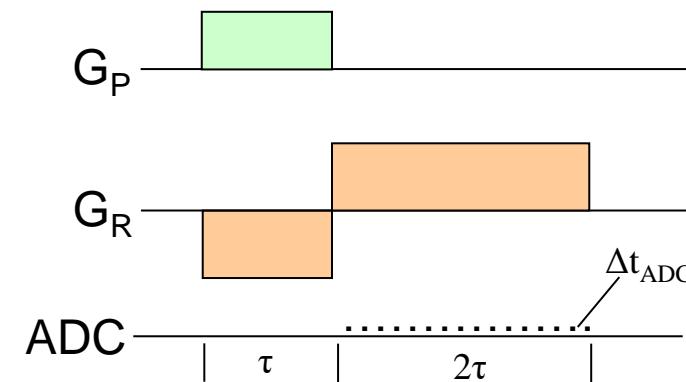
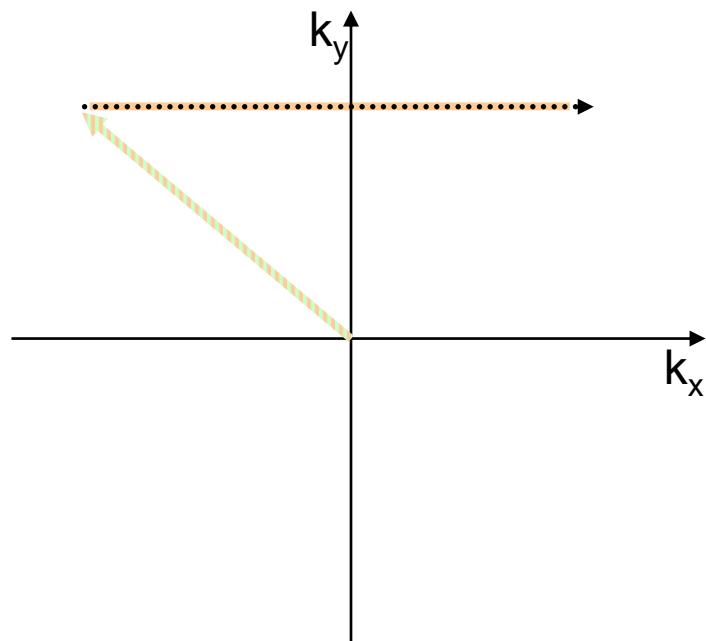
$$s(k_x, k_y) = \iint_{xy} m(x, y) \exp(-i2\pi k_x x) \exp(-i2\pi k_y y) dx dy$$

# MRT mit Gradienten-Echo



# Abtastung des $k$ -Raums

Durch die Kombination von Phasen- und Frequenzkodierung zeichnen wir eine Zeile im  $k$ -Raum auf:

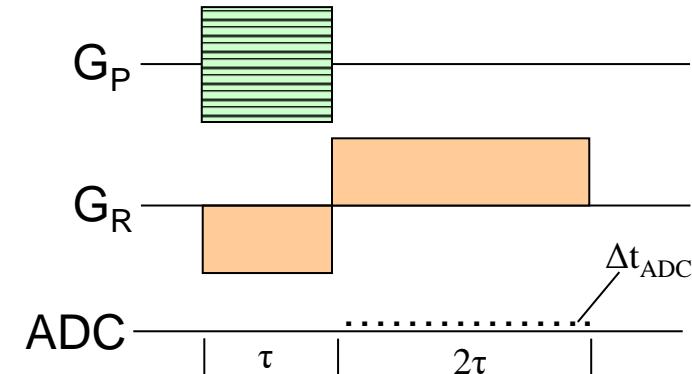
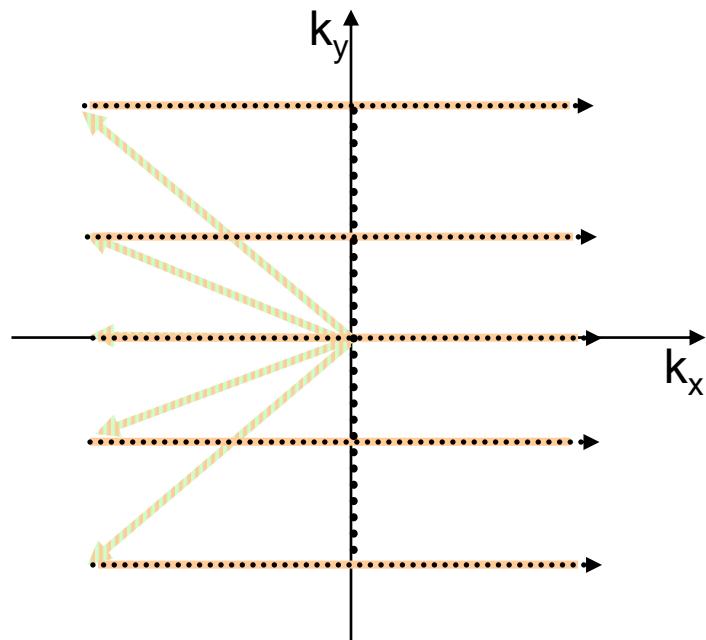


$$k_x = \frac{\gamma}{2\pi} (-G_x \tau + G_x \{0, \Delta t_{ADC}, 2\Delta t_{ADC}, \dots, 2\tau\})$$

$$k_y = \frac{\gamma}{2\pi} G_y \tau = const.$$

# Abtastung des $k$ -Raums

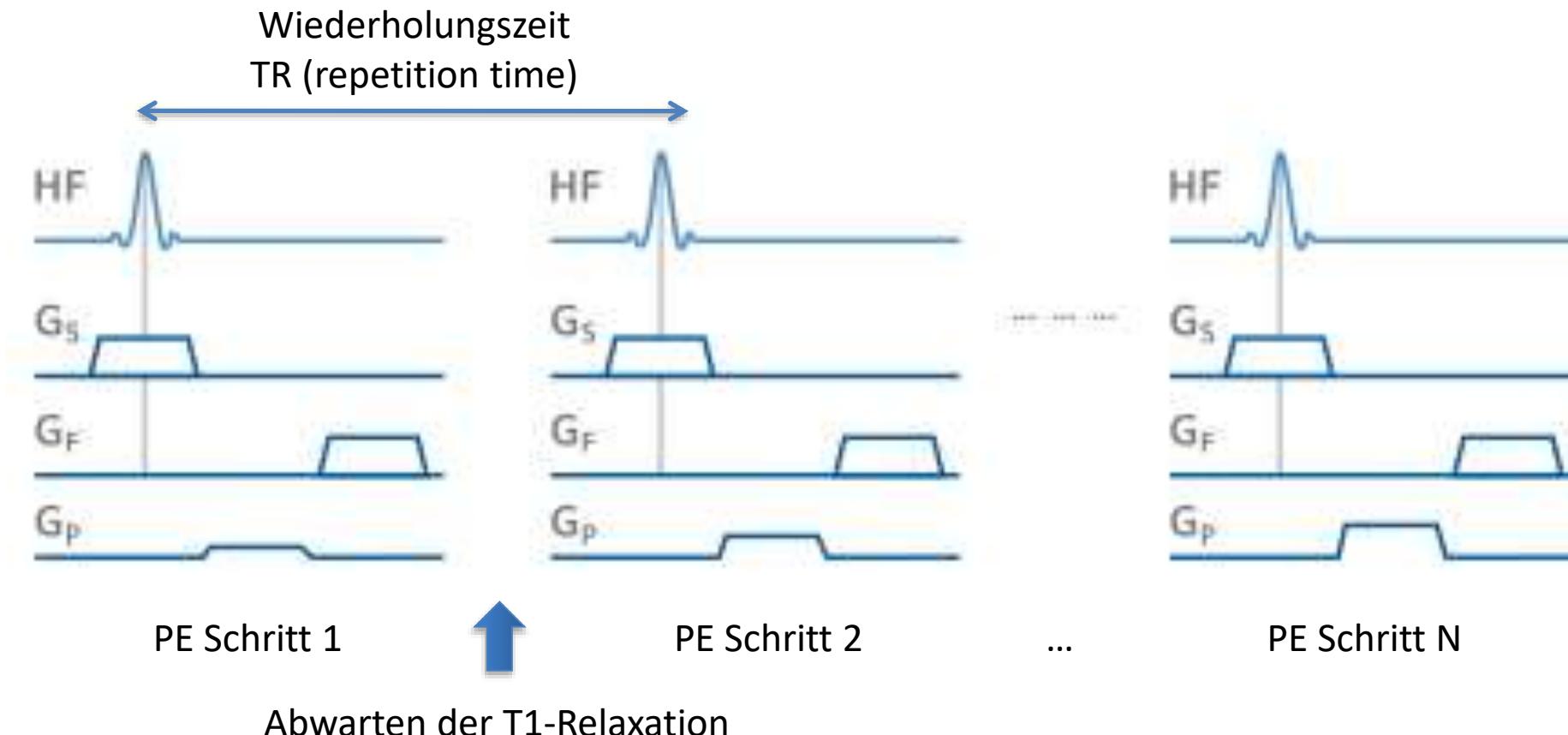
Um den **kompletten zweidimensionalen  $k$ -Raum abzutasten** wird die Sequenz mit anderer Stärke des  $y$ -Gradienten wiederholt



$$k_x = \frac{\gamma}{2\pi} (-G_x \tau + G_x \{0, \Delta t_{ADC}, 2\Delta t_{ADC}, \dots, 2\tau\})$$

$$k_y = \frac{\gamma}{2\pi} \{G_{y,\max}, \dots, G_{y,\min}\} \tau$$

# Gesamtablauf: Phasen- und Frequenzkodierung



**Die Gesamtzeit einer MRT-Aufnahme ergibt sich als Produkt der Wiederholungszeit TR und der Zahl der Phasenkodierschritte.**

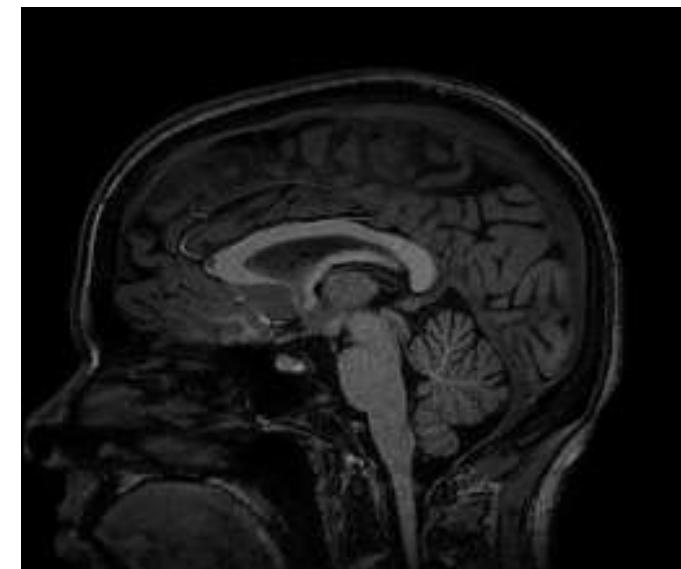
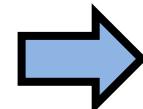
# MRT: Bildrekonstruktion

Sobald der  $k$ -Raum vollständig gefüllt ist, kann man das Bild durch eine **inverse 2D Fourier-Transformation** rekonstruieren

- Theoretisch sollte es reichen, die Hälfte des  $k$ -Raums abzutasten
- Vollständige Abtastung verringert Einfluss von Rauschen und Artefakten
- Absolutbetrag der Rekonstruktion ergibt ein reellwertiges Bild



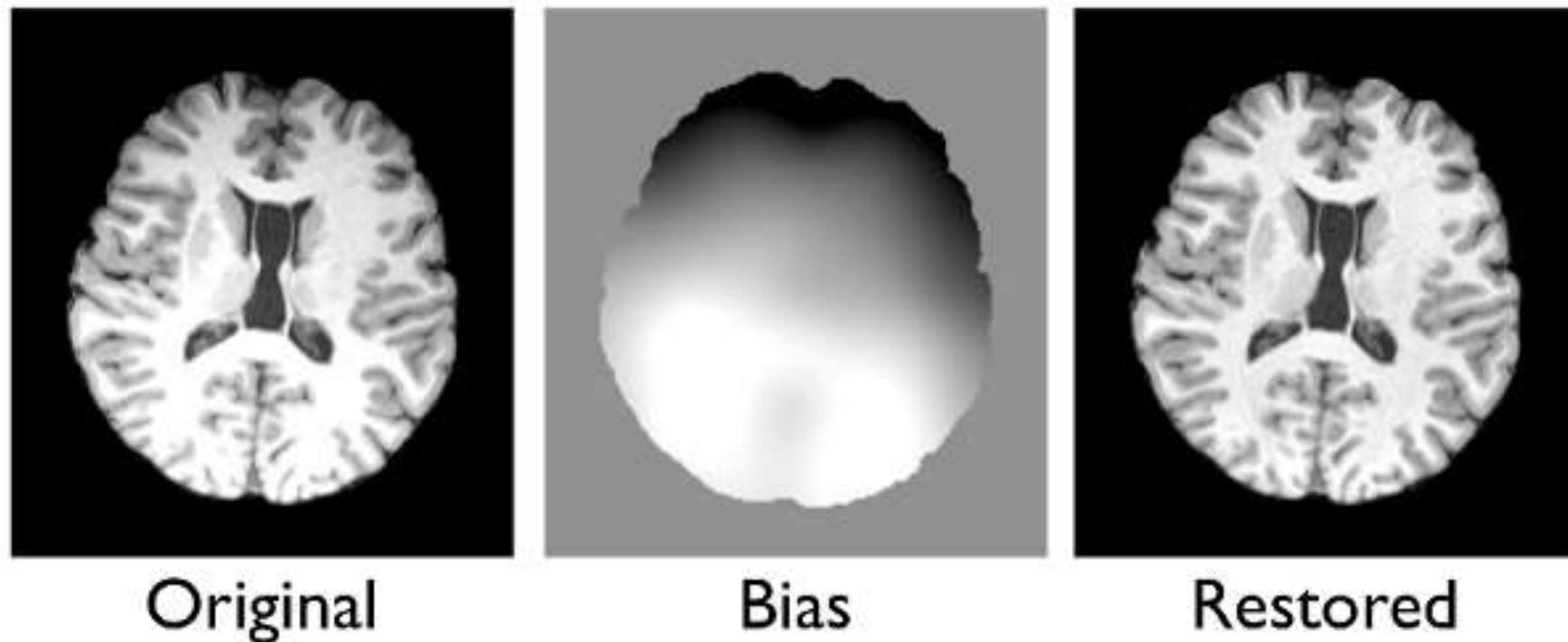
$k$ -Raum



Bildraum

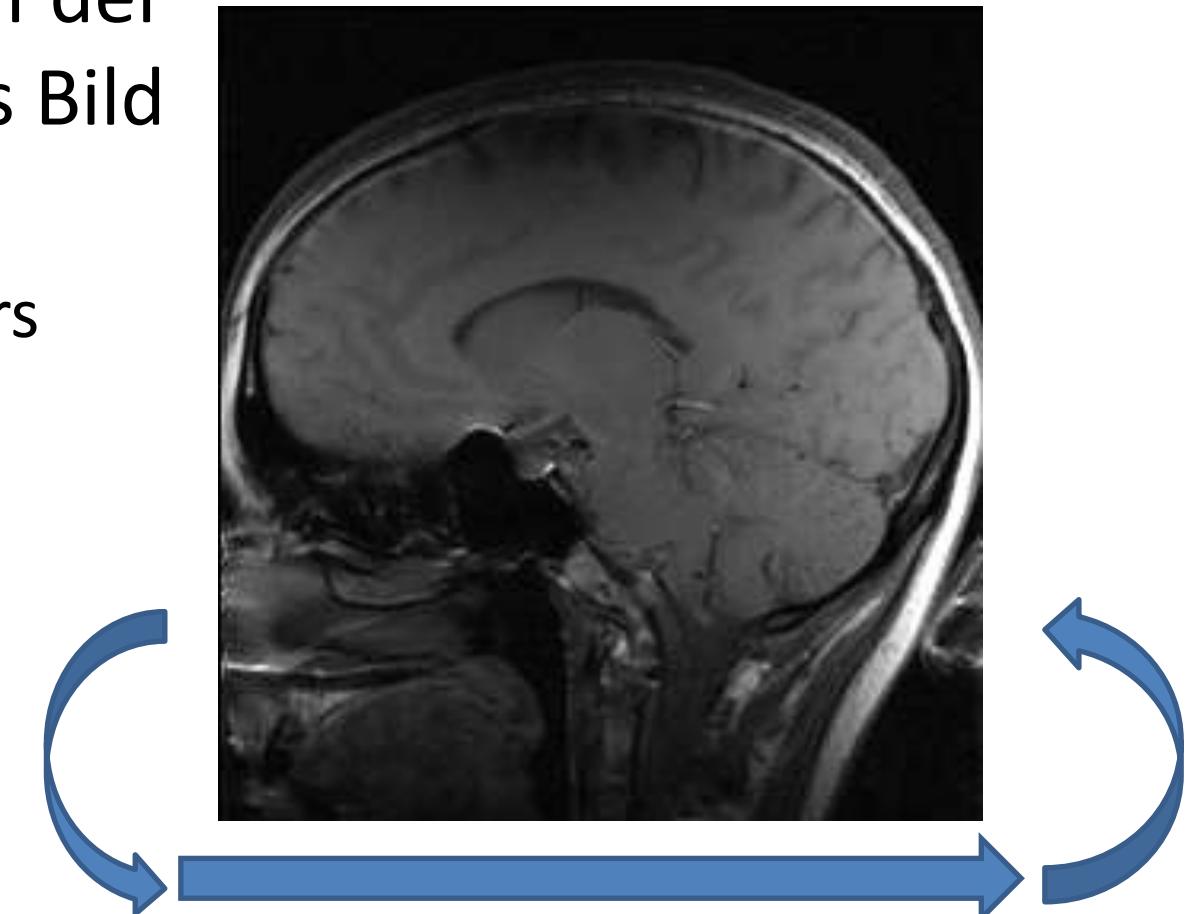
# MRT-Artefakte: Inhomogenitäten

- Das Magnetfeld in MR-Scannern ist nicht perfekt **homogen**, der HF-Puls nicht überall genau gleich stark
  - Einflüsse der Umgebung und des Probanden selbst
  - Anregungswinkel  $\alpha$  ist nicht überall gleich
  - Führt zu ungleichmäßiger „Ausleuchtung“ des Bildes



# Einfaltungsartefakte

- Körperteile/Objekte im Scanner, aber außerhalb des Untersuchungsfeldes werden auf der gegenüberliegenden Seite in das Bild „eingefaltet“
  - In Phasenkodierrichtung besonders schwierig zu unterdrücken



# Zusammenfassung: Magnetresonanztomographie

- MRT eignet sich besonders für **Weichgewebe** und **Organe**
- Ein **MRT-Scan** besteht aus folgenden Schritten:
  1. **Polarisierung** von Wasserstoff-Kernen durch starkes Magnetfeld
  2. **Selektive Anregung** der Kerne durch einen resonanten HF-Puls
  3. **Phasen- und Frequenzkodierung** der Position in 2D-Schicht
  4. **Aufzeichnung des Echoes** und Eintrag in den  $k$ -Raum
    - Vollständiges Füllen des  $k$ -Raums durch Wiederholung der Schritte 2-4
  5. **Rekonstruktion** durch inverse Fourier-Transformation
- **Bildkontrast** ergibt sich durch verschiedene Protonendichten sowie **Relaxationszeiten**
  - Longitudinal:  $T_1$ , Transversal:  $T_2$  bzw.  $T_2^*$

## **3.5 Ultraschall**

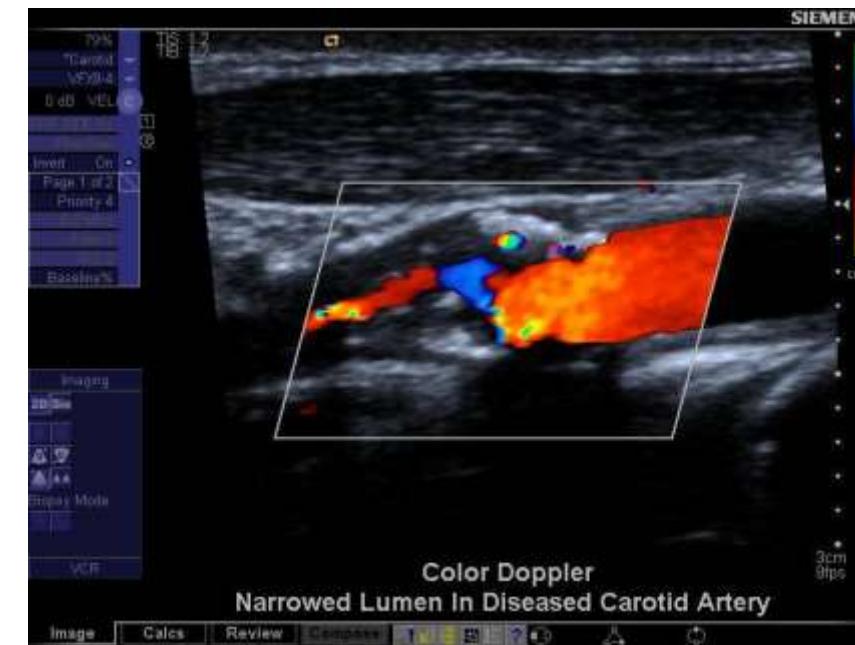
# Vorteile von Ultraschall

- Bildgebung mit **Ultraschall** (Sonographie) ist mobil, preiswert und sicher
- *Grundprinzip:* Ein **Ultraschall-Wandler** strahlt Schallwellen oberhalb des hörbaren Bereichs (1-40 MHz) in den Körper und erzeugt aus den zurückgeworfenen Echos ein Bild
  - Der Doppler-Effekt ermöglicht bei Bedarf die Darstellung von Blutfluss
- *Hinweis:* Aus Zeitgründen behandeln wir dieses Thema knapp.



# Anwendungen von Ultraschall

Zum Einsatz kommt Ultraschall u.a. zur Untersuchung des Herzens, Verdauungstrakts, von Blutgefäßen/Blutfluss und von ungeborenen Kindern



# Schallimpedanz

- Die **Schallimpedanz**  $Z = \rho \cdot c$  beschreibt den Widerstand, den ein Material der Schallausbreitung entgegensetzt
  - Produkt der Dichte  $\rho$  und Schallgeschwindigkeit  $c$

Material	$c$ [m/s]	$\rho$ [g/cm <sup>3</sup> ]	$Z$ [g cm <sup>-2</sup> s <sup>-1</sup> ]
Luft	331	0,0013	43
Fett	1470	0,97	$1,42 \cdot 10^5$
Wasser	1492	0,9982	$1,48 \cdot 10^5$
Hirn	1530	1,02	$1,56 \cdot 10^5$
Muskel	1568	1,04	$1,63 \cdot 10^5$
Knochen	3600	1,7	$6,12 \cdot 10^5$

# Reflexion und Transmission

- Der **Reflexionskoeffizient  $R$**  an der Grenzfläche zweier Materialien mit Impedanzen  $Z_1$  und  $Z_2$  ist (bei senkrechttem Einfall)

$$R = \frac{(Z_2 - Z_1)^2}{(Z_2 + Z_1)^2}$$

- Grenzen mit sehr kleinen Impedanzunterschieden sind kaum sichtbar, sehr große Unterschiede machen alle Strukturen dahinter unsichtbar



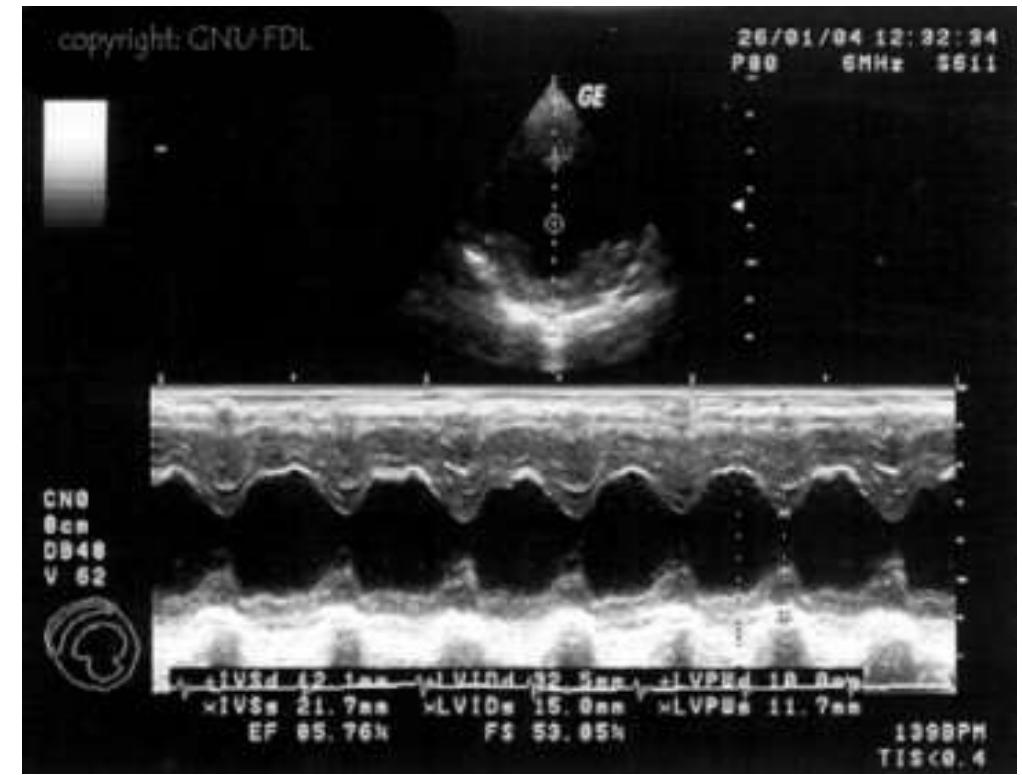
# Eindringtiefe und Auflösung

- Die **Schallintensität** nimmt auf dem Weg durch homogenes Material durch Absorption und Streuung exponentiell ab
  - Höhere Frequenzen haben einen höheren Schwächungskoeffizienten
- Höhere Frequenzen erlauben jedoch auch eine höhere **Ortsauflösung**

Frequenz [MHz]	Eindringtiefe [cm]	Anwendungsbeispiele	Auflösung [mm]	
			lateral	axial
3,5	15	Fetus, Leber, Herz, Niere	1,7	0,5
7,5	7	Prostata	0,8	0,3
10	5	Pankreas (intraoperativ)	0,6	0,2
20	1,2	Auge, Haut	0,4	0,15

# A- und M-Mode

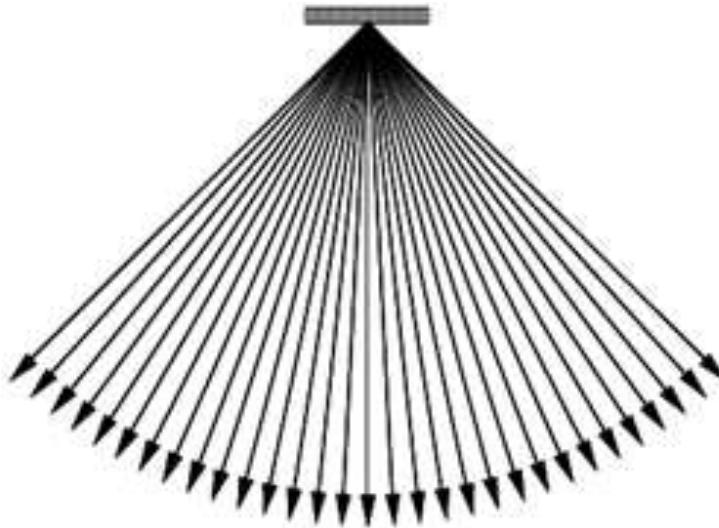
- Im **A-Mode** (Amplitude) werden die Echozeiten eines einzelnen Ultraschall-Strahls in Tiefe umgerechnet und die Intensitäten der Echos dargestellt
  - Zeitabhängige Verstärkung (*engl.* Time Gain Compensation) kompensiert die Abschwächung durch Absorption und Streuung
- Im **M-Mode** (Motion) wird eine A-Mode über die Zeit hinweg als Bild dargestellt



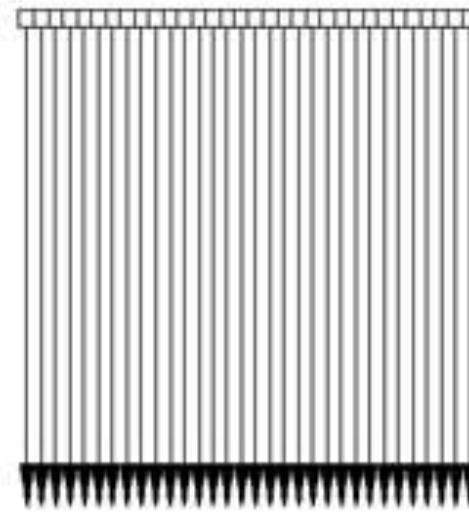
Hundeherz in B-Mode (oben) /  
M-Mode (unten)

# B-Mode

- Der **B-Mode** (Brightness) stellt die Echos eines zweidimensionalen Strahlenfächers als Bild dar
  - Verschiedene Sondentypen, je nach Anwendung
  - Schwenken des 2D-Fächers ermöglicht auch 3D-Bilder



Sektorscanner



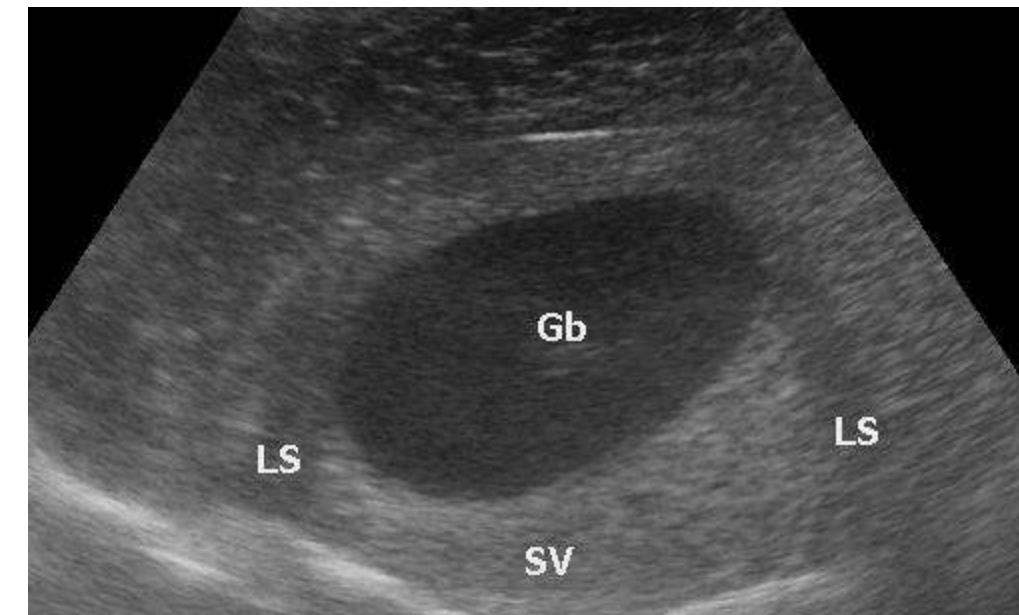
Linearscanner



Konvexscanner

# Bildstörungen im Ultraschall

- Interpretation von Ultraschall erfordert viel Erfahrung
  - Darstellung der Anatomie hängt von Positionierung des Ultraschallkopfs ab
  - Interferenzen der Schallwellen führen zu Bildrauschen (Speckle)
  - Starke Reflexionen erzeugen „Schatten“
  - Mehrfachreflexionen führen zu „Trugbildern“
  - Strukturen hinter schwach dämpfendem Material werden übermäßig hell
  - Unterschiede zwischen angenommener und tatsächlicher (materialabhängiger) Schallgeschwindigkeit verzerrten das Bild



LS = Laterale Schatten, SV = Schallverstärkung

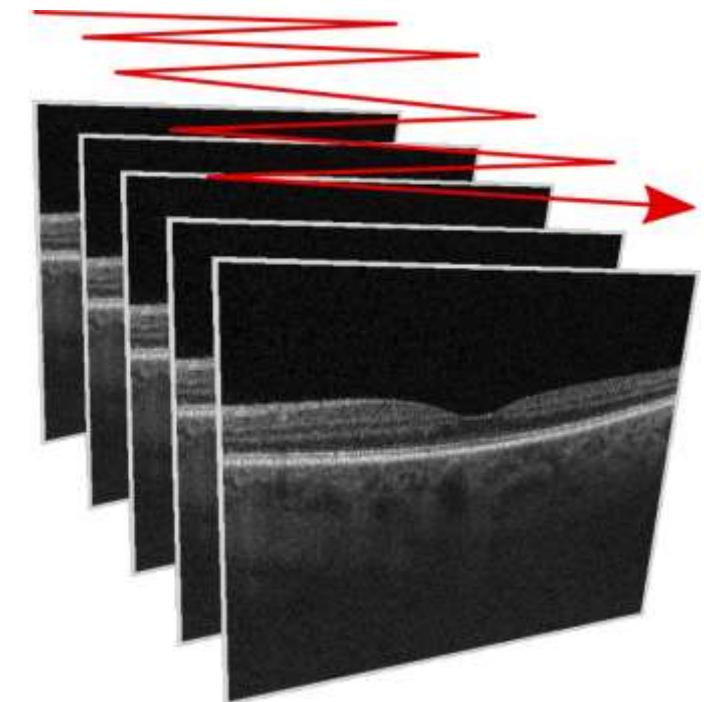
# Zusammenfassung: Sonographie

- Sonographie macht Unterschiede **akustischer Impedanz** an den Grenzflächen verschiedener Gewebe / Materialien sichtbar
- **Schallfrequenz** (im MHz-Bereich) ist anwendungsabhängig
  - Höhere Frequenzen haben bessere Auflösung, geringere Eindringtiefe
- Ultraschall ist bei korrektem Einsatz **unbedenklich**
  - Sicherheitsaspekte: Wärmeentwicklung und Kavitation
- Ultraschall ist relativ **einfach, mobil und breit verfügbar**, die Bilder haben jedoch zahlreiche **Störungen** und sind daher schwer zu interpretieren

## **3.6 Optische Kohärenztomographie**

# Grundprinzip der OCT

- Die **Optische Kohärenztomographie** (*engl.* Optical Coherence Tomography, OCT) bestimmt durch Interferenz kohärenten Lichts den Abstand, aus dem Licht reflektiert wird
  - Grundidee ähnlich wie im Ultraschall, Laser-Licht im Infrarotbereich statt Schallwellen
  - Erreicht Eindringtiefe von wenigen mm, Auflösung im  $\mu\text{m}$ -Bereich
  - Aus Zeitgründen verzichten wir auf weitere Details
- **Anwendungsbereiche:**
  - Untersuchungen der Netzhaut
  - Diagnostik der Haut
  - Intravaskuläre Bildgebung



# Anwendungsbeispiel: OCT

- Unsere Gruppe beschäftigt sich aktuell u.a. mit der Quantifizierung krankhafter Veränderungen im Kontext der **altersbedingten Makuladegeneration** (AMD), z.B. Drusen

Normale Sicht



Sicht mit AMD

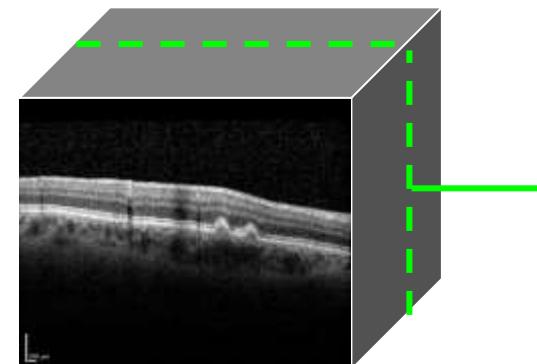


Bildquelle: National Eye Institute,  
National Institutes of Health

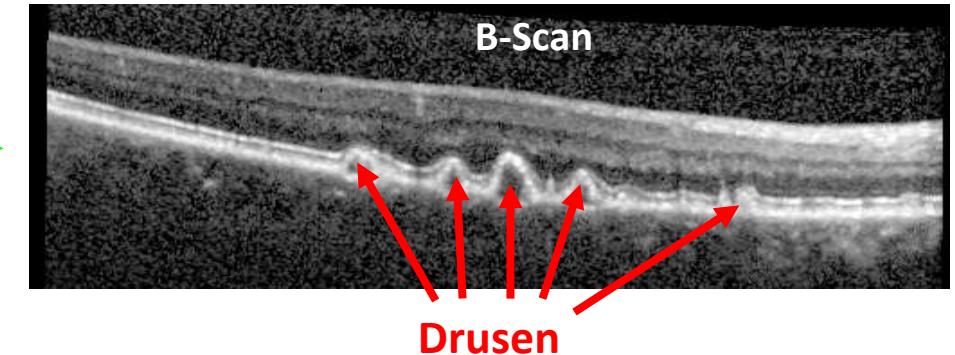


Bildquelle: Heidelberg Engineering

OCT Scan



B-Scan



# Zum Nach- und Weiterlesen

- Olaf Dössel: Bildgebende Verfahren in der Medizin. Springer, 2016
- Andreas Maier et al. (Eds.): Medical Imaging Systems. Springer LNCS 11111, 2018 (*Open Access Book*)
- B. Preim, C. Botha: *Visual Computing for Medicine: Theory, Algorithms, and Applications*, Morgan Kaufmann, 2014
  - E-book available within UBonnn network!
- M. A. Flower (Ed): *Webb's Physics of Medical Imaging*, 2nd edition, CRC Press 2012
- *Magnets, Spins and Resonances*, Siemens Healthcare (available online)

# Kapitel 4: Bildsegmentierung

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

2./9./16. Dezember 2024

## **4.1 Problemstellung und Evaluierung**

# Zielsetzung

- Grundlegende Aufgabe bei der **Interpretation** von Bildern ist das Erkennen relevanter **Bildinhalte** (z.B. Organ, Tumor, Gewebetyp)
  - Diese lassen sich meist nicht aus den Intensitäten einzelner Pixel schließen, sondern erfordern die sinnvolle **Gruppierung** von Pixeln
- **Segmentierung** bezeichnet eine vollständige und überdeckungsfreie Zerlegung eines Bildes nach bestimmten Kriterien
  - z.B. Regionen- oder Kantenbasiert
  - Berücksichtigung von Vorwissen, z.B. über die gesuchte Form
- Bildet u.a. die Grundlage von **Quantifizierung** (z.B. Volumen, Form, Textur) und **Behandlungsplänen**

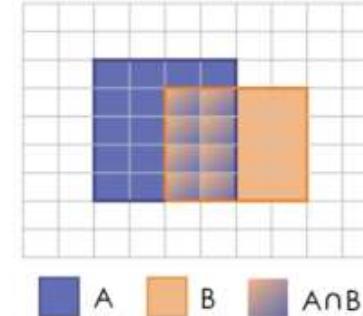
# Verschiedene Arten der Segmentierung

- **Semantische Segmentierung** ordnet jedem Pixel eine klare Bedeutung zu („Label“)
- **Instanz-Segmentierung** zerlegt ein Bild pixelgenau in einzelne Objekte (z.B. einzelne Zellen in einem Gewebeschnitt)
- Die Kombination beider Aufgaben (Zerlegung in Instanzen und Benennung/Labeling dieser) wird manchmal als **panoptische Segmentierung** bezeichnet

# Evaluierung von Segmentierungen: Überlapp

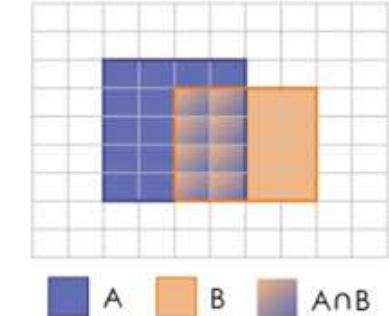
- Den **Überlapp** zwischen
  - Segmentierung A und
  - Referenz („Ground Truth“) Bquantifiziert man häufig per
  - Dice-Score (DSC) oder
  - Verhältnis von Schnitt- und Vereinigungsmenge
    - IoU = Intersection over Union

(a) DSC



$$\begin{aligned} \text{DSC}(A,B) &= \frac{2}{\boxed{\text{A}} + \boxed{\text{B}}} \\ &= \frac{2 |A \cap B|}{|A| + |B|} \end{aligned}$$

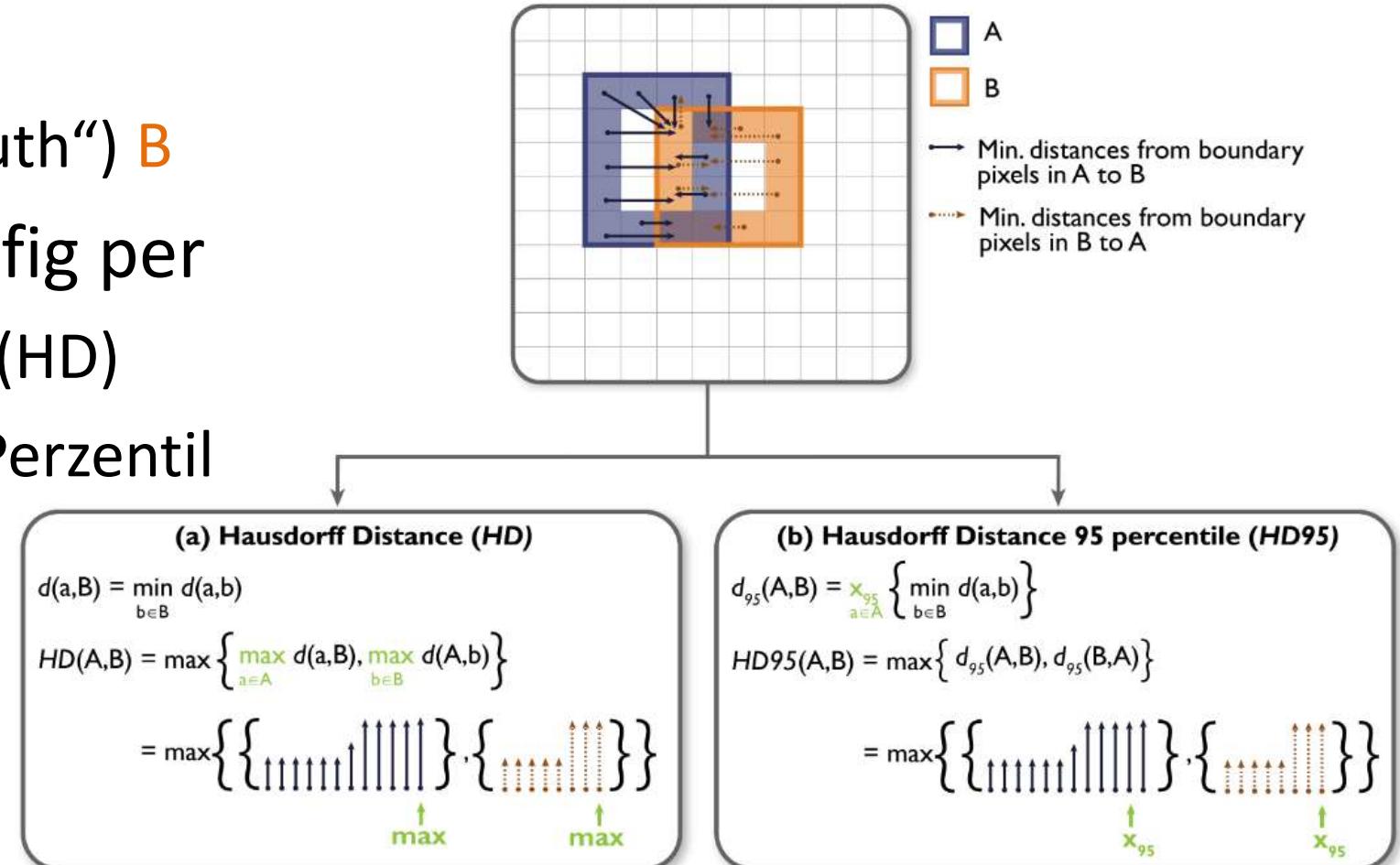
(b) IoU



$$\begin{aligned} \text{IoU}(A,B) &= \frac{\boxed{\text{A} \cap \text{B}}}{\boxed{\text{A}} + \boxed{\text{B}} - \boxed{\text{A} \cap \text{B}}} \\ &= \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \\ &= \frac{|A \cap B|}{|A \cup B|} \end{aligned}$$

# Evaluierung von Segmentierungen: Kontur-basiert

- Die maximale Abweichung der **Konturen** von
  - Segmentierung A und
  - Referenz („Ground Truth“) Bquantifiziert man häufig per
  - Hausdorff-Distanz (HD)
  - Robuster: HD mit Perzentil



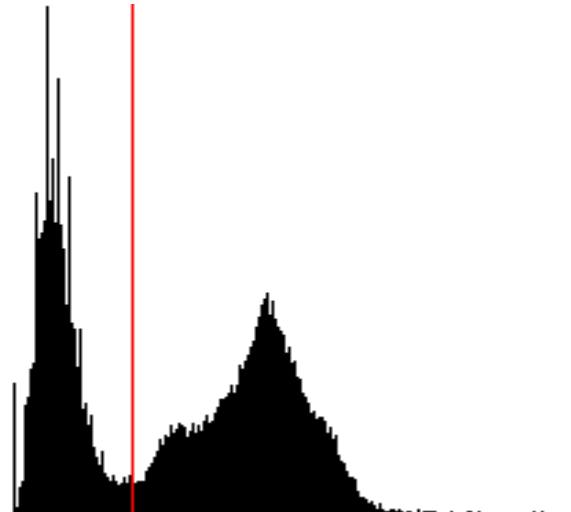
## **4.2 Grundlegende Verfahren**

# Schwellenwert-Segmentierung

- **Histogrammbasierte Segmentierung** klassifiziert die Pixel nur aufgrund ihrer individuellen Intensität
  - Anwendung von einem oder mehreren **Schwellenwerten**
  - **Täler im Histogramm** sind häufig sinnvolle Schwellenwerte



Bild  $g(\mathbf{x})$



Histogramm von  $g$  mit  
Schwellenwert  $\theta = 75$



Binarisiertes Bild  
 $g(\mathbf{x}) \geq \theta$

# Otsu-Verfahren zur Schwellenwert-Bestimmung

- **Idee von Otsu:** Optimaler Schwellenwert  $\theta$  sollte das Verhältnis  $\sigma_B^2 / \sigma_W^2$  der Varianz zwischen (*between*) und innerhalb (*within*) der Klassen maximieren
- **Berechnung** basiert auf Intensitäten  $g$ , Histogramm  $h(g)$ 
  - Zahl der Pixel in den beiden Klassen:

$$N_1 = \sum_{g=0}^{\theta-1} h(g), \quad N_2 = \sum_{g=\theta}^{g_{\max}} h(g), \quad N = N_1 + N_2$$

- Mittelwerte

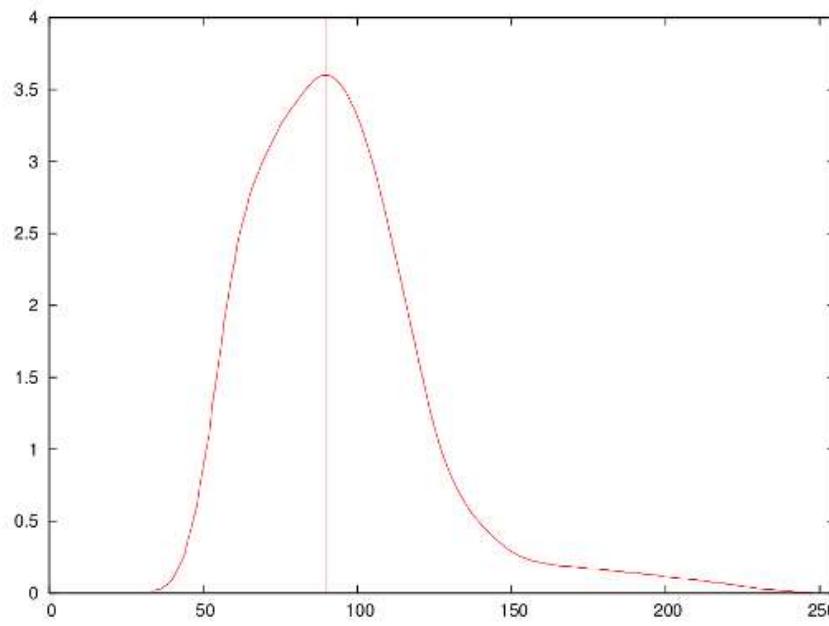
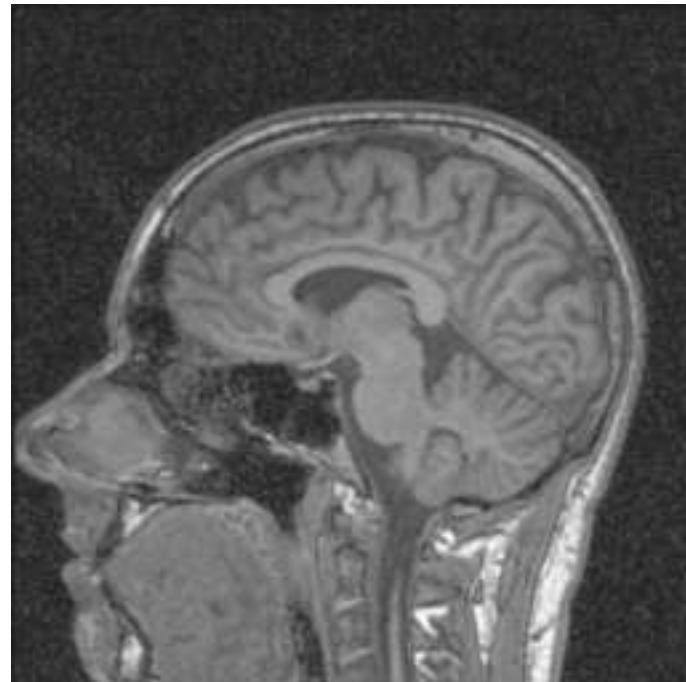
$$\mu_1 = \frac{1}{N_1} \sum_{g=0}^{\theta-1} g \cdot h(g), \quad \mu_2 = \frac{1}{N_2} \sum_{g=\theta}^{g_{\max}} g \cdot h(g), \quad \mu = \frac{N_1 \mu_1 + N_2 \mu_2}{N}$$

- Varianzen

$$\begin{aligned} \sigma_1^2 &= \frac{1}{N_1} \sum_{g=0}^{\theta-1} (g - \mu_1)^2 h(g), & \sigma_2^2 &= \frac{1}{N_2} \sum_{g=\theta}^{g_{\max}} (g - \mu_2)^2 h(g) \\ \sigma_B^2 &= \frac{N_1(\mu_1 - \mu)^2 + N_2(\mu_2 - \mu)^2}{N}, & \sigma_W^2 &= \frac{N_1 \sigma_1^2 + N_2 \sigma_2^2}{N} \end{aligned}$$

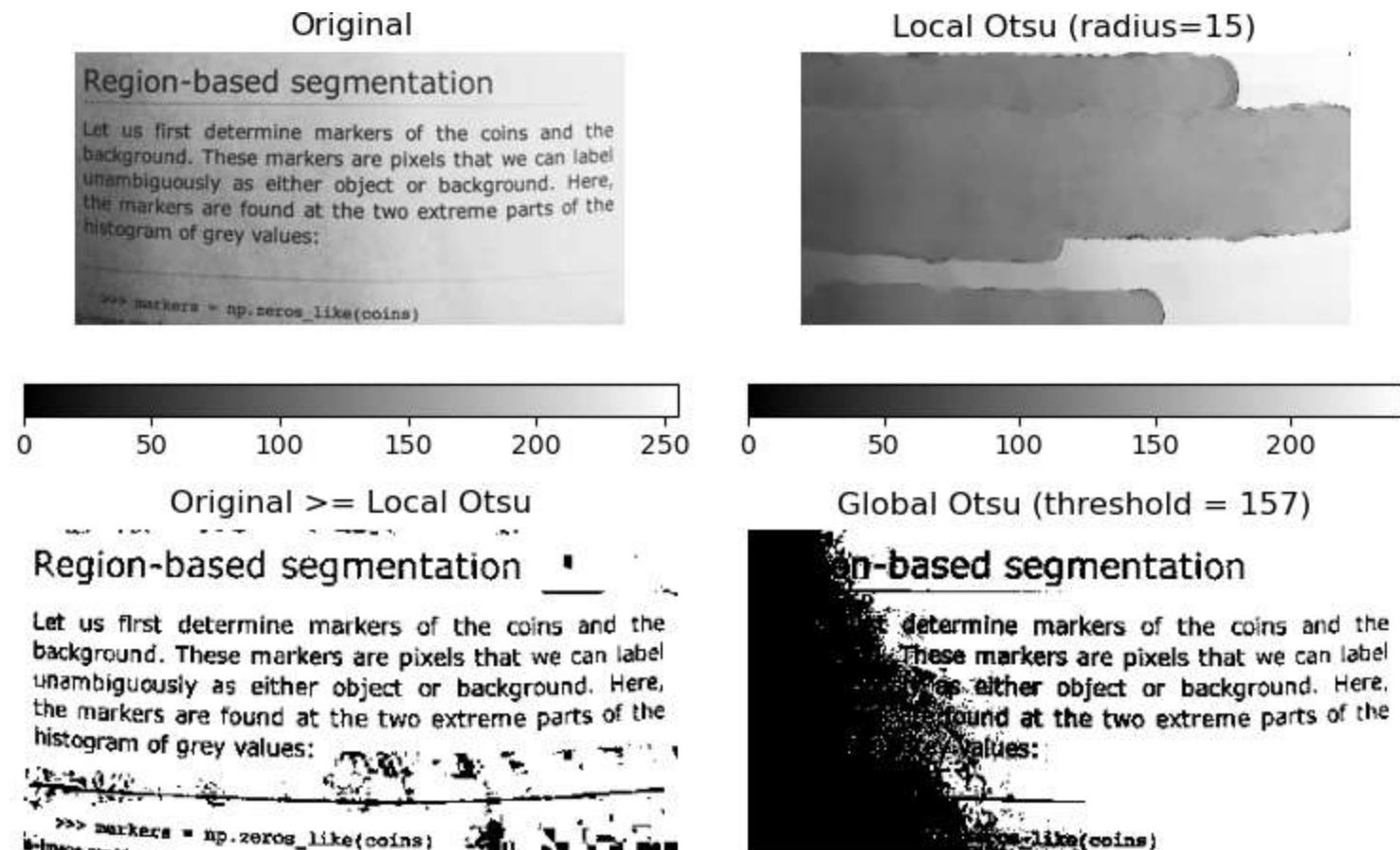
# Ergebnis des Otsu-Verfahrens

- Optimierung von  $\sigma_B^2/\sigma_W^2$  durch Ausprobieren aller  $\theta$  ergibt in unserem Beispielbild  $\theta = 90$



# Adaptive Schwellenwerte

- Bei ungleichmäßigen Hintergründen können **Schwellenwerte adaptiv** aufgrund des Histogramms lokaler Nachbarschaften bestimmt werden



# Vor- und Nachverarbeitung

- **Probleme der histogrammbasierten Segmentierung:**

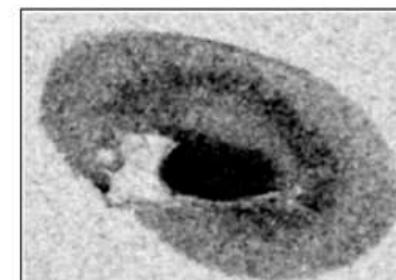
- Variationen um den Schwellenwert führen leicht zu kleinen Löchern oder Inseln
  - Rauschen kann Täler im Histogramm verwischen

- Häufige Schritte zur **Vor- und Nachverarbeitung:**

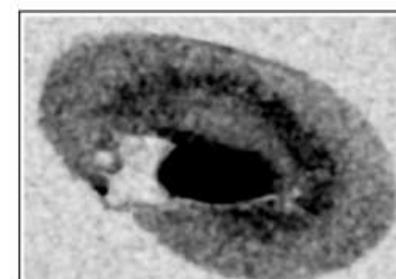
- **Glättung** des Bildes (s. Kapitel 1)
  - Nachbearbeitung der Segmentierungsmaske mit **morphologischen Operationen**
  - Analyse von **Zusammenhangskomponenten**



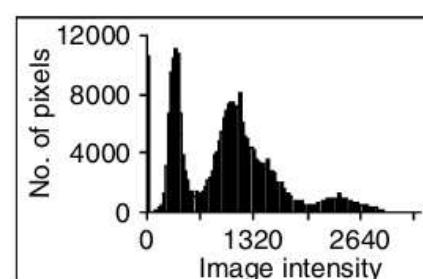
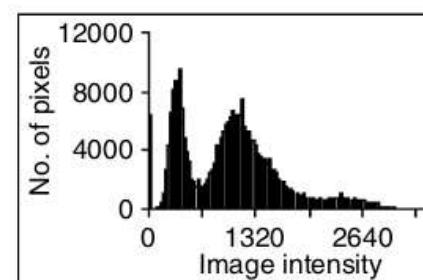
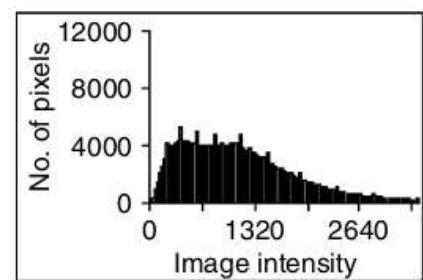
(a)



(b)



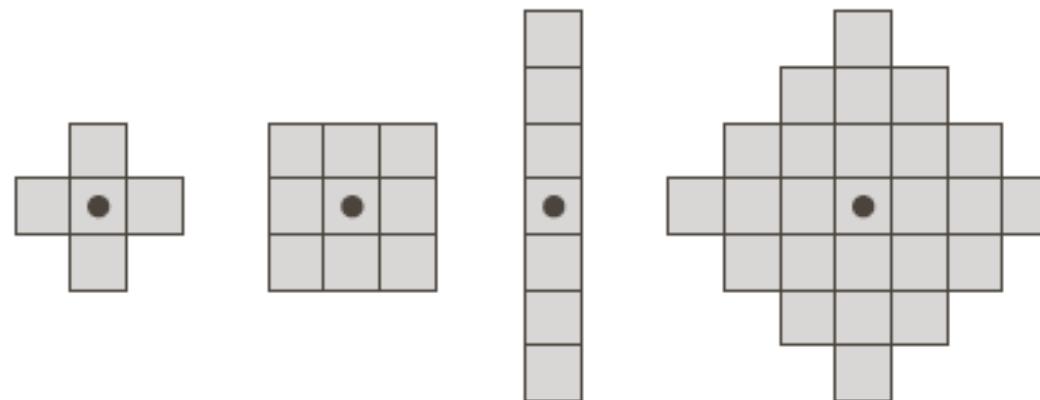
(c)



Medianfilterung:  $7 \times 7$  (b) bzw.  $9 \times 9$  (c) 12

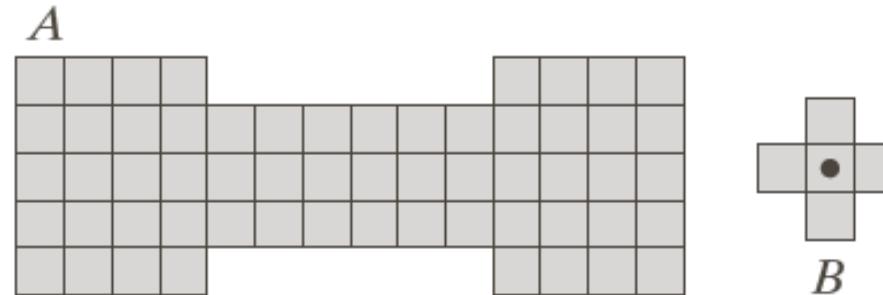
# Morphologische Bildverarbeitung

- **Morphologische Operationen**
  - Können auf Graustufen- oder Binärbilder angewandt werden (0=Hintergrund, 1=Vordergrund)
  - Nutzen ein **Strukturelement**, dessen Ankerpunkt – ähnlich dem Kern einer Kreuzkorrelation – auf alle Pixel verschoben wird
    - Form des Strukturelements kann an die relevanter Objekte angepasst werden
    - Grundoperationen ähneln der Median-Filterung, aber nutzen Minimum und Maximum

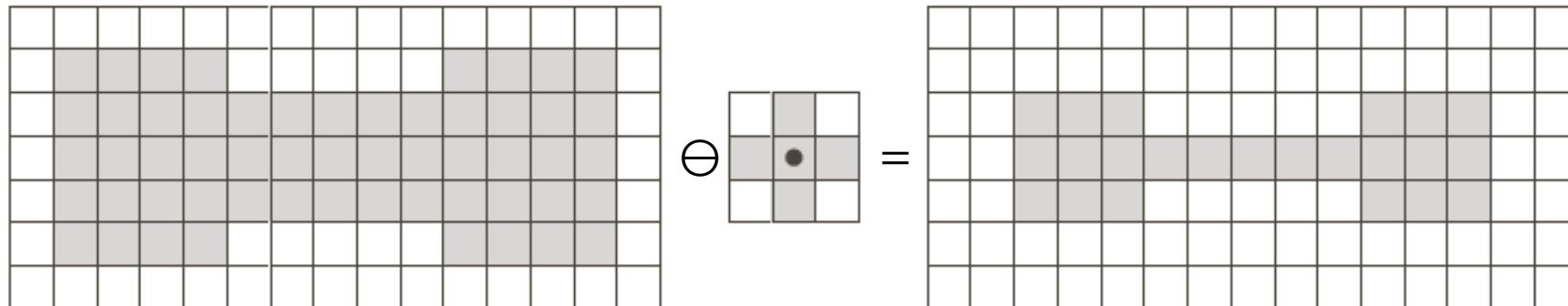


# Erosion

- *Gegeben:* Binärbild  $A$  und Strukturelement  $B$ :

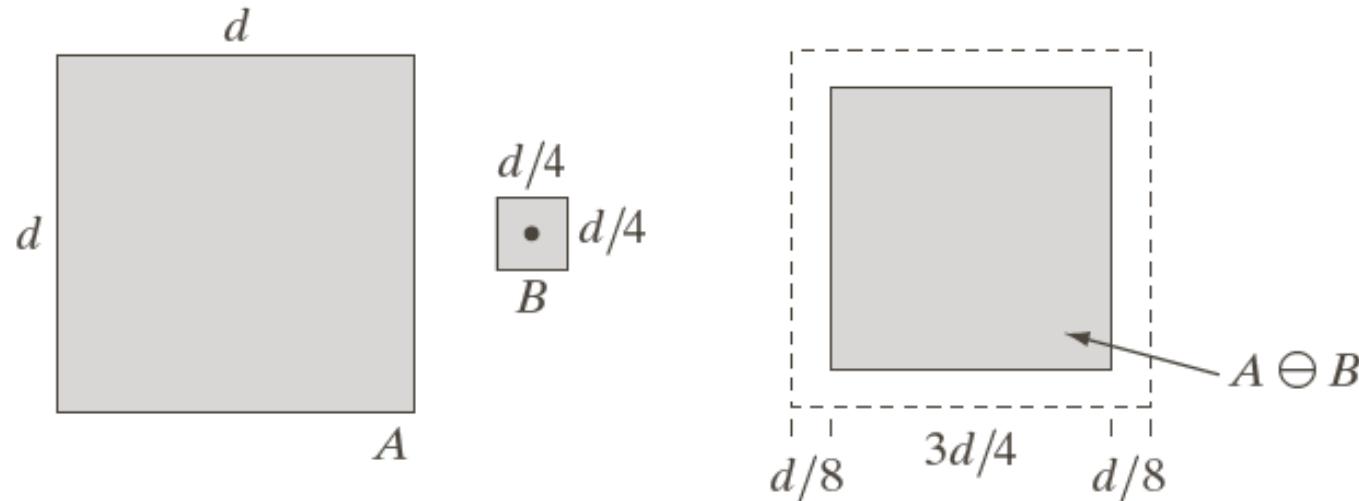


- **Erosion**  $\ominus$  entspricht Minimum-Filterung
  - Ausgabe-Pixel  $p$  ist genau dann Vordergrund, wenn das auf  $p$  zentrierte Strukturelement  $B$  vollständig im Vordergrund von  $A$  liegt
  - Am Rand wird  $A$  mit Null (Hintergrund) aufgefüllt

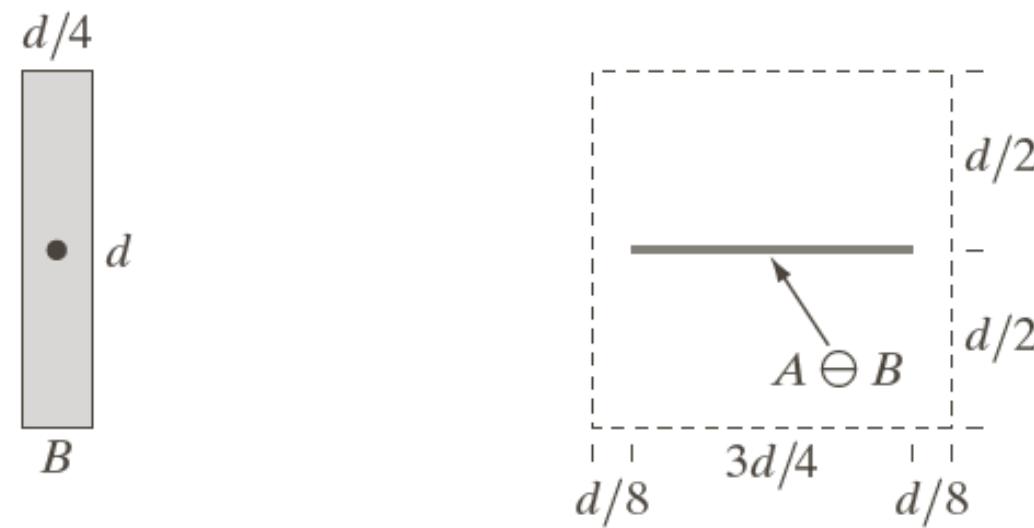


# Erosion: Einfluss des Struktur-Elements

Beispiel 1

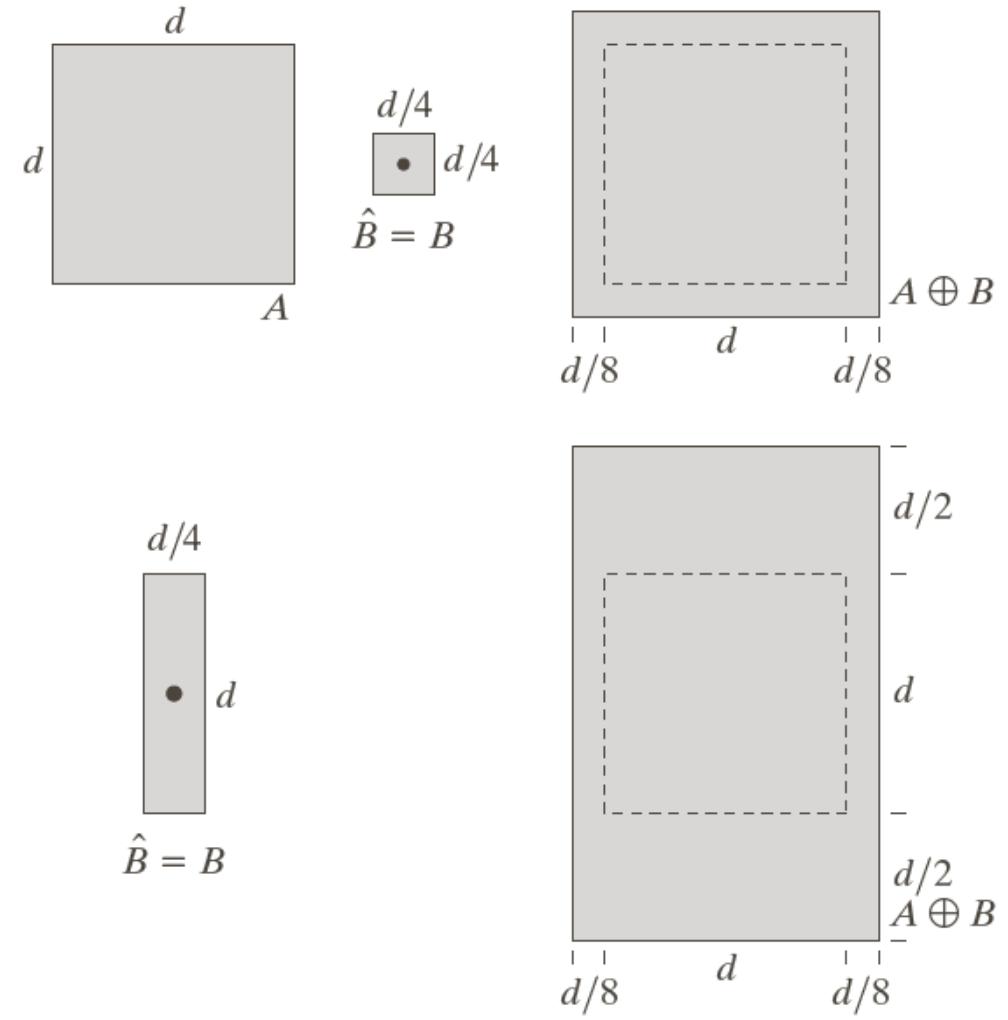


Beispiel 2



# Dilatation

- **Dilatation**  $\oplus$  entspricht Maximum-Filterung
  - Ausgabe-Pixel  $p$  ist genau dann Vordergrund, wenn das auf  $p$  zentrierte Strukturelement  $B$  mit mindestens einem Vordergrund-Pixel von  $A$  überlappt



# Morphologisches Öffnen und Schließen

- Definition des **morphologischen Öffnens** (Opening)

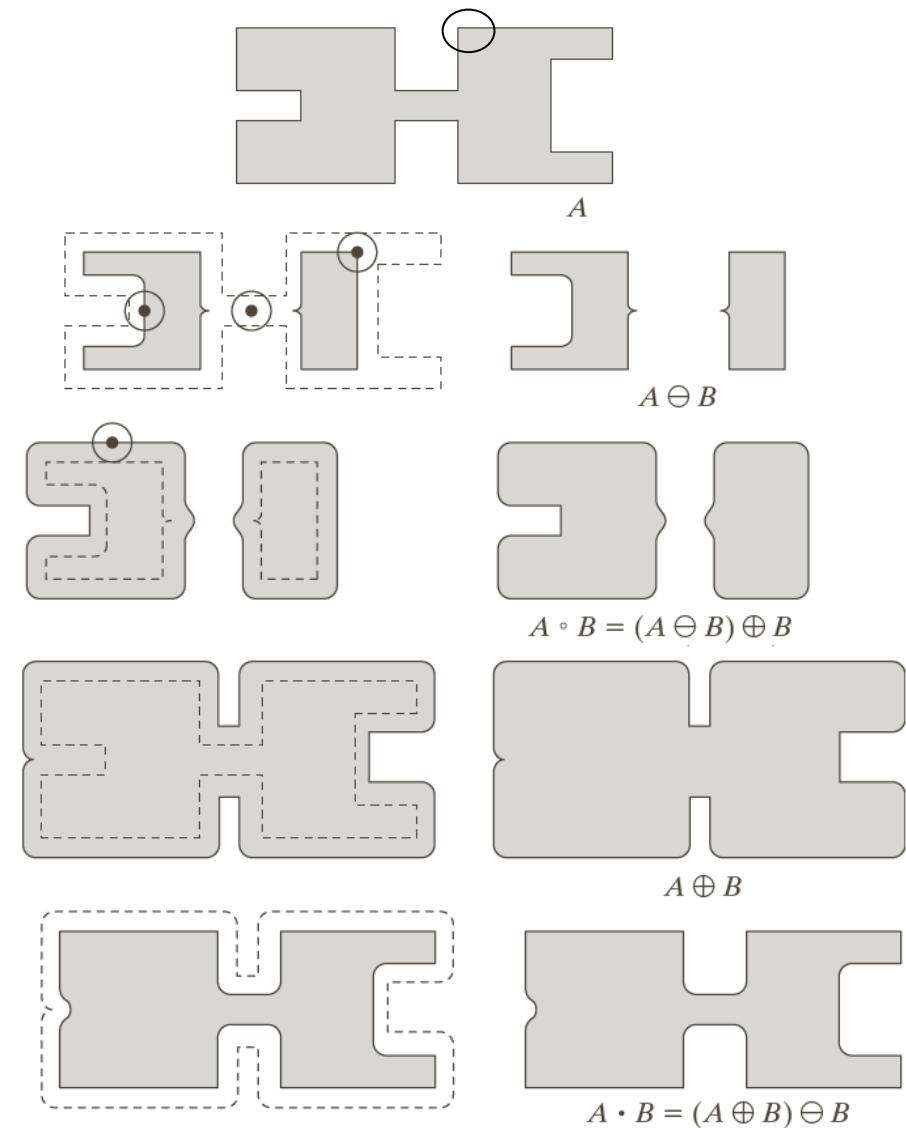
$$A \circ B = (A \ominus B) \oplus B$$

- Beseitigt schmale Vorsprünge / Verbindungen
- Glättet Objektränder

- Definition des **morphologischen Schließens** (Closing)

$$A \bullet B = (A \oplus B) \ominus B$$

- Beseitigt kleine Löcher
- Füllt schmale Lücken und enge Einbuchtungen auf



# Zusammenhangskomponenten

- Die Definition von **Zusammenhangskomponenten** in Binärbildern entspricht der aus der Graphentheorie
  - Vordergrund-Pixel als Knoten, Kanten verbinden benachbarte Pixel
  - Reicht in manchen Fällen aus um Objekte zu trennen
  - Leider verhindern oft wenige Pixel erwünschte Trennung / Verbindung
  - Beispiel:* Versuch der Hirnsegmentierung als Zusammenhangskomponente nach Otsu-Schwellenwert und morphologischer Öffnung:



Binarisiertes Bild



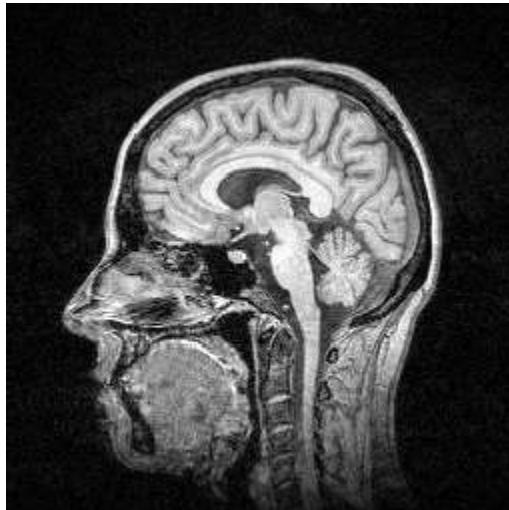
Opening mit Kreis (11x11)



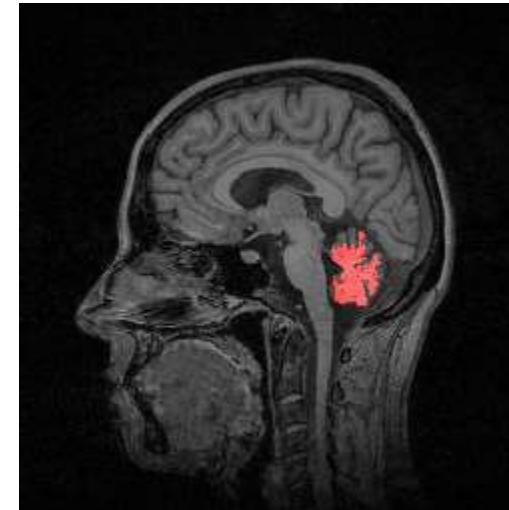
Zusammenhangskomponenten

# Region Growing

- **Region Growing** fügt der Segmentierung von einem Startpunkt ausgehend so lange benachbarte Pixel hinzu, wie ein Homogenitätskriterium erfüllt ist
  - Basiert meist auf Intensitätsunterschieden (z.B. bezüglich des Startpunkts, des Nachbarpixels, des aktuellen Mittelwerts der Region)



Bild



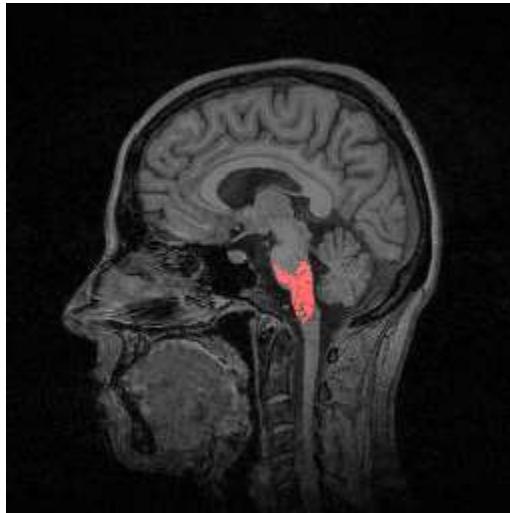
Startpunkt im Kleinhirn,  
Schwelle=20



Startpunkt im Hirnstamm,  
Schwelle=17

# Probleme des Region Growing

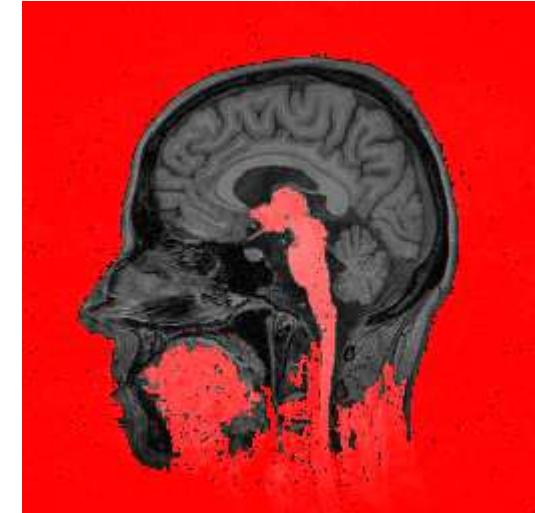
- Übliche Probleme des region growing sind
  - Plötzliches „Auslaufen“ oberhalb eines Schwellenwerts
  - Auslassen einzelner Pixel aufgrund von Bildrauschen
- Lösungsansätze ähnlich wie bei Schwellenwert-Segmentierung



Startpunkt im Hirnstamm,  
Schwelle=12



Startpunkt im Hirnstamm,  
Schwelle=17

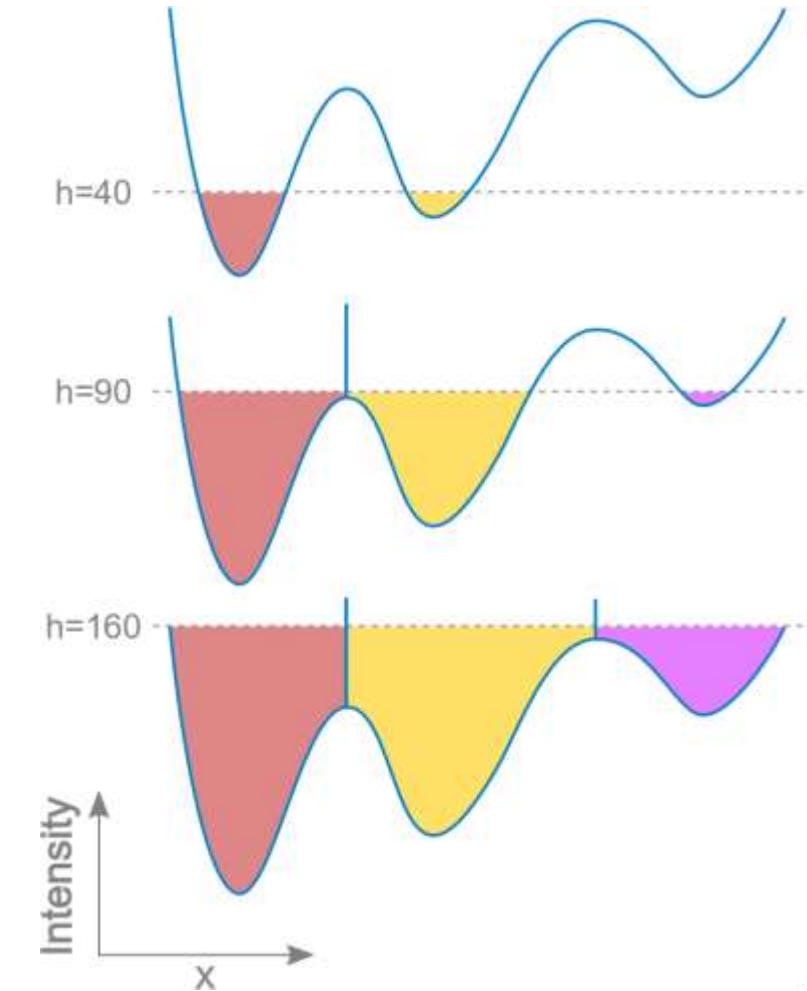


Startpunkt im Hirnstamm,  
Schwelle=18

# Wasserscheidentransformation

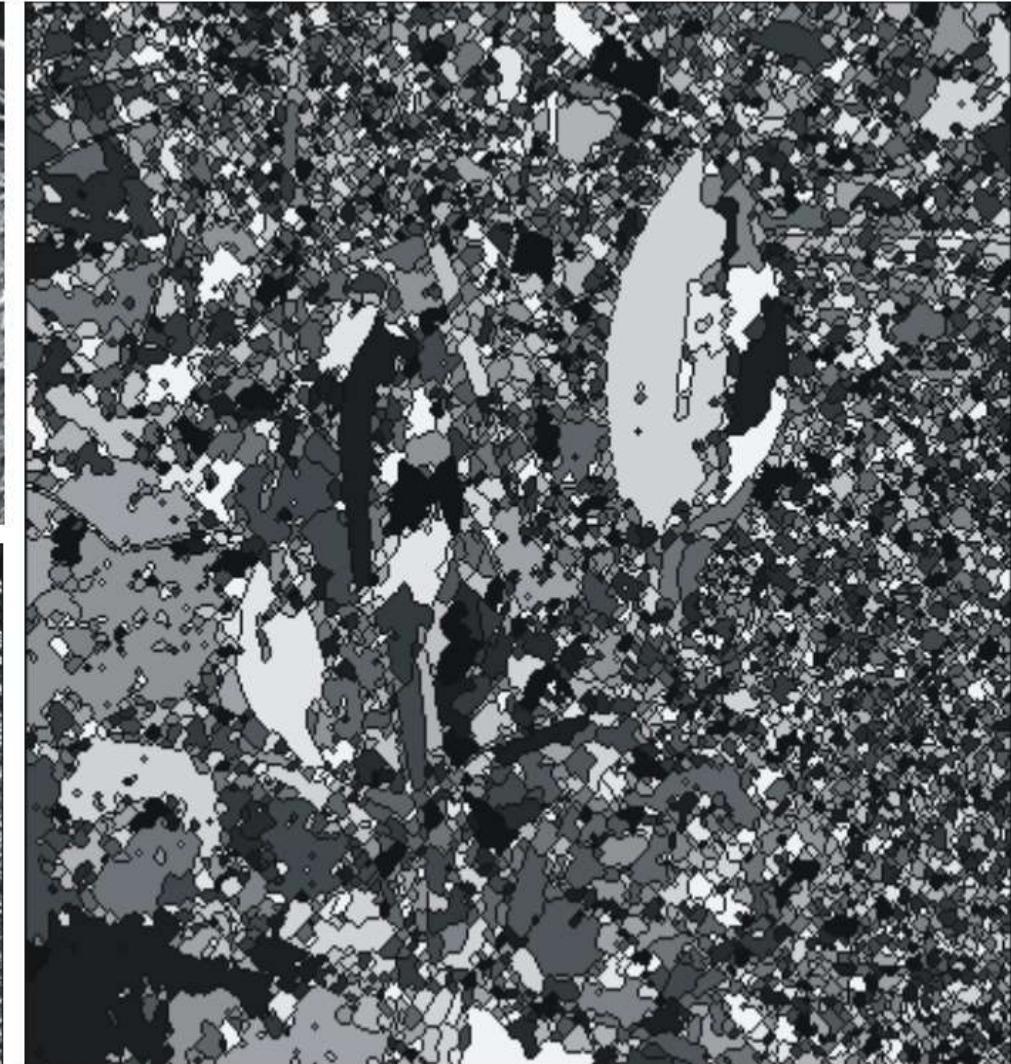
Die **Wasserscheidentransformation** (engl. watershed transformation) fasst alle Punkte im Bild zusammen, von denen aus ein Gradientenabstieg im selben Minimum endet

- Wenn wir Intensitäten als Höhenfeld auffassen, trennen **Wasserscheiden** im geografischen Sinne diese Gebiete
- *Häufige Vorstellung:* Steigendes Wasserniveau im Gebirge, „Dämme“ verhindern Zusammenfließen verschiedener Staubecken



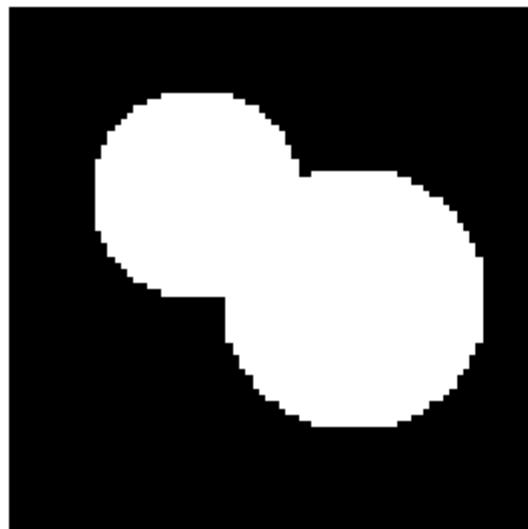
# Bildsegmentierung per Wasserscheidentransformation

- Wendet man die **Wasserscheiden-Transformation** auf die **Gradientenstärke** an, erhält man Wasserscheiden an Kanten
- Führt meist zu einer **Übersegmentierung** des Bildes. Ansätze:
  - Zusammenfassen von Regionen
  - Quellen durch Marker vorgeben

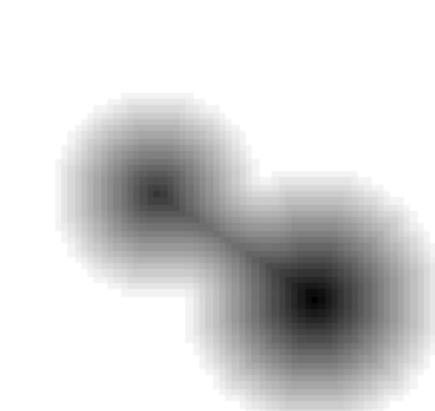


# Trennung von Objekten: Distanztransformation

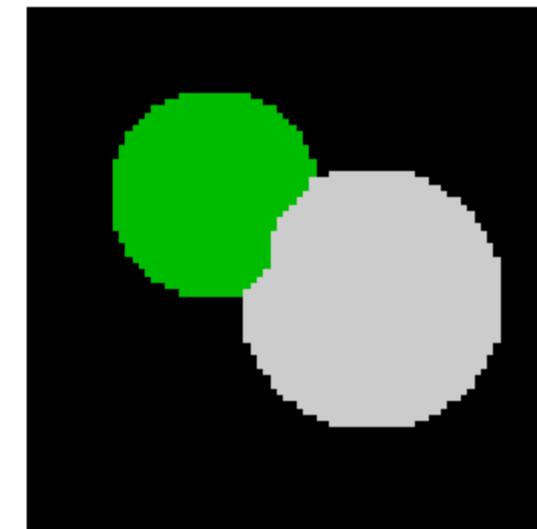
- Die **Distanztransformation** weist jedem Pixel einer binären Maske seinen Abstand vom Hintergrund zu
- Die **Wasserscheidentransformation** der Distanztransformation eignet sich dafür, sich berührende Objekte in einer Segmentierungsmaske zu trennen



Ursprüngliche Maske



Distanztransformation



Getrennte Objekte

# Zusammenfassung: Einfache Segmentierungsverfahren

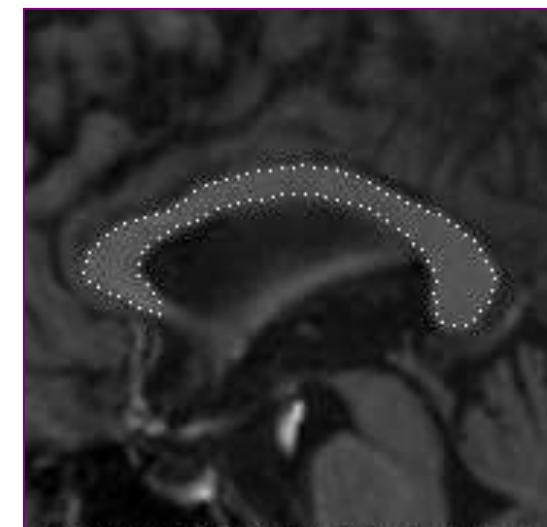
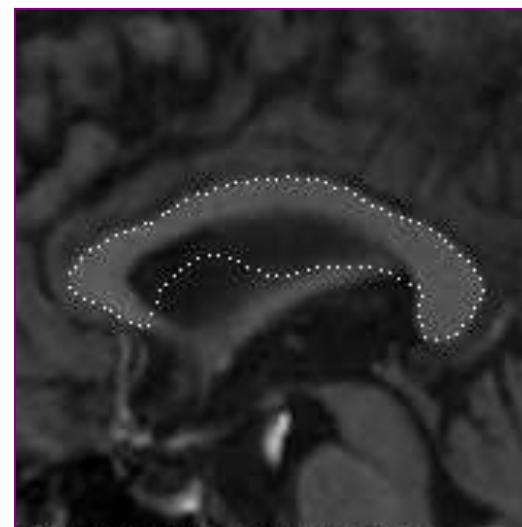
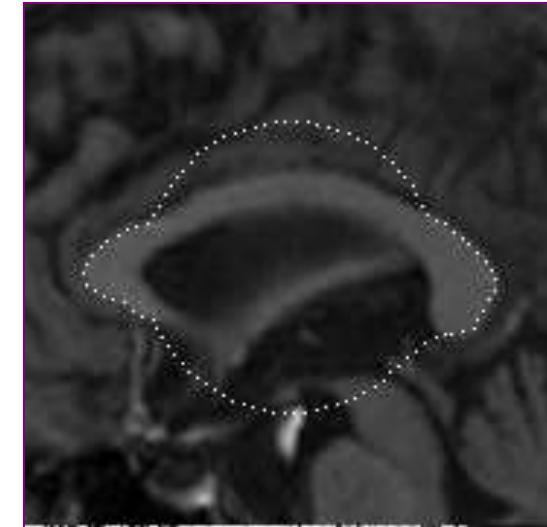
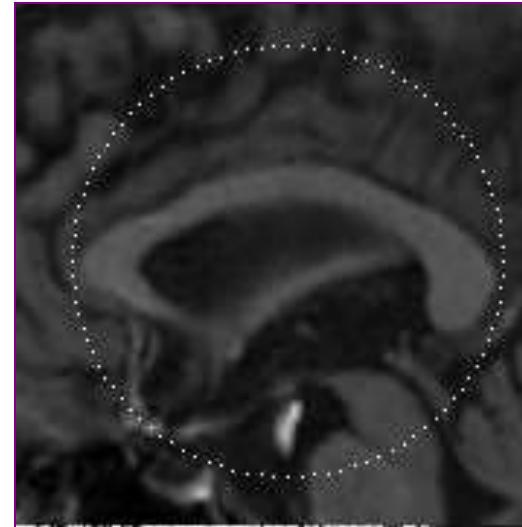
Grundlegende Segmentierungsverfahren sind:

- Segmentierung per **Schwellenwert**
  - Häufig per **Histogramm** bestimmt (*Beispiel*: Otsu)
  - Bei ungleichmäßigen Hintergründen **adaptiv** (lokale Nachbarschaft)
- **Region Growing** (flood fill)
- **Wasserscheidentransformation**
  - Anwendung meist auf **Gradientenbilder** oder **Distanztransformation**
- Vor- und Nachverarbeitung per Glättung, **morphologischen Operationen** und **Zusammenhangskomponenten**

## **4.3 Deformierbare Modelle**

# Grundidee: Segmentierung mit Aktiven Konturen

- **Aktive Konturen** werden im Bild initialisiert und verformen sich mit dem Ziel eine Energie zu minimieren, die meist aus zwei Teilen besteht:
  - **Externe Energie** verbindet die Kontur mit Bildinhalten, zieht sie z.B. in Richtung von Bildkanten
  - **Interne Energie** bewertet die Plausibilität der Kontur an sich, z.B. glatt und nicht zu lang
- *Beispiel:* Segmentierung des Balkens im Gehirn mittels einer aktiven Kontur



Bilder aus [Davatzikos et al. 1996]

# Explizite Deformierbare Modelle

- **Deformierbare Modelle** in 2D lassen sich explizit als Kurven mit Parameter  $s \in [0,1]$  schreiben:

$$\mathbf{v}(s) = \begin{pmatrix} x(s) \\ y(s) \end{pmatrix}$$

- **Aktive Konturen** optimieren  $\mathbf{v}(s)$  im Hinblick auf eine Energie, die aus einer externen (Bildterm) und einer internen Energie (Glattheitsterm) besteht:

$$E = E_{\text{ext}} + E_{\text{int}}$$

- Betrachtet man den Prozess der iterativen Energieminimierung als Animation, kriechen die Kurven wie Schlangen über das Bild. Aktive Konturen werden daher auch als „**Snakes**“ bezeichnet.

# Externe Energie

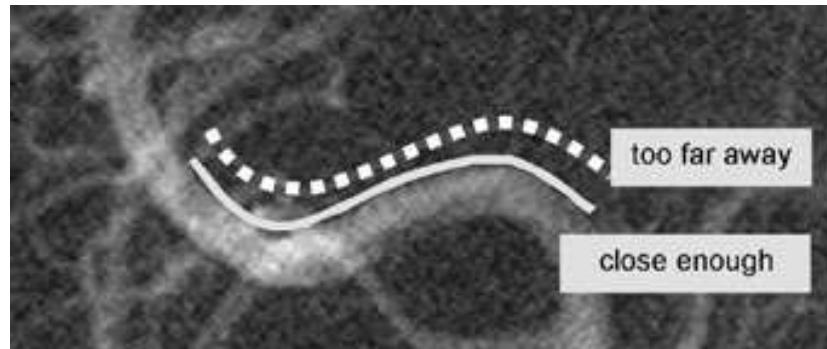
- **Externe Energie**  $E_{\text{ext}} = \frac{1}{2} \int_0^1 P(\mathbf{v}(s)) ds$  zieht die Kontur in Richtung der gewünschten Bildstrukturen. *Beispiele* für Potentialfunktionen  $P(\mathbf{x})$ :
  - Suche nach einer bestimmten **Bildintensität**:

$$P(\mathbf{v}(s)) = (I(\mathbf{v}(s)) - I_{\text{target}})^2$$

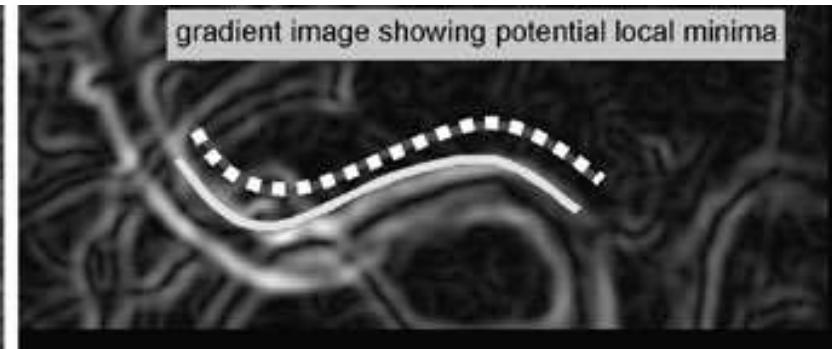
- Suche nach **Bildkanten**:

$$P(\mathbf{v}(s)) = -\|\nabla I(\mathbf{v}(s))\|^2$$

- Ableitungen erfordern in der Regel eine Glättung (s. Kapitel 1)



Bild



Bildgradient

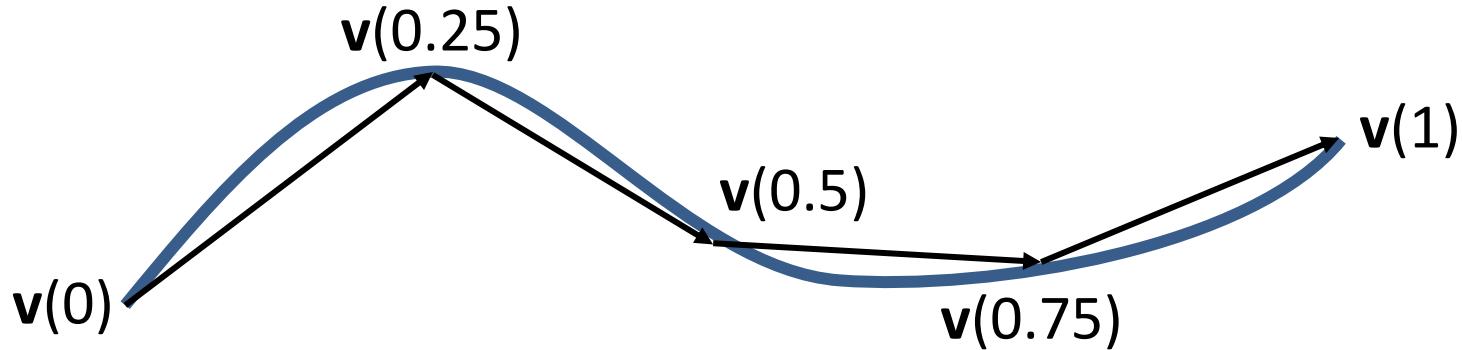
# Interne Energie

- **Interne Energie** beruht bei [Kass et al. 1998] auf den ersten und zweiten Ableitungen von  $\mathbf{v}(s)$ :

$$E_{\text{int}} = \frac{1}{2} \int_0^1 \left[ w_1(s) \left\| \frac{d\mathbf{v}(s)}{ds} \right\|^2 + w_2(s) \left\| \frac{d^2\mathbf{v}(s)}{ds^2} \right\|^2 \right] ds$$

- **Erster Term** macht die Kurve *kürzer*
  - Optimum wenn  $\mathbf{v}(0) \neq \mathbf{v}(1)$  fest sind: Gerade Verbindung
  - Optimum mit Randbedingung  $\mathbf{v}(0) = \mathbf{v}(1)$ : Kurve schrumpft auf einen Punkt
- **Zweiter Term** macht die Kurve *glatter*
- Gewichte  $w_1$  und  $w_2$  bestimmen den jeweiligen Einfluss
  - $w_2(s)=0$  ermöglicht an der Stelle  $s$  eine scharfe Ecke

# Anschauung: Länge einer Parametrischen Kurve



- Approximation der Länge durch  $n$  Liniensegmente:

$$l \approx \sum_{i=0}^{n-1} \left\| \mathbf{v}\left(\frac{i+1}{n}\right) - \mathbf{v}\left(\frac{i}{n}\right) \right\| = \sum_{i=0}^{n-1} \left\| \frac{\mathbf{v}\left(\frac{i}{n} + \Delta s\right) - \mathbf{v}\left(\frac{i}{n}\right)}{\Delta s} \right\| \Delta s$$

- Im Grenzfall  $n \rightarrow \infty$  erhalten wir die genaue Länge:

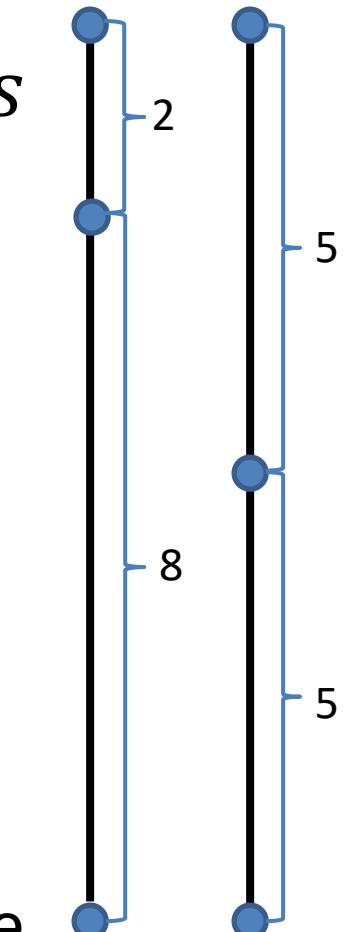
$$l = \int_0^1 \left\| \frac{d\mathbf{v}(s)}{ds} \right\| ds$$

# Anschauung: Was ist mit dem Quadrat?

- Statt  $\int_0^1 \left\| \frac{d\mathbf{v}(s)}{ds} \right\| ds$  minimiert die Snake  $\int_0^1 \left\| \frac{d\mathbf{v}(s)}{ds} \right\|^2 ds$ 
  - Das Quadrat vereinfacht spätere Rechnungen
- Aufgrund der Cauchy-Schwarz-Ungleichung gilt

$$\left( \int_0^1 \left\| \frac{d\mathbf{v}(s)}{ds} \right\| ds \right)^2 \leq \int_0^1 \left\| \frac{d\mathbf{v}(s)}{ds} \right\|^2 ds$$

- Die Snake minimiert eine obere Schranke der Länge
- Zusätzlich bevorzugt sie gleichmäßig verteilte Stützpunkte
- Bei perfekter Verteilung erhalten wir das Quadrat der Länge



$$\begin{array}{ll} 2 + 8 = 10 & 5 + 5 = 10 \\ 2^2 + 8^2 = 68 & 5^2 + 5^2 = 50 \\ 31 & \end{array}$$

# Anschauung: Krümmung

- **Krümmung**  $\kappa$  gibt die lokale Abweichung der Kurve von einer Geraden an
  - Ihre Definition basiert auf der Ableitung des Einheitstangentenvektors

$$\mathbf{T}(s) = \frac{\frac{d\mathbf{v}(s)}{ds}}{\left\| \frac{d\mathbf{v}(s)}{ds} \right\|}$$



- Im Falle einer *Parametrisierung nach Bogenlänge* (d.h.  $\left\| \frac{d\mathbf{v}(s)}{ds} \right\| = 1$ ) ist
$$\kappa = \left\| \frac{d^2\mathbf{v}(s)}{ds^2} \right\|$$
- Wir dürfen diese Formel verwenden, wenn wir die Stützpunkte ungefähr in festen Abständen verteilen (gleichmäßig, weder zu dicht noch zu weit)

# Illustration: Interne Energie



Wenig dehnfähig (großes  $w_1$ )



Dehnfähiger, aber steif  
(kleineres  $w_1$ , großes  $w_2$ )



Dehnfähig und biegsam  
(kleines  $w_1$ , kleines  $w_2$ )

# Variationsrechnung

- Differentialrechnung:
  - Betrachtet **Funktionen**  $f(x)$  die reelle Zahlen auf reelle Zahlen abbilden
  - **Minima** sind Punkte  $x$ , für die für alle hinreichend kleinen  $\Delta x$  gilt:  
$$f(x + \Delta x) > f(x)$$
  - **Notwendige Bedingung:**  
$$\frac{df(x)}{dx} = 0$$
- Variationsrechnung:
  - Betrachtet **Funktionale**  $F(f)$ , die Funktionen auf reelle Zahlen abbilden
  - **Minima** sind Funktionen  $f(x)$ , für die für alle hinreichend kleinen  $\varepsilon$  und jede beliebig oft differenzierbare Testfunktion  $\eta(x)$  gilt:  
$$F(f + \varepsilon \cdot \eta) > F(f)$$
  - Notwendige Bedingung: **Euler-Lagrange-Gleichung**

# Euler-Lagrange-Gleichung

- **Satz** aus der Variationsrechnung:

Eine notwendige Bedingung für die Minimierung eines Funktionals, das sich mittels einer Lagrange-Funktion  $L$  in der Form

$$F(f) = \int_a^b L(x, f(x), f'(x), f''(x)) dx$$

darstellen lässt, ist die Erfüllung der Euler-Lagrange-Gleichung

$$\frac{\partial L}{\partial f} - \frac{\partial}{\partial x} \left( \frac{\partial L}{\partial f'} \right) + \frac{\partial^2}{\partial x^2} \left( \frac{\partial L}{\partial f''} \right) = 0$$

# Kurze Erinnerung: Taylor-Entwicklung

- Die **Taylor-Entwicklung** ermöglicht die Approximation glatter Funktionen in der Umgebung einer Stelle  $x_0$  durch Polynome

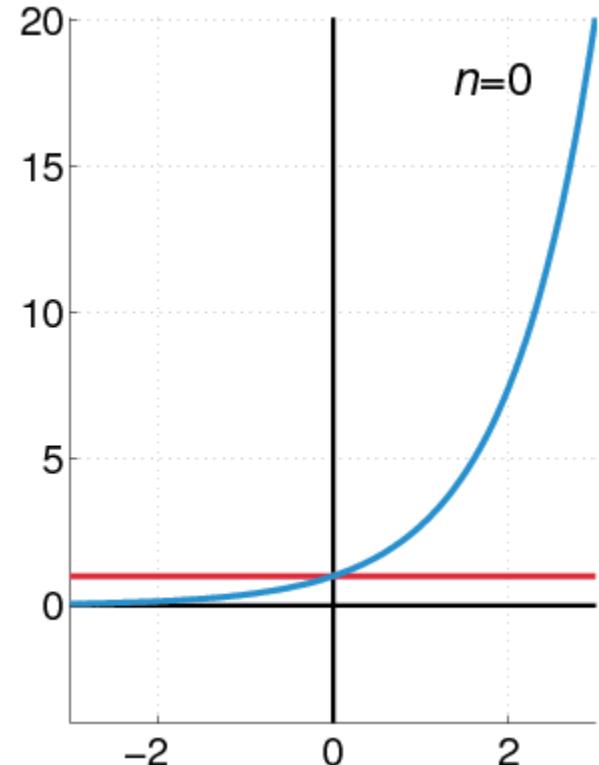
$$f(x) \approx f(x_0)$$

$$+ \frac{d}{dx} f(x_0)(x - x_0)$$

$$+ \frac{1}{2} \frac{d^2}{dx^2} f(x_0)(x - x_0)^2 + \dots$$

$$+ \frac{1}{n!} \frac{d^n}{dx^n} f(x_0)(x - x_0)^n$$

$$+ O((x - x_0)^{n+1})$$



Bildquelle: Wikipedia

# Herleitung der Euler-Lagrange-Gleichung, Teil 1

- Betrachte  $F(f) = \int_a^b L(x, f(x), f'(x)) dx$ 
  - Zur Vereinfachung der Notation schreiben wir ab jetzt  $L(x, f, f')$
- **Notwendige Bedingung** für Extrema: Für jede Testfunktion  $\eta(x)$  ist

$$\lim_{\epsilon \rightarrow 0} \frac{1}{\epsilon} \int_a^b L(x, f + \epsilon\eta, f' + \epsilon\eta') - L(x, f, f') dx = 0$$

- Einsetzen der Taylor-Entwicklung von  $L$ :

$$\begin{aligned} \lim_{\epsilon \rightarrow 0} \frac{1}{\epsilon} \int_a^b L(x, f, f') + \frac{\partial L}{\partial f} \epsilon\eta + \frac{\partial L}{\partial f'} \epsilon\eta' + O(\epsilon^2) - L(x, f, f') dx &= 0 \\ \Rightarrow \int_a^b \frac{\partial L}{\partial f} \eta + \frac{\partial L}{\partial f'} \eta' dx &= 0 \end{aligned}$$

# Herleitung der Euler-Lagrange-Gleichung, Teil 2

- Notwendige Bedingung (aus Teil 1):  $\int_a^b \frac{\partial L}{\partial f} \eta + \frac{\partial L}{\partial f'} \eta' dx = 0$
- Partielle Integration von  $\int_a^b \frac{\partial L}{\partial f'} \eta' dx$  mit  $u = \frac{\partial L}{\partial f'}, dv = \eta'$ :  
$$\left[ \frac{\partial L}{\partial f'} \eta \right]_a^b - \int_a^b \frac{\partial}{\partial x} \left( \frac{\partial L}{\partial f'} \right) \eta \, dx$$
- Fordern wir zur Einhaltung der Randbedingungen  $\eta(a) = \eta(b) = 0$ , wird die notwendige Bedingung zu

$$\int_a^b \frac{\partial L}{\partial f} \eta - \frac{\partial}{\partial x} \left( \frac{\partial L}{\partial f'} \right) \eta \, dx = \underbrace{\int_a^b \left( \frac{\partial L}{\partial f} - \frac{\partial}{\partial x} \left( \frac{\partial L}{\partial f'} \right) \right) \eta \, dx}_{=} = 0$$

Um für beliebige  $\eta$  Null zu erhalten muss dieser Teil Null sein -> Euler-Lagrange

# Anwendung auf das Snake-Modell

Dritter Term der ELG analog (wiederholte partielle Integration):

$$\frac{\partial L}{\partial f} - \frac{\partial}{\partial x} \left( \frac{\partial L}{\partial f'} \right) + \frac{\partial^2}{\partial x^2} \left( \frac{\partial L}{\partial f''} \right) = 0$$

Anwendung der ELG auf das Snake-Modell

$$E = \frac{1}{2} \int_0^1 P(\mathbf{v}(s)) ds + \frac{1}{2} \int_0^1 \left[ w_1(s) \left\| \frac{d\mathbf{v}(s)}{ds} \right\|^2 + w_2(s) \left\| \frac{d^2\mathbf{v}(s)}{ds^2} \right\|^2 \right] ds$$

mit externem Potential  $P$  ergibt

$$\frac{1}{2} \nabla P(\mathbf{v}(s)) - \frac{d}{ds} \left( w_1(s) \frac{d\mathbf{v}(s)}{ds} \right) + \frac{d^2}{ds^2} \left( w_2(s) \frac{d^2\mathbf{v}(s)}{ds^2} \right) = \mathbf{0}$$

Da  $\mathbf{v}(s)$  vektorwertig ist, ergibt sich ein **System zweier Gleichungen** (in 2D-Bildern) für die  $x$ - bzw.  $y$ -Koordinaten unserer Kurve

# Energieminimierung im Snake-Modell

- Die linke Seite der Euler-Lagrange-Gleichung des Snake-Modells

$$\frac{1}{2} \nabla P(\mathbf{v}(s)) - \frac{d}{ds} \left( w_1(s) \frac{d\mathbf{v}(s)}{ds} \right) + \frac{d^2}{ds^2} \left( w_2(s) \frac{d^2\mathbf{v}(s)}{ds^2} \right) = \mathbf{0}$$

gibt eine lokale Änderung an  $\mathbf{v}(s)$  an, die  $E$  so steil wie möglich ansteigen lässt.

- Wir nutzen sie zum **Gradientenabstieg**:

- Einführung eines (künstlichen) Zeitparameters  $\mathbf{v}(s, t)$
- Ersetzen der rechten Seite durch  $-\partial_t \mathbf{v}(s, t)$
- Diskretisierung der Zeit in uniforme Schritte
- Diskretisierung der Kurve  $\mathbf{v}(s, t)$  als Polygonzug
- Iterative Anwendung der Updates auf  $\mathbf{v}(s, t)$  bis  $\partial_t \mathbf{v}(s, t) \approx 0$

# Interpretation als Summe von Kräften

Die Ableitung nach der Zeit in

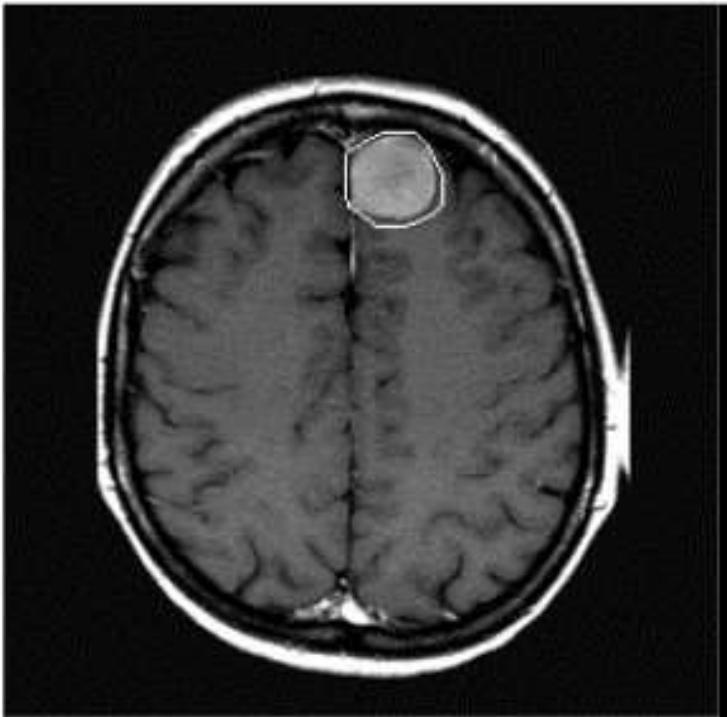
$$\frac{1}{2} \nabla P(\mathbf{v}(s, t)) - \frac{\partial}{\partial s} \left( w_1(s) \frac{\partial \mathbf{v}(s, t)}{\partial s} \right) + \frac{\partial^2}{\partial s^2} \left( w_2(s) \frac{\partial^2 \mathbf{v}(s, t)}{\partial s^2} \right) = - \frac{\partial \mathbf{v}(s, t)}{\partial t}$$

durch finite Differenzen zu ersetzen ergibt

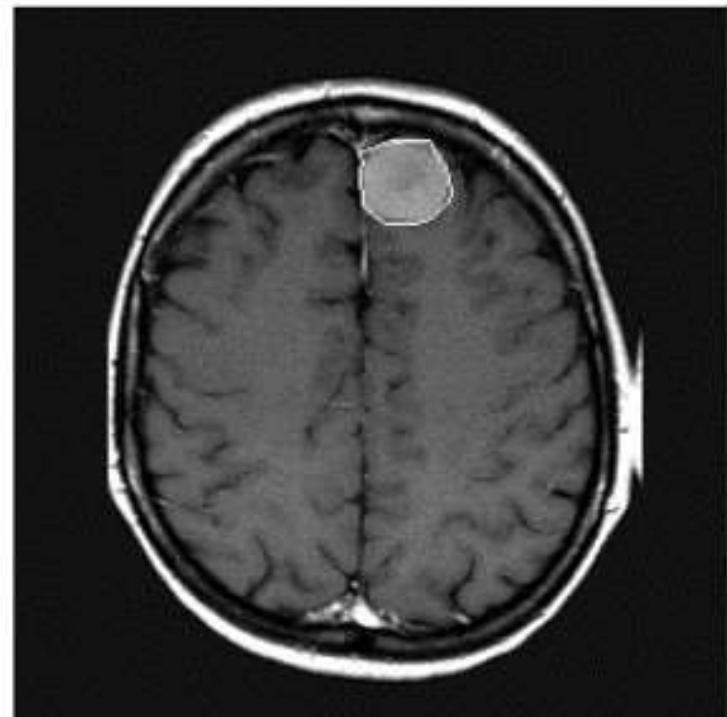
$$\text{Linke Seite} = - \frac{\mathbf{v}(s, t + \Delta t) - \mathbf{v}(s, t)}{\Delta t}$$
$$\Rightarrow \mathbf{v}(s, t + \Delta t) = \mathbf{v}(s, t) + \Delta t \times (-\text{L. S.})$$

- Aus jedem Energieterm wird eine Kraft, iterative Updates verschieben  $\mathbf{v}(s, t)$  aufgrund der Summe dieser Kräfte mit Schrittweite  $\Delta t$
- Mit geeigneter Schrittgröße konvergieren die Updates zu einer Lösung der Euler-Lagrange-Gleichung, am Optimum gleichen die Kräfte einander aus

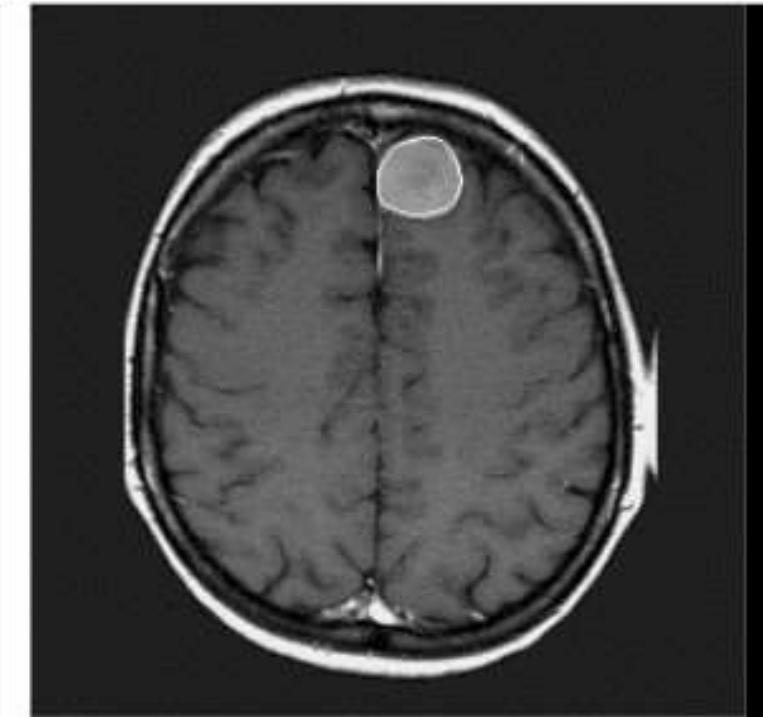
# Beispiel: Tumorsegmentierung per Snake



Initialkontur



2. Iterationsschritt

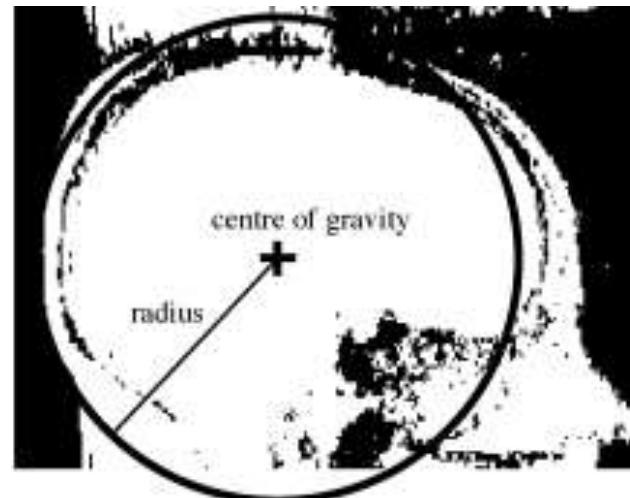
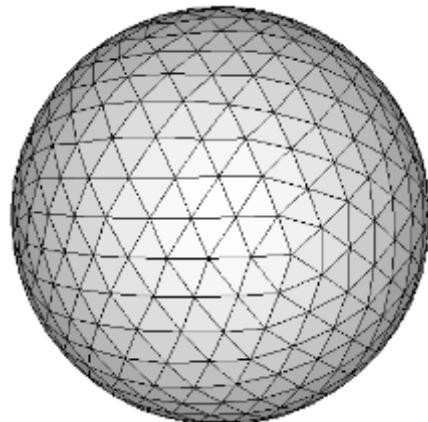


Konvergenz

# Fallbeispiel: Brain Extraction Tool (BET)



- **Brain Extraction Tool (BET)** aus FSL, <http://fsl.fmrib.ox.ac.uk/>
  - Bis heute sehr aktiv genutzt (>12000 Zitationen auf google scholar)
  - Fast vollständig automatisiert
    - Zwei Parameter für 1) Über-/Untersegmentierung 2) Intensitätsgradienten
  - Basiert auf einem deformierbaren Modell
    1. Explizite Repräsentation als Dreiecksnetz
    2. Heuristische Initialisierung: Kleine Kugel innerhalb des Gehirns

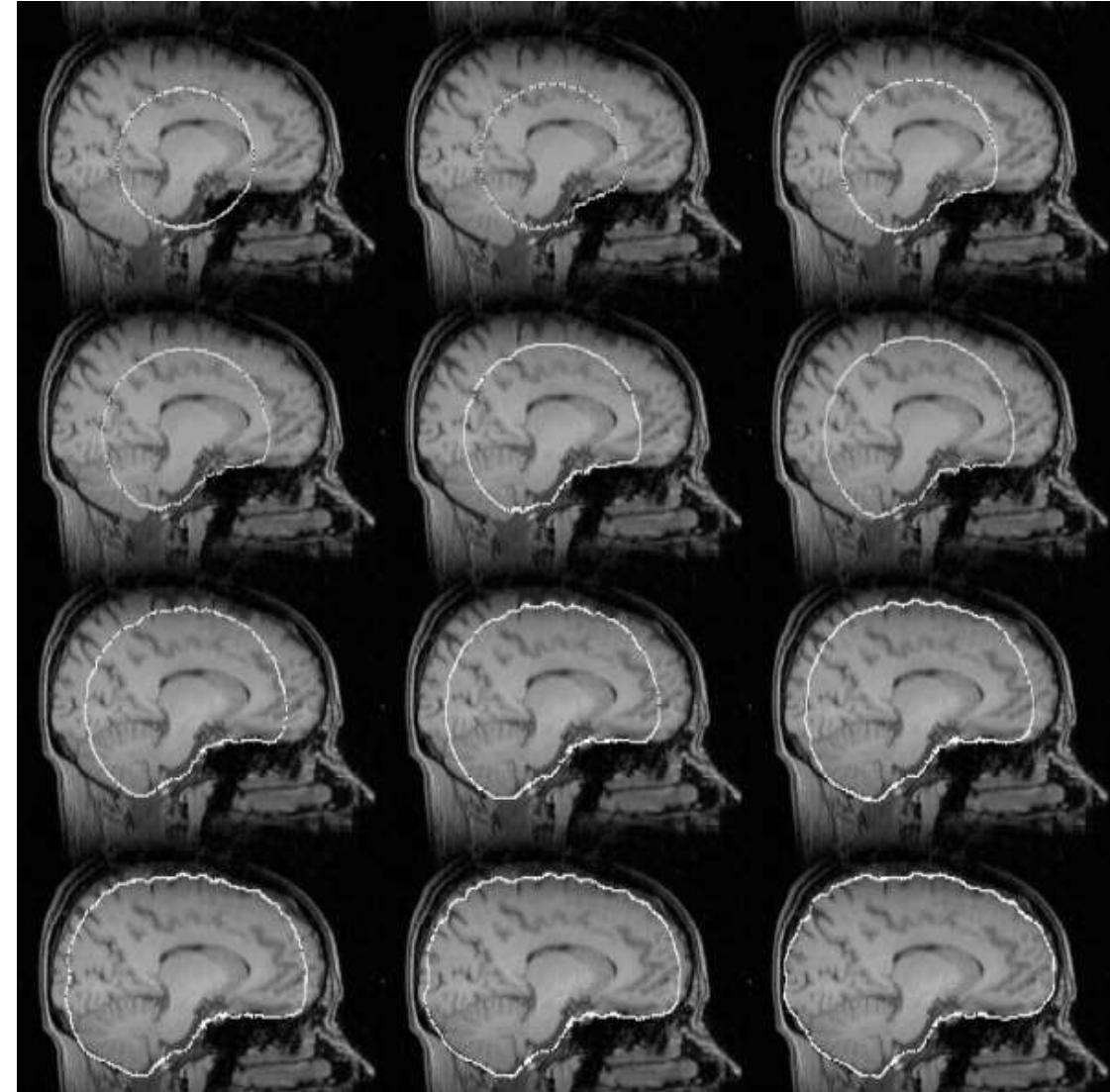


# BET: Grundidee und Beispiel



**Entwicklung der Fläche** durch die Summe von drei ad hoc definierten Kräften:

1. Drückt die Fläche nach außen, bis die Bildintensität unter einen adaptiven Schwellenwert sinkt
2. Wirkt starken Krümmungen entgegen
3. Verteilt die Stützpunkte möglichst gleichmäßig über die Fläche



Bildquelle: [Smith 2002]

# Vor- und Nachteile Aktiver Konturen

- **Vorteile:**
  - *Transparent*: Energiefunktional drückt in klarer mathematischer Sprache die Ziele der Segmentierung aus
  - *Relativ schnell*: Updates der Stützpunkte meist leicht auszurechnen
  - Enorme Flexibilität in der Formulierung von Potentialen
    - z.B. Integration von manuellen Vorgaben, Ballon-Kräften, Vorwissen
- **Nachteile:**
  - Berechnung der Euler-Lagrange-Gleichung zuweilen kompliziert
    - *Alternative*: Statt sie aus einem Energiefunktional herzuleiten werden die Kräfte manchmal einfach „ad hoc“ angegeben
  - Beibehalten gleichverteilter Stützpunkte und Änderungen in der Topologie erfordern komplexen Programmcode

## **4.4 Level-Set-Methode**

# Nachteile Expliziter Repräsentationen

- Explizite Repräsentation von Kurven durch Polygonzüge hat **Vorteile**:
  - Verschieben der Stützpunkte in jedem Schritt einfach und effizient
- **Nachteile:** Benötigt nichttrivialen Code um
  - eine angemessene Zahl von Stützpunkten sicherzustellen
  - topologische Änderungen durchzuführen
    - Verbinden oder Auftrennen von Kurven
    - Erzeugen oder Verschwinden von Komponenten

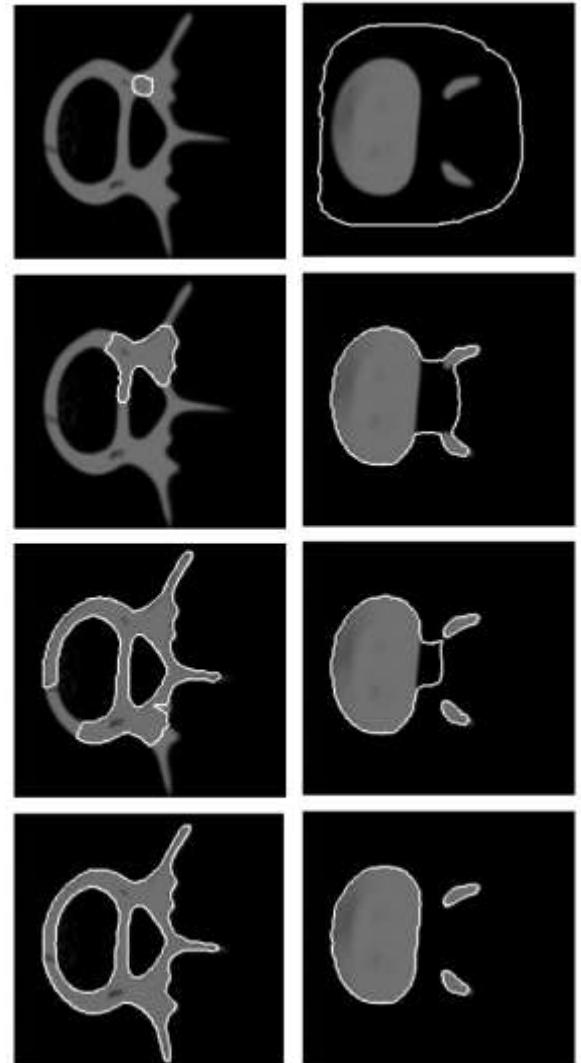
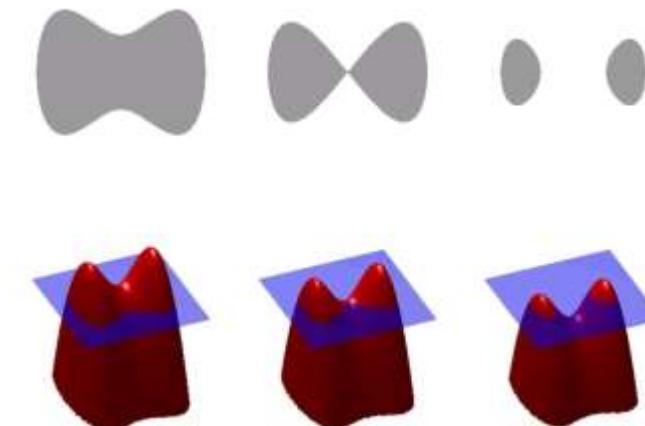
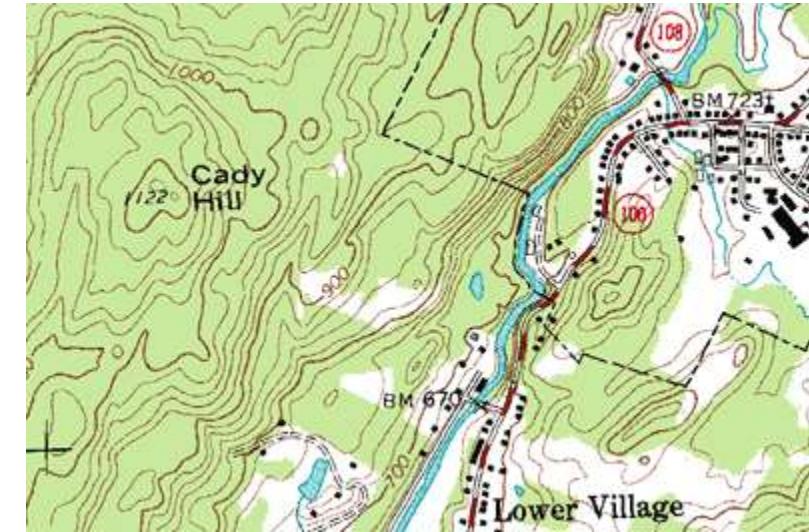


Bild: [McInerney/Terzopoulos 1999]

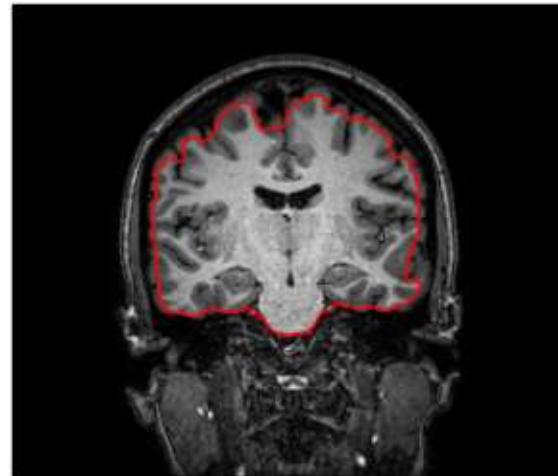
# Alternative: Darstellung durch Level Sets

- **Implizit** können Kurven als Niveaumenge (level set) einer Höhenfunktion  $\phi(x, y)$  dargestellt werden
- **Definition** einer Niveaumenge:  
$$\{(x, y) \mid \phi(x, y) = c\}$$
- **Vorteile:**
  - Topologische Änderungen der Kurve entsprechen kontinuierlichen Änderungen von  $\phi$ , stellen keine besonderen Schwierigkeit mehr dar
  - Darstellung komplexer Formen erfordert keine Erhöhung der Zahl von Stützpunkten

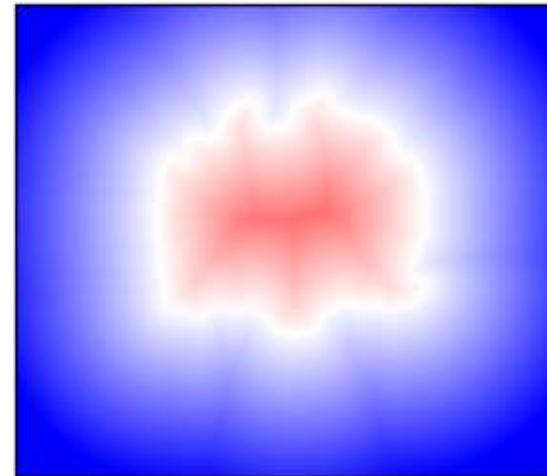


# Vorzeichenbehaftete Distanztransformation

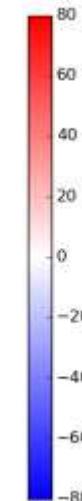
- Aus der initialen Kurve gewinnen wir ein entsprechendes Höhenfeld mit der **vorzeichenbehafteten Distanztransformation** (*engl.* signed distance transform, SDT)
  - Betrag gibt Abstand von der Kurve an
  - Vorzeichen gibt Vorder- oder Hintergrund an
    - *Konvention:* Vordergrund positiv, Hintergrund negativ



Kurve

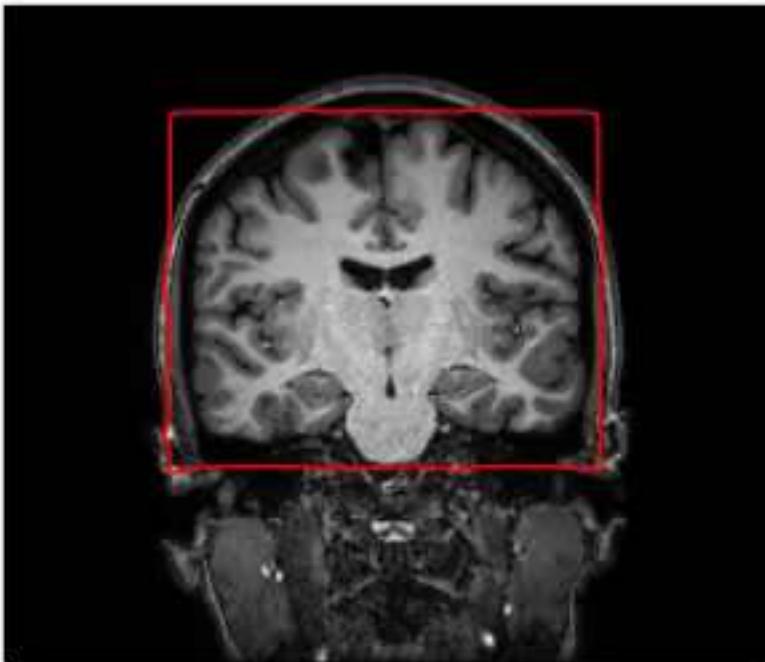


Höhenfunktion  $\phi$

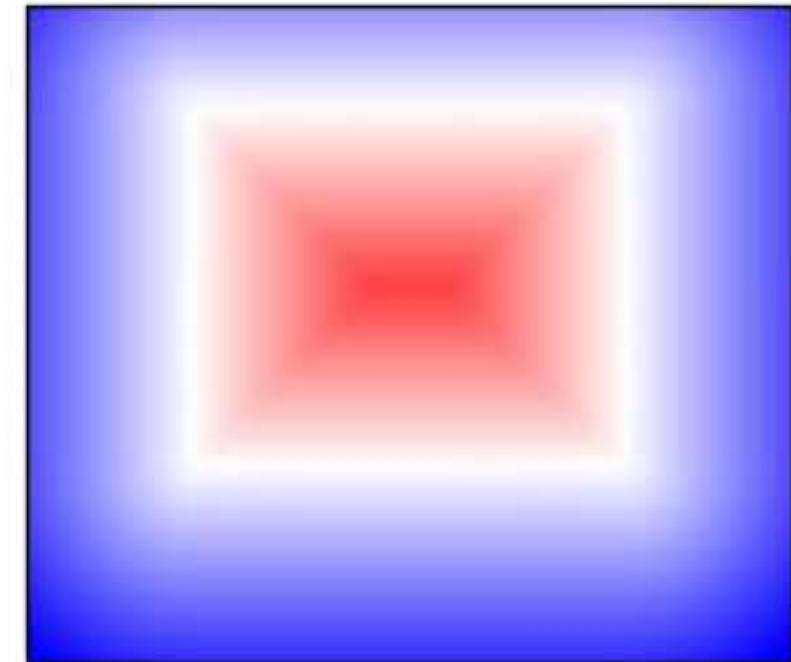


# Entwicklung von Level Sets

- Durch Level Sets dargestellte Kurven werden durch iterative **Updates ihrer Höhenfunktion** deformiert
  - Genaue Formel ergibt sich häufig aus einem Energiefunktional



Entwicklung der Kontur

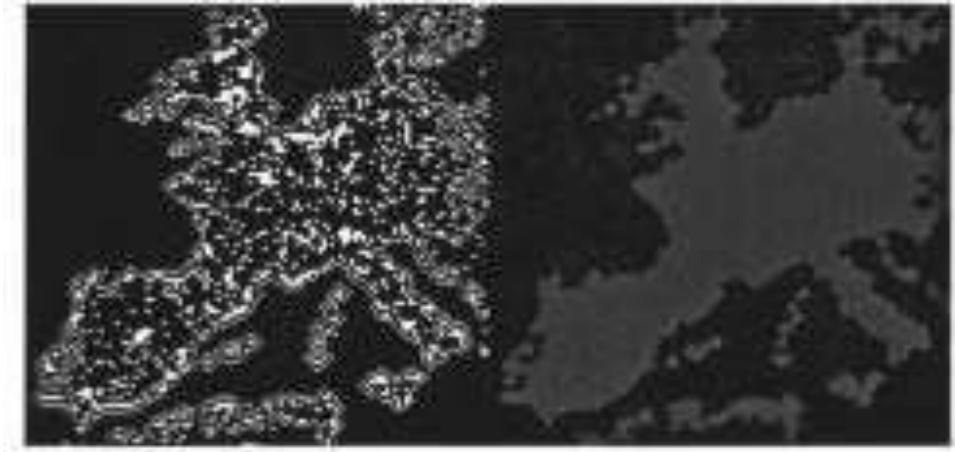


Entsprechendes Höhenfeld

# Beispiel: Energie-Funktional zur Segmentierung

- Erneut formalisiert ein **Energie-Funktional** unser Segmentierungsziel
  - *Beispiel:* Segmentierung in Regionen ungefähr konstanter Intensität, die durch möglichst kurze Kurven getrennt werden
    - Vereinfachte Version des Modells von [Mumford/Shah 1989]
  - *Formalisierung:* Bestimme Partition  $\Omega_1, \Omega_2$  des Definitionsbereichs eines Bildes  $\Omega$  ( $\Omega_2 = \Omega \setminus \Omega_1$ ) und Intensitäten  $\mu_1, \mu_2$ , die die folgende Energie  $E_{\text{MS}}$  minimieren:

$$E_{\text{MS}}(\mu_i, \Omega_i) = \int_{\Omega_1} (I(x) - \mu_1)^2 dx + \int_{\Omega_2} (I(x) - \mu_2)^2 dx + \nu |\partial \Omega_1|$$



# Update-Gleichung: Skizze der Herleitung

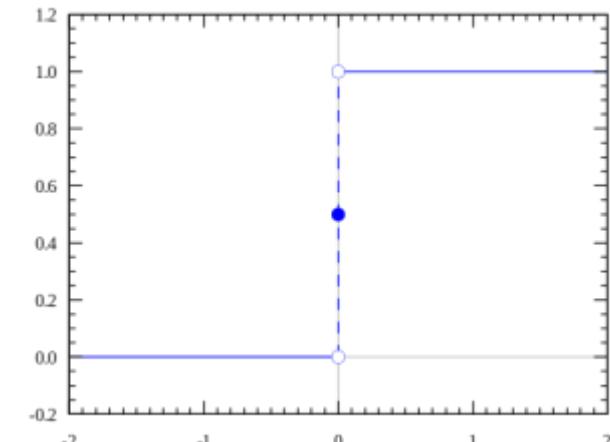
Zunächst schreiben wir

$$E_{\text{MS}}(\mu_i, \Omega_i) = \int_{\Omega_1} (I(x) - \mu_1)^2 dx + \int_{\Omega_2} (I(x) - \mu_2)^2 dx + \nu |\partial\Omega_1|$$

mittels der Heaviside (Sprung-)Funktion  $H$  als Funktional der Höhenfunktion  $\phi(x, y)$

$$\begin{aligned} E_{\text{MS}}(\mu_i, \phi) \\ = \int_{\Omega} H(\phi(x)) (I(x) - \mu_1)^2 + (1 - H(\phi(x))) (I(x) - \mu_2)^2 + \nu \|\nabla H(\phi(x))\| dx \end{aligned}$$

- Nach jedem Update von  $\phi$  werden  $\mu_1$  und  $\mu_2$  auf Mittelwerte von / inner- bzw. außerhalb gesetzt
- Gradientenabstieg per Euler-Lagrange-Gleichung erfordert eine Regularisierung von  $H$ .

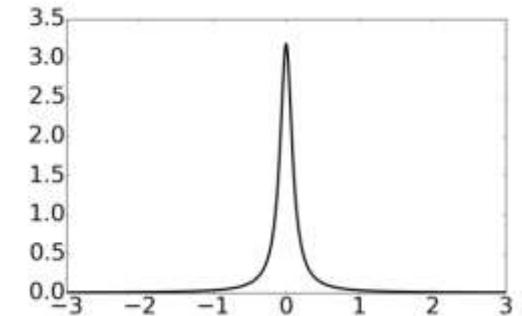


# Update-Regel nach Chan und Vese

- Nach Regularisierung von  $H$  lässt sich folgende Update-Regel herleiten [Chan/Vese 2001]:

$$\frac{\partial \phi}{\partial t} = \delta_\epsilon(\phi) \left[ (I - \mu_2)^2 - (I - \mu_1)^2 + \nu \operatorname{div} \left( \frac{\nabla \phi}{|\nabla \phi|} \right) \right]$$

wobei  $\delta_\epsilon(\phi) = \frac{1}{\pi} \frac{\epsilon}{\epsilon^2 + \phi^2}$  mit  $\epsilon > 0$

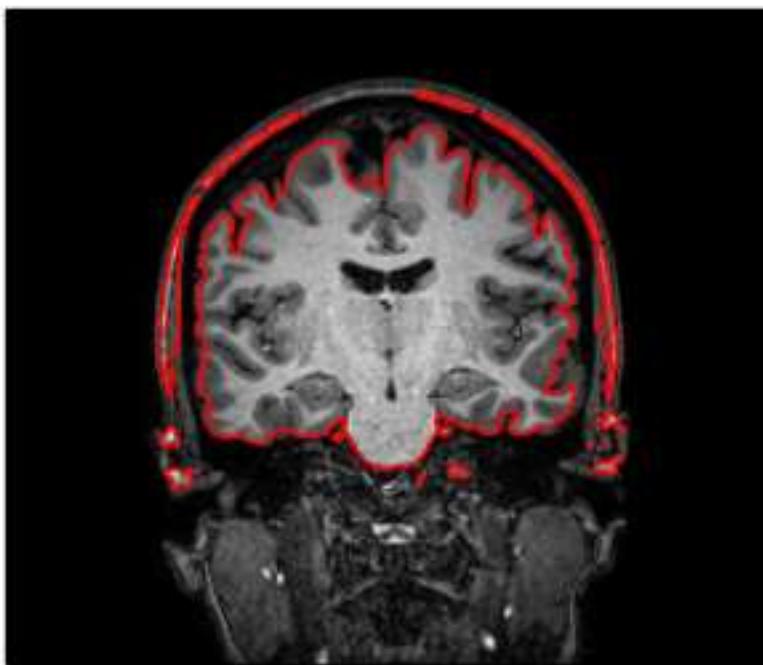


- Hinweis: Die genaue Herleitung des Divergenz-Terms zeigen wir nicht.
- Außerdem zu beachten:
  - Kontinuierliches „Nachführen“ von  $\mu_1$  und  $\mu_2$
  - $\phi$  sollte eine gültige Distanzfunktion bleiben
    - Weiterer Term in der Update-Funktion, den wir nicht weiter besprechen

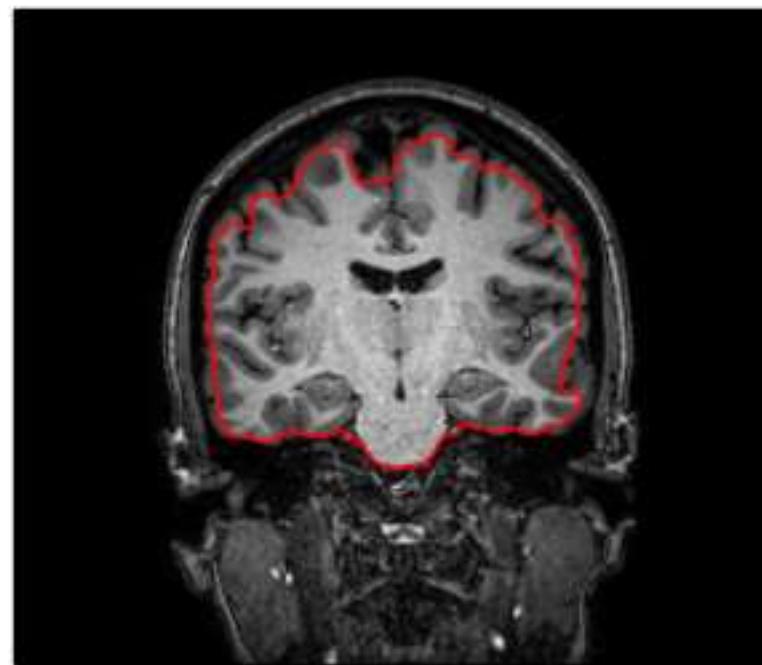
# Beispiel: Effekt des Regularisierungs-Parameters

Wahl von  $\nu$ :

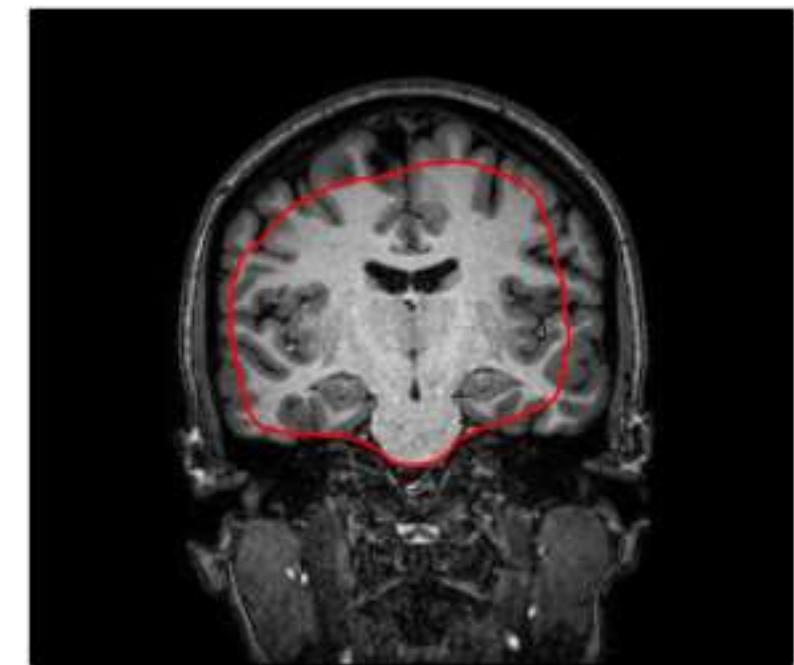
Wie stark bestrafen wir die Länge der Kontur?



$$\nu = 0.1$$



$$\nu = 1$$



$$\nu = 10$$

# Ausblick: Varianten der Level Sets

- Formulierung verschiedener Energiefunktionale ermöglicht hohe **Flexibilität** im Hinblick auf Segmentierungsziele, z.B.
  - Kantenbasierte Modelle
  - Komplexere regionenbasierte Modelle
  - Berücksichtigung von Vorwissen und interaktivem Feedback
- Verarbeitung von **Farb-** und **3D-Bildern**
- Einteilung in **mehrere Klassen** (gekoppelte Level Sets)

# Zusammenfassung: Level Sets

- **Vorteile der Level Sets**
  - **Einfach:** Topologische Änderungen verursachen keinen besonderen Implementierungsaufwand
  - **Transparent:** Im Energie-Funktional sind die Modell-Annahmen explizit ablesbar
  - **Flexibel:** Sowohl im Hinblick auf komplexe Konturen, als auch auf Umsetzbarkeit verschiedener Zielkriterien
- **Nachteile der Level Sets**
  - **Rechenaufwand**, insbesondere für große und 3D-Bilder
    - *Ansatz:* Update der Höhenfunktion nur in schmalem Band um die Kontur
  - Abhängigkeit von der **Initialisierung**

## **4.5 Aktive Formmodelle**

# Ein berühmtes Beispiel zur Motivation

- Was zeigt dieses Bild?
- Wie könnten wir es sinnvoll segmentieren?



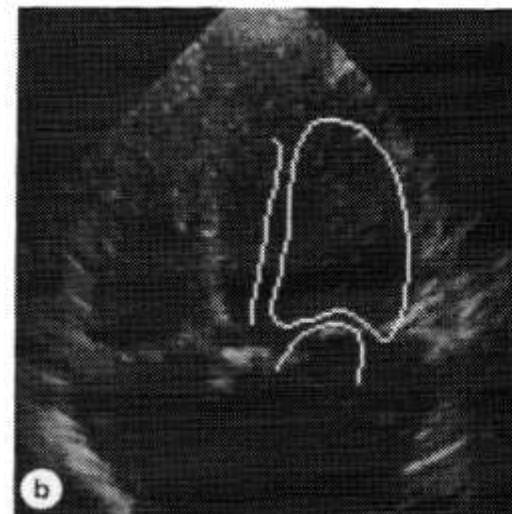
# Grundidee: Modellbasierte Segmentierung

- **Modellbasierte Segmentierung** nutzt spezifisches Vorwissen über das gesuchte Objekt aus
  - **Statistische Modelle** werden aus Beispielen gelernt
- **Formmodelle** sind hilfreich, wenn das gesuchte Objekt eine charakteristische Form hat (z.B. Knochen, Organe)
  - Modellieren typische Form und mögliche Abweichungen davon
  - Getrennt betrachtet werden Größe, Position und Orientierung, die zusammen als **Pose** der Form bezeichnet werden
- Bei der **Bildsegmentierung** mittels Formmodellen werden Pose und Formparameter bestimmt, die am besten zu dem Bildinhalt passen

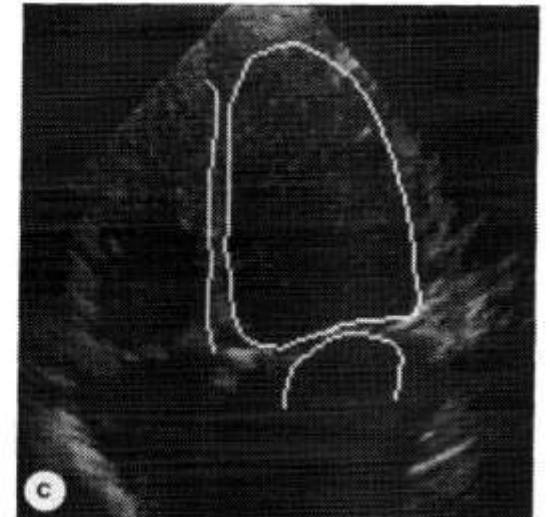
# Beispiel: Segmentierung mit Aktiven Formmodellen

Beispiel aus Cootes et al., *Active Shape Models – Their Training and Application* (1995):

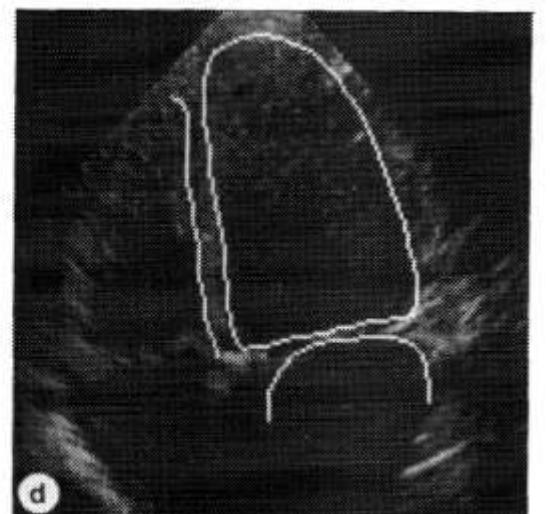
- Segmentierung einer Herzkammer in einem Echokardiogramm



Initialisierung



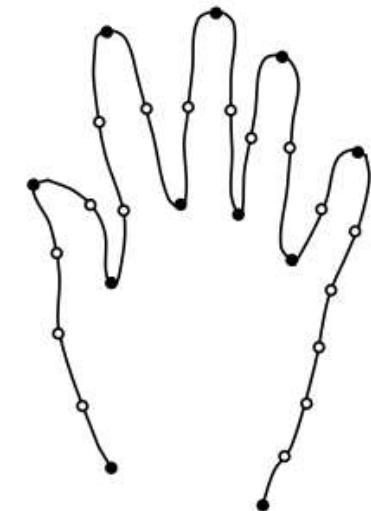
Nach 80 Iterationen



Nach 200 Iterationen 60

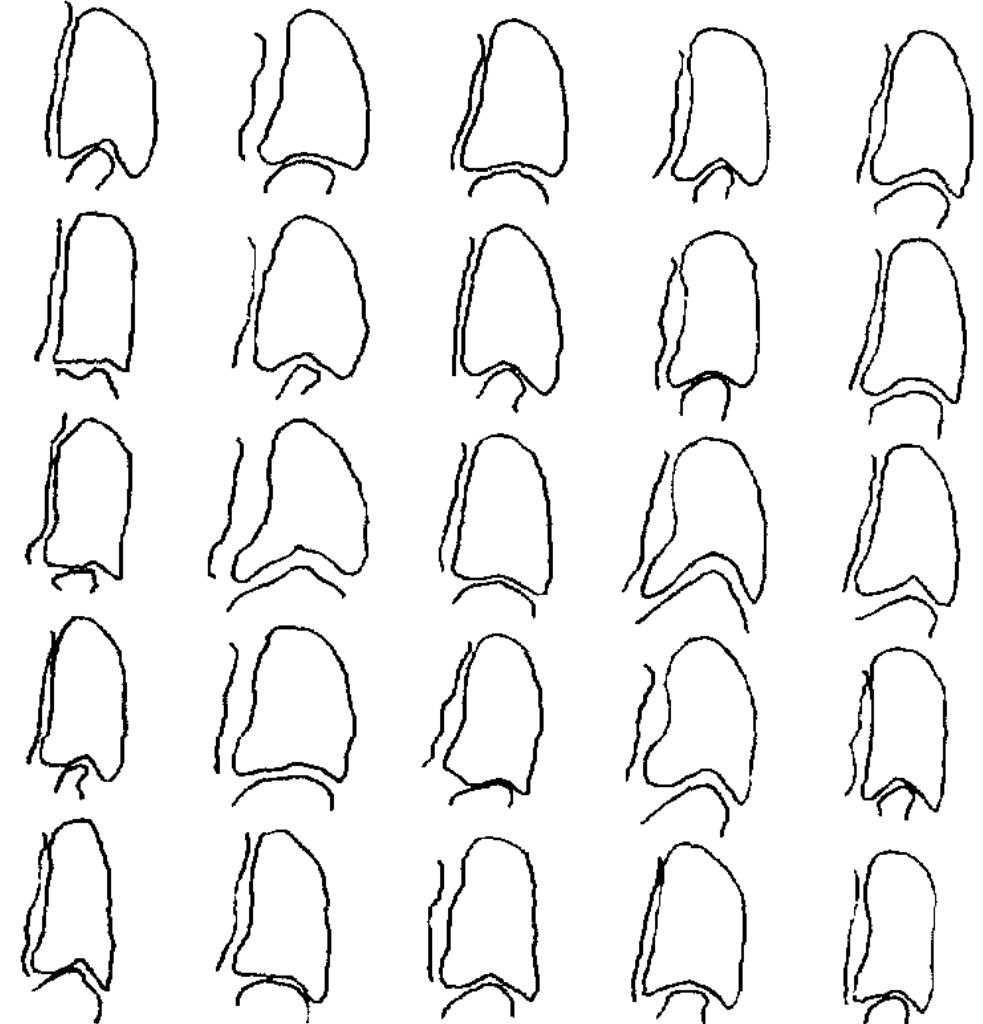
# Punktverteilungsmodell

- Ein **Punktverteilungsmodell** stellt eine Form durch die Koordinaten von  $n$  Stützpunkten in einer Referenzpose sowie durch ihre Verbindungen dar
  - Die **Stützpunkte** umfassen
    - **Landmarken**, eindeutig erkennbare Punkte, die in verschiedenen Bildern in Korrespondenz gebracht werden
    - Genügend **Hilfspunkte**, um die genaue Form zwischen den Landmarken hinreichend genau linear zu approximieren
  - 2D-Formen werden durch  $2n$ -dimensionale Vektoren  $\mathbf{x}_i$  beschrieben
    - Wir lernen das Formmodell aus  $m$  Beispielen,  $i = 1, 2, \dots, m$
  - Mögliche Variationen der Form werden durch eine **gemeinsame Wahrscheinlichkeitsverteilung** der Punktkoordinaten erfasst



# Erlernen des Formmodells aus Beispielen

- Um die typische Form und ihre möglichen Variationen zu erlernen, werden zunächst **Trainingsbilder** von Hand annotiert
  - Punkte in jedem Beispiel müssen **korrespondieren**, daher ist die Auswahl geeigneter Landmarken wichtig
  - Anhaltspunkt zur **benötigten Zahl** von Trainingsbildern: Kann jede Form sinnvoll durch ein Modell erzeugt werden, das aus den übrigen  $m - 1$  trainiert wurde?



Teilmenge von 66 Trainingsbeispielen

# Ausrichtung zweier Formen

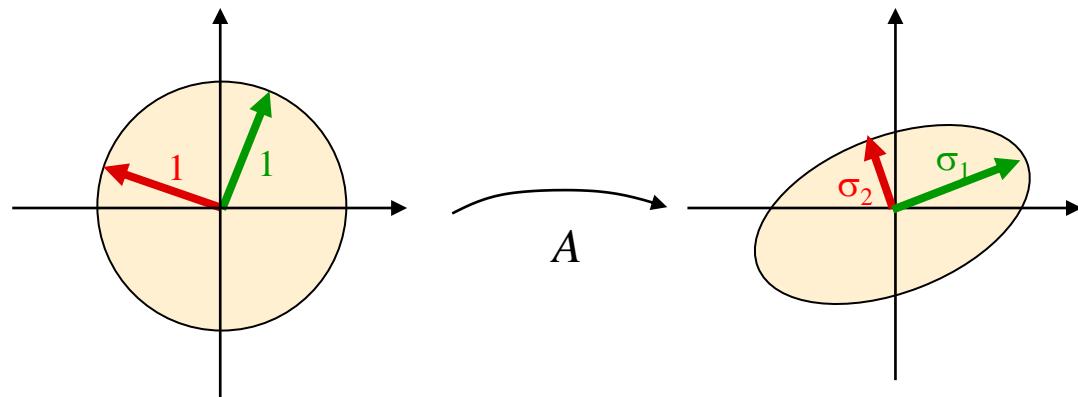
- Um die Abweichung zwischen zwei Formen  $\mathbf{x}_1$  und  $\mathbf{x}_2$  zu bestimmen, benötigen wir zunächst ihre **relative Pose**, d.h. Skalierung  $s$ , Verschiebung  $\mathbf{t}$  und Rotation  $\theta$ , die  $\|T_{s,\mathbf{t},\theta}(\mathbf{x}_1) - \mathbf{x}_2\|$  minimieren
- Die entsprechende Ausrichtung von Formen bezeichnet man als **Prokrustes-Analyse**
- Beliebte Implementierung im Kontext von Formmodellen:
  1. **Verschiebung** um die Differenz der Mittelpunkte  $\mathbf{t} = \mathbf{m}_2 - \mathbf{m}_1$
  2. Skalierung auf dieselbe **Norm**,  $s = \frac{\|\mathbf{z}_2\|}{\|\mathbf{z}_1\|}$ 
    - $\mathbf{z}_1$  und  $\mathbf{z}_2$  sind  $\mathbf{x}_1$  und  $\mathbf{x}_2$  nach Zentrierung auf den Ursprung
  3. Rotation mittels der **Singulärwertzerlegung** (s. nächste Folie)

# Singulärwertzerlegung

- Satz:** Für jede Matrix  $A \in \mathbb{R}^{m \times n}$  existiert eine **Singulärwertzerlegung** (engl. Singular value decomposition, SVD)

$$A = U\Sigma V^T$$

- Spaltenvektoren der orthogonalen Matrizen  $U \in \mathbb{R}^{m \times m}$  und  $V \in \mathbb{R}^{n \times n}$  heißen Links- bzw. Rechts-Singulärvektoren
- positive Diagonalelemente  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p \geq 0$  ( $p = \min\{m, n\}$ ) der diagonalen Matrix  $\Sigma \in \mathbb{R}^{m \times n}$  heißen Singulärwerte



$$AV = U\Sigma \Rightarrow Av_i = \sigma_i u_i$$

$$A [v_1 \ v_2 \ \dots \ v_n] = [u_1 \ u_2 \ \dots \ u_n] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}$$

# Rotation per Singulärwertzerlegung



**Algorithmus** zur Berechnung der Rotation der auf den Ursprung zentrierten Formen  $\mathbf{z}_1$  und  $\mathbf{z}_2$ :

1. Stelle die 2D-Formen durch  $n \times 2$  Matrizen  $\mathbf{Z}_1$  und  $\mathbf{Z}_2$  dar
2. Berechne die  $2 \times 2$  Kreuzkovarianzmatrix  $\mathbf{W} = \mathbf{Z}_1^T \mathbf{Z}_2$
3. Berechne die Singulärwertzerlegung  $\mathbf{W} = \mathbf{U} \Sigma \mathbf{V}^T$
4. Die Rotation, die  $\mathbf{z}_1$  optimal auf  $\mathbf{z}_2$  abbildet, ergibt sich als
$$\mathbf{R} = \mathbf{V} \mathbf{U}^T$$
  - Streng genommen müssen wir überprüfen, dass  $\det(\mathbf{R}) = 1$ . Dies ist in der Praxis jedoch meistens der Fall.

# Berechnung der Referenzform: Problemstellung

- Mittels Prokrustes-Analyse können wir nun aus allen  $m$  Formen eine mittlere **Referenzform** berechnen
- Die **Pose** der Referenzform  $\bar{\mathbf{x}}$  geben wir dabei vor:
  - **Skalierung** des Formvektors  $\bar{\mathbf{x}}$  auf Norm  $\|\bar{\mathbf{x}}\| = 1$
  - **Position**: Zentrierung auf den Ursprung
  - **Orientierung**: Durch ein initiales Beispiel festgelegt
- Ziel ist die Berechnung einer mittleren Form  $\bar{\mathbf{x}}$  und von Posen der einzelnen Formen, um die quadratische Abweichung
$$D = \sum_i \|T_{s_i, \mathbf{t}_i, \theta_i}(\mathbf{x}_i) - \bar{\mathbf{x}}\|^2$$
zu minimieren

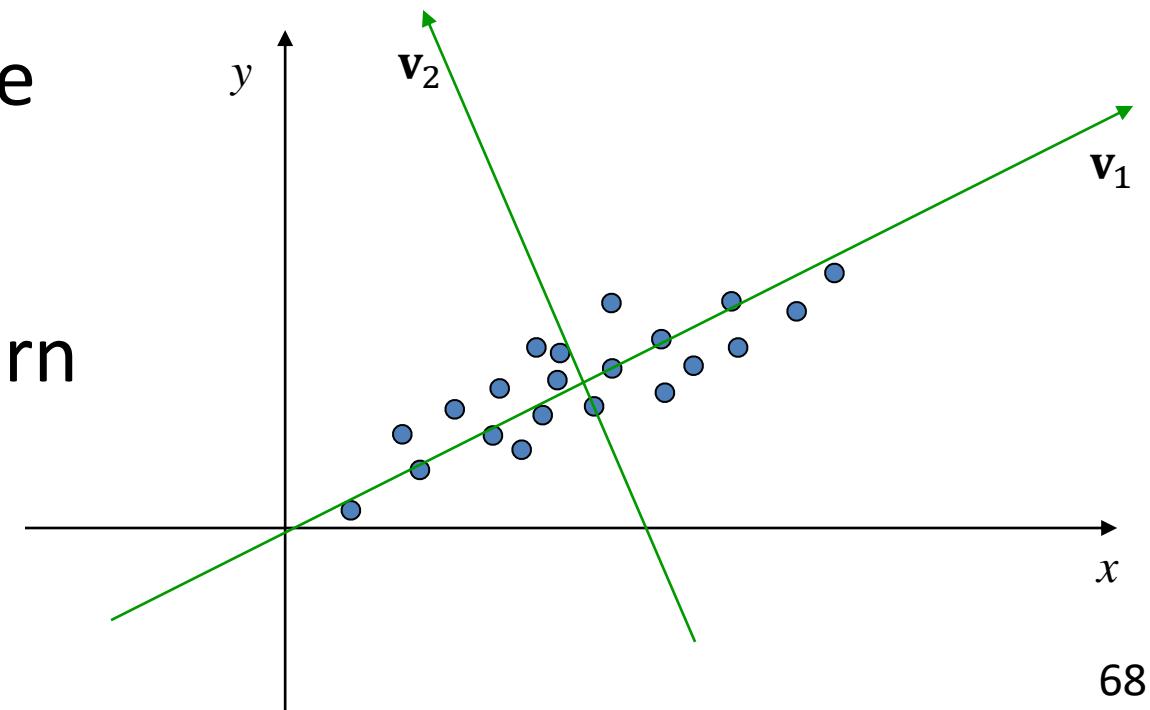
# Berechnung der Referenzform: Vorgehen

**Algorithmus** zur iterativen Berechnung der Referenzform:

1. Zentriere jede Form um den Ursprung und normiere sie auf  $\|\mathbf{x}_i\| = 1$
2. Wähle ein so normiertes Beispiel als initiale Referenz  $\bar{\mathbf{x}}_0$  aus
3. Wiederhole für  $j = 0, 1, 2, \dots$ , bis  $\|\bar{\mathbf{x}}_{j+1} - \bar{\mathbf{x}}_j\| < \epsilon$ :
  - i. Richte alle  $\mathbf{x}_i$  an  $\bar{\mathbf{x}}_j$  aus, nenne die Resultate  $\mathbf{x}'_i$
  - ii. Berechne  $\bar{\mathbf{x}}_{j+1} := \frac{1}{m} \sum_{i=1}^m \mathbf{x}'_i$
  - iii. Richte  $\bar{\mathbf{x}}_{j+1}$  an  $\bar{\mathbf{x}}_0$  aus und normiere auf  $\|\bar{\mathbf{x}}_{j+1}\| = 1$

# Hauptkomponenten-Analyse: Anschauung

- Die **Hauptkomponentenanalyse** (*engl.* Principal Component Analysis, PCA) bestimmt aus  $m$  gegebenen Punkten  $\mathbf{x}_i \in \mathbb{R}^p$  so  $p$  orthogonale Richtungen  $\mathbf{v}_i$ , dass die Projektionen von  $\mathbf{x}_i$  auf die Unterräume  $\langle \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_t \rangle$  jeweils so viel Varianz wie möglich erhalten.
- Formmodelle nutzen sie dazu, die an  $\bar{\mathbf{x}}$  ausgerichteten Trainingsformen  $\mathbf{x}_i \in \mathbb{R}^{2n}$  auf eine handhabbare Zahl von Formparametern zu reduzieren
  - Schränkt das Modell auf plausible Deformationen ein



# Hauptkomponenten-Analyse: Algorithmus

**Algorithmus** zur Berechnung der Hauptkomponenten-Analyse:

1. Berechne den **Mittelwert** der Punkte,  $\mu = \frac{1}{m} \sum_{i=1}^m \mathbf{x}_i$
2. Berechne die **Kovarianzmatrix** der Punkte,

$$S = \frac{1}{m-1} \sum_{i=1}^m (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$$

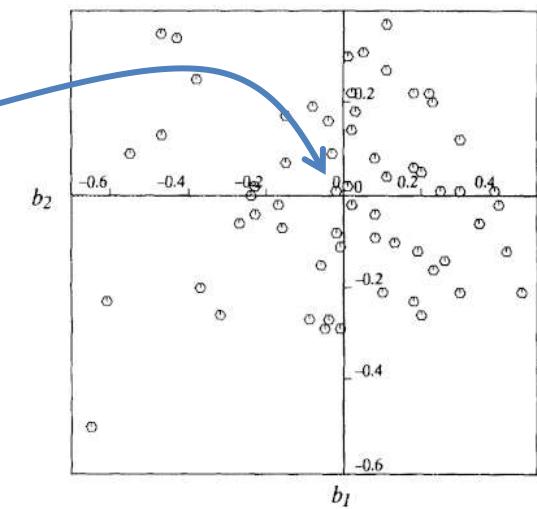
3. Berechne die **Eigenvektoren**  $\mathbf{v}_i$  von  $S$  und die zugehörigen **Eigenwerte**  $\lambda_i$ . Sortiere sie absteigend ( $\lambda_i \geq \lambda_{i+1}$ )

Die Eigenvektoren  $\mathbf{v}_i$  sind die Hauptkomponenten, die Eigenwerte  $\lambda_i$  geben die Varianz in der jeweiligen Richtung an. Die totale Varianz ergibt sich als Summe aller Eigenwerte.

# Dimensionsreduktion per Hauptkomponenten

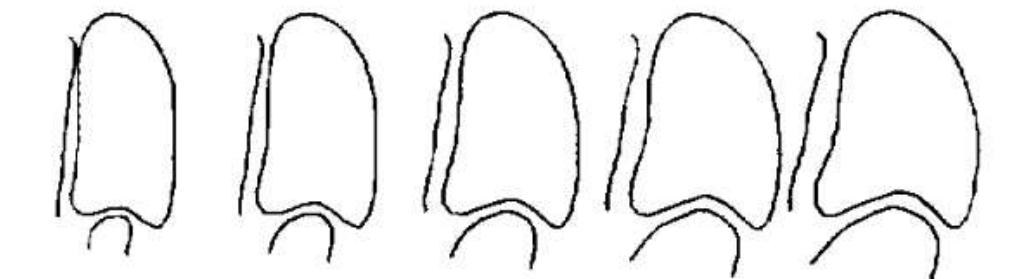
- Formmodelle nutzen nur die wichtigsten Hauptkomponenten
  - *Mögliches Kriterium:* Kleinste Dimension  $t$ , die mindestens z.B.  $\theta = 98\%$  der totalen Varianz erhält:  
$$\frac{\sum_{i=1}^t \lambda_i}{\sum_{i=1}^p \lambda_i} \geq \theta$$
  - Bilde aus den entsprechenden Spaltenvektoren  $\mathbf{v}_i$  eine Matrix  $\mathbf{P}$
  - Berechnung der Formparameter  $\mathbf{b} \in \mathbb{R}^t$  per Projektion  
$$\mathbf{b} = \mathbf{P}^T(\mathbf{x} - \bar{\mathbf{x}})$$
  - Erzeugen einer Form aus  $\mathbf{b}$  per  $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}$
  - Koeffizienten von  $\mathbf{b}$  sollten im Rahmen der Trainingsvarianz liegen, z.B.  $b_i \in [-3\sqrt{\lambda_i}, 3\sqrt{\lambda_i}]$

Eigenvalue	$\frac{\lambda_i}{\lambda_T} \times 100\%$
$\lambda_1$	37%
$\lambda_2$	17%
$\lambda_3$	13%
$\lambda_4$	7%
$\lambda_5$	6%
$\lambda_6$	4%

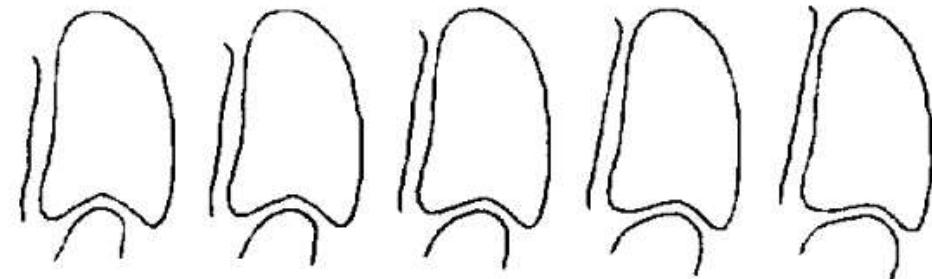


# Illustration: Hauptkomponenten eines Formmodells

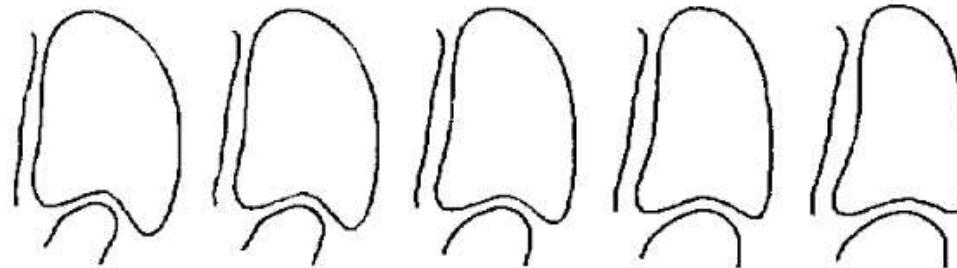
- Visualisierung der Hauptkomponenten zeigt die vom Modell zugelassenen Deformationen



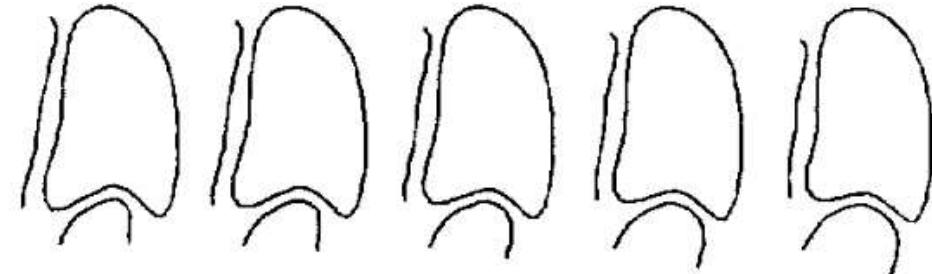
$$-2\sqrt{\lambda_1} \xleftarrow{b_1} \xrightarrow{2\sqrt{\lambda_1}}$$



$$-2\sqrt{\lambda_2} \xleftarrow{b_2} \xrightarrow{2\sqrt{\lambda_2}}$$



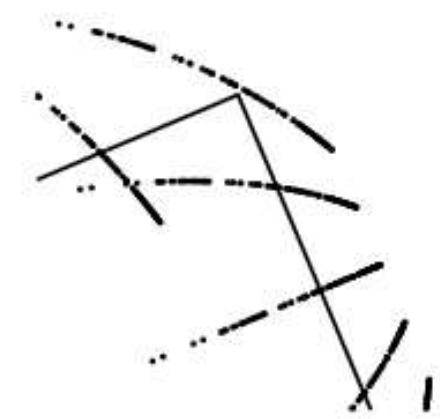
$$-2\sqrt{\lambda_3} \xleftarrow{b_3} \xrightarrow{2\sqrt{\lambda_3}}$$



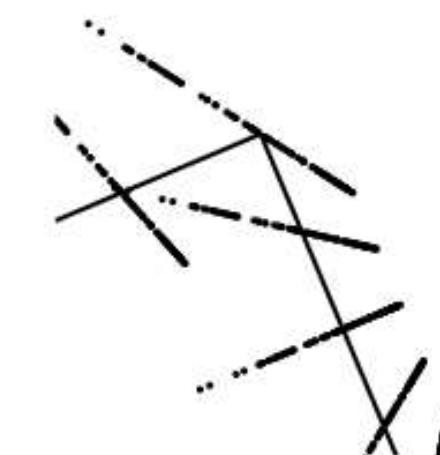
$$-2\sqrt{\lambda_4} \xleftarrow{b_4} \xrightarrow{2\sqrt{\lambda_4}}$$

# Technisches Detail: Tangentialraum von $\bar{x}$

- Deformationen von  $\bar{x}$  in Richtung  $\bar{x}$  skalieren die Form. Da Skalierung Teil der Pose ist, ist dies unerwünscht.
- Zentrierte Formen  $z$  werden daher vor ihrer Einbettung in den Formenraum so auf  $x'$  abgebildet, dass die entsprechende Deformation zu  $\bar{x}$  orthogonal ist
  - Kriterium:  $(x' - \bar{x}) \cdot \bar{x} = 0$
  - Wegen  $\|\bar{x}\| = 1$  ergibt sich die Bedingung  $x' \cdot \bar{x} = 1$
  - Führt zur Skalierung  $x' = \frac{1}{z \cdot \bar{x}} z$
  - Wird als „Projektion auf den Tangentialraum“ bezeichnet



Punkte von  $\frac{z}{\|z\|}$



Punkte von  $x'$

# Anpassen des Modells an gegebene Punkte

**Algorithmus** zur iterativen Berechnung der Form- und Posenparameter aus gegebenen Punkten  $\mathbf{y}_B$  im Bildraum:

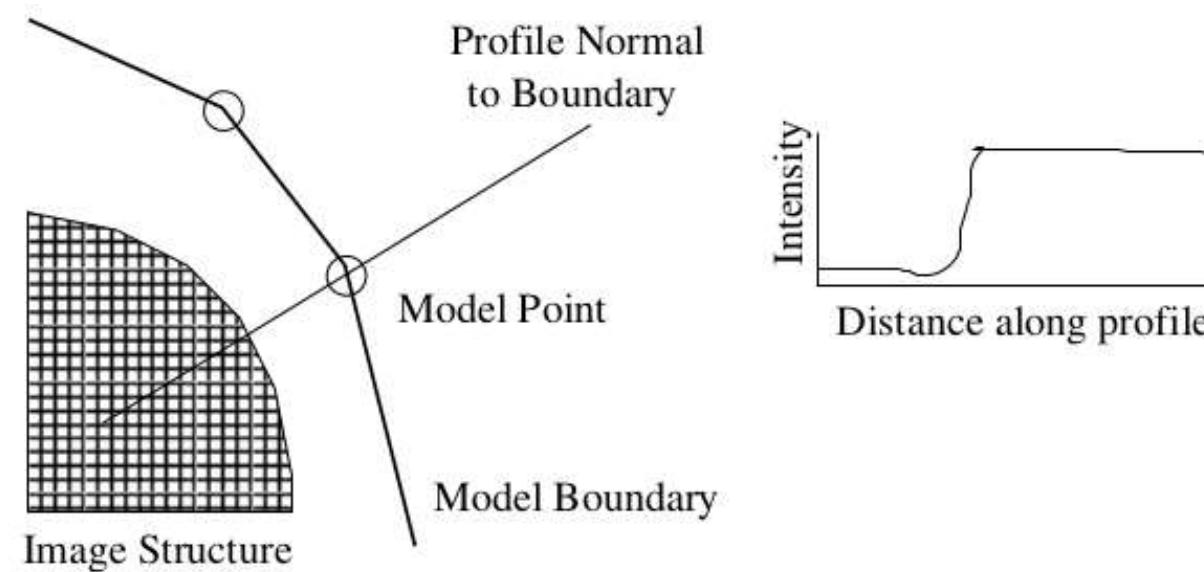
1. Initialisiere die Formparameter auf  $\mathbf{b} = 0$
2. Wiederhole, solange Pose oder Form sich um mehr als  $\epsilon$  ändern:
  - i. Erzeuge die aktuelle Form:  $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}$
  - ii. Bestimme  $s, \mathbf{t}, \theta$ , die  $\mathbf{x}$  an  $\mathbf{y}_B$  ausrichten.
  - iii. Transformiere  $\mathbf{y}_B$  in den Modellraum:  $\mathbf{y} = T_{s,\mathbf{t},\theta}^{-1}(\mathbf{y}_B)$
  - iv. Projizierte  $\mathbf{y}$  in den Tangentialraum von  $\bar{\mathbf{x}}$ :  $\mathbf{y}' = \frac{1}{\mathbf{y} \cdot \bar{\mathbf{x}}} \mathbf{y}$
  - v. Berechne die zu  $\mathbf{y}'$  passenden Modellparameter:  $\mathbf{b} = \mathbf{P}^T (\mathbf{y}' - \bar{\mathbf{x}})$

# Bildsegmentierung mit Formmodellen

- Ähnlich wie **aktive Konturen** können sich grob initialisierte Formmodelle zur Segmentierung aktiv an Bildinhalte anpassen
- Grundsätzlich iteriert man dabei folgende Schritte, bis sich die Modell-Parameter kaum noch ändern:
  1. Finde in der Umgebung jeden Punkts der aktuellen Form eine optimale Position im Bild (Details: nächste Folien)
  2. Passe das Modell an diese neuen Punkte  $\mathbf{y}_B$  an
  3. Schränke die gefundenen Parameter auf plausible Werte ein
    - Übliche Bedingung:  $|b_i| < 3\sqrt{\lambda_i}$

# Suche nach stärkster Kante

- Einfache **kantenbasierte Suche** in Schritt 1:
  - Bestimme innerhalb eines Suchradius senkrecht zur aktuellen Kontur den Ort der stärksten Kante (s. Kapitel 1)
  - Nutze ggf. Vorwissen über die **Orientierung**: Ist das gesuchte Objekt heller oder dunkler als seine Umgebung?



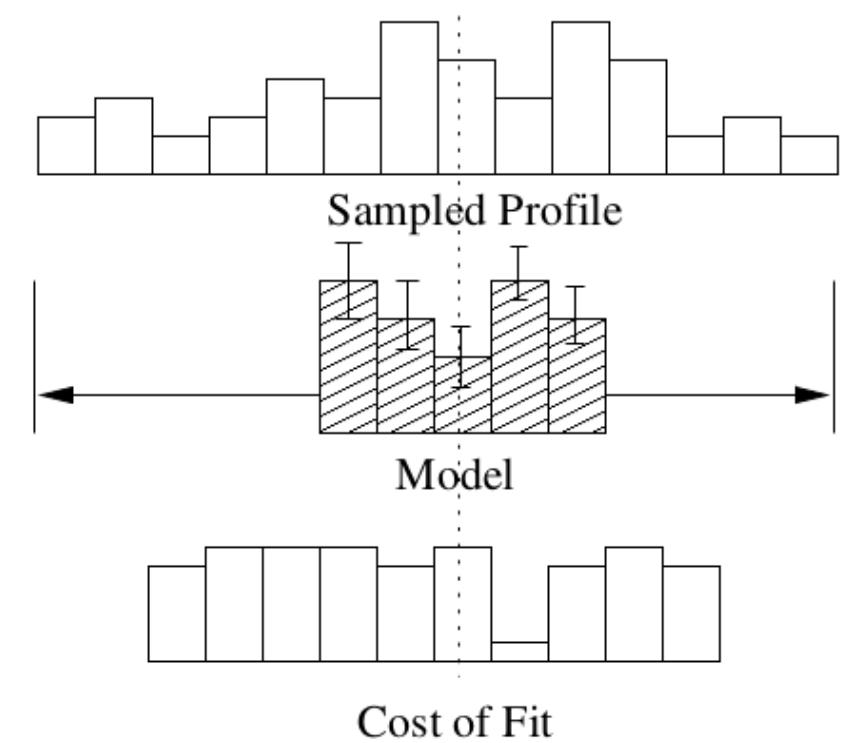
# Lernen eines Kantenprofils

- *Idee:* Lerne aus den Trainingsbildern für jeden Stützpunkt nicht nur die Position, sondern auch ein Kantenprofil, um in Schritt 1 eine gezieltere Suche zu ermöglichen
  - Bestimme für  $k$  Pixel auf jeder Seite der Kontur die Intensitätsableitungen entlang der Normalen. Normiere die resultierenden Vektoren  $\mathbf{g}_i \in \mathbb{R}^{2k+1}$ 
$$\tilde{\mathbf{g}}_i = \frac{\mathbf{g}_i}{\|\mathbf{g}_i\|_1}$$
um globale Unterschiede im Kontrast auszugleichen
  - Berechne für jeden Punkt ein mittleres Profil  $\bar{\mathbf{g}}$  und eine Kovarianz  $\mathbf{S}_{\mathbf{g}}$

# Suche nach plausiblem Kantenprofil

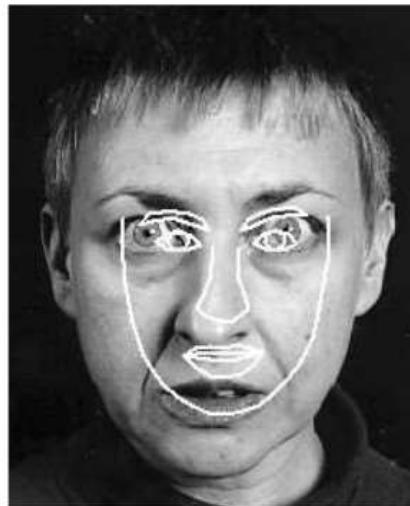
- Verfeinerte **Suche mittels des gelernten Kantenprofils:**
  - Berechne in Schritt 1 für jeden Kandidatenpixel  $p$  senkrecht zur Kontur das entsprechende normierte Kantenprofil  $\tilde{\mathbf{g}}_p$
  - Wähle den Kandidaten mit der geringsten **Mahalanobis-Distanz** als optimale Position

$$D(\tilde{\mathbf{g}}_p) = (\tilde{\mathbf{g}}_p - \bar{\mathbf{g}})^T \mathbf{S}_g^{-1} (\tilde{\mathbf{g}}_p - \bar{\mathbf{g}})$$



Cost of Fit

# Beispiele: Segmentierung mit Aktiven Formmodellen



Initial



After 2 iterations

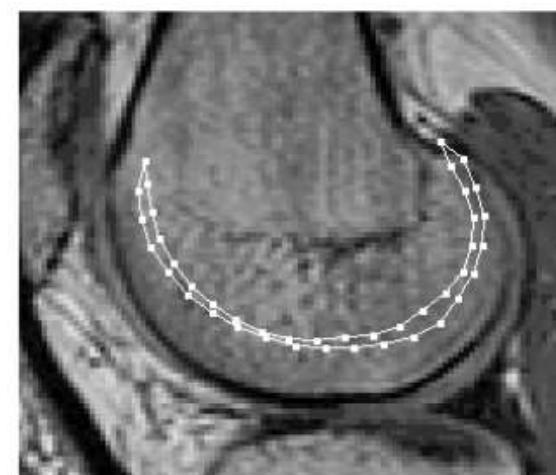


After 6 iterations

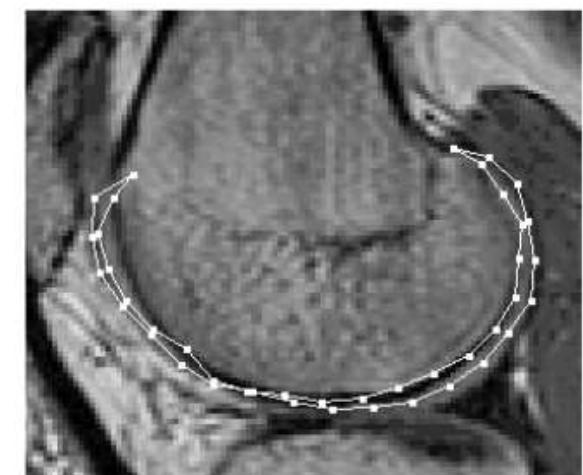


After 18 iterations

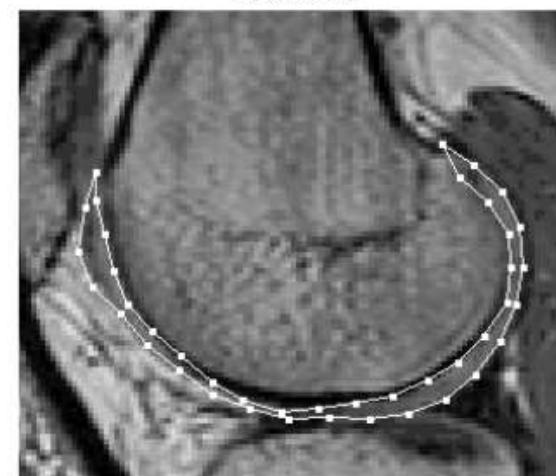
Gesichtserkennung



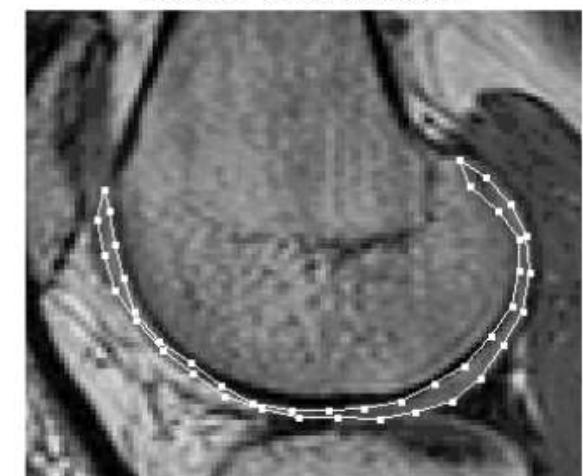
Initial



After 1 iteration



After 6 iterations



After 14 iterations

Knorpelschicht in Knie-MRT

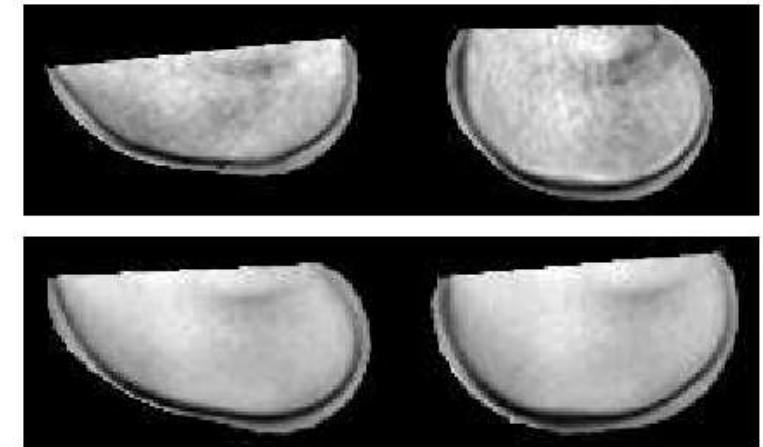
# Zusammenfassung: Aktive Formmodelle

- **Aktive Formmodelle** nutzen zur Segmentierung Vorwissen über die Form der gesuchten Struktur
  - *Vorteil:* Hohe Robustheit, wenn das Modell tatsächlich passt
  - Nicht geeignet für anomales Wachstum (z.B. Tumore) oder andere Verletzungen der Modellannahmen (z.B. Knochenbrüche)
- Wesentliche **Ideen und Bausteine** der Formmodelle sind:
  - Erlernen eines Modells aus annotierten **Trainingsdaten**
  - **Prokrustes-Analyse** zur Bestimmung relativer Posen
  - **Hauptkomponenten-Analyse** zur Dimensionsreduktion
  - Suche nach geeigneten Modellparametern über **stärkste Kanten** oder erlernte **Kantenprofile**

# Ausblick: Aktive Erscheinungsmodelle

**Aktive Erscheinungsmodelle** (*engl. Active Appearance Models*) erweitern aktive Formmodelle um ein statistisches Modell des erwarteten Erscheinungsbilds (Grau- bzw. Farbwerte) in der Referenzform

- Ansatz: Analyse durch Synthese



Initial

2 its

Converged (11 its)

## Zum Nach- und Weiterlesen

- Heinz Handels: *Medizinische Bildverarbeitung.* Vieweg+Teubner, 2. Auflage, 2009
- Klaus D. Toennies: *Guide to Medical Image Analysis. Methods and Algorithms.* Springer, 2012
- I.N. Bankman: *Handbook of Medical Imaging. Processing and Analysis.* Academic Press, 2000

# Zum Nach- und Weiterlesen: Level-Set-Methode

- **YouTube-Video:**
  - Vorlesung „Variational Methods for Computer Vision“ von Prof. Daniel Cremers (TUM)
- **Bücher:**
  - Sethian: Level Set Methods and Fast Marching Methods, Cambridge University Press, 1999
  - Osher/Fedkiw: Level Set Methods and Dynamic Implicit Surfaces, Springer 2003
  - Mitiche/Ayed: Variational and Level Set Methods in Image Segmentation, Springer 2011

# Zum Nach- und Weiterlesen: Aktive Formmodelle

- TF Cootes, CJ Taylor, DH Cooper, J Graham: *Active Shape Models – Their Training and Application*. Computer Vision and Image Understanding, 1995
- Tim Cootes: *An Introduction to Active Shape Models*. Kapitel 7 in „Model-Based Methods in Analysis of Biomedical Images“, Oxford University Press, 2000
- TF Cootes, GJ Edwards, CJ Taylor: *Active Appearance Models*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2001
- Mikkel B. Stegmann: *Active Appearance Models. Theory, Extensions & Cases*. MSc-Arbeit an der Technical University of Denmark (DTU), 2000

# Kapitel 5: Bildregistrierung

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

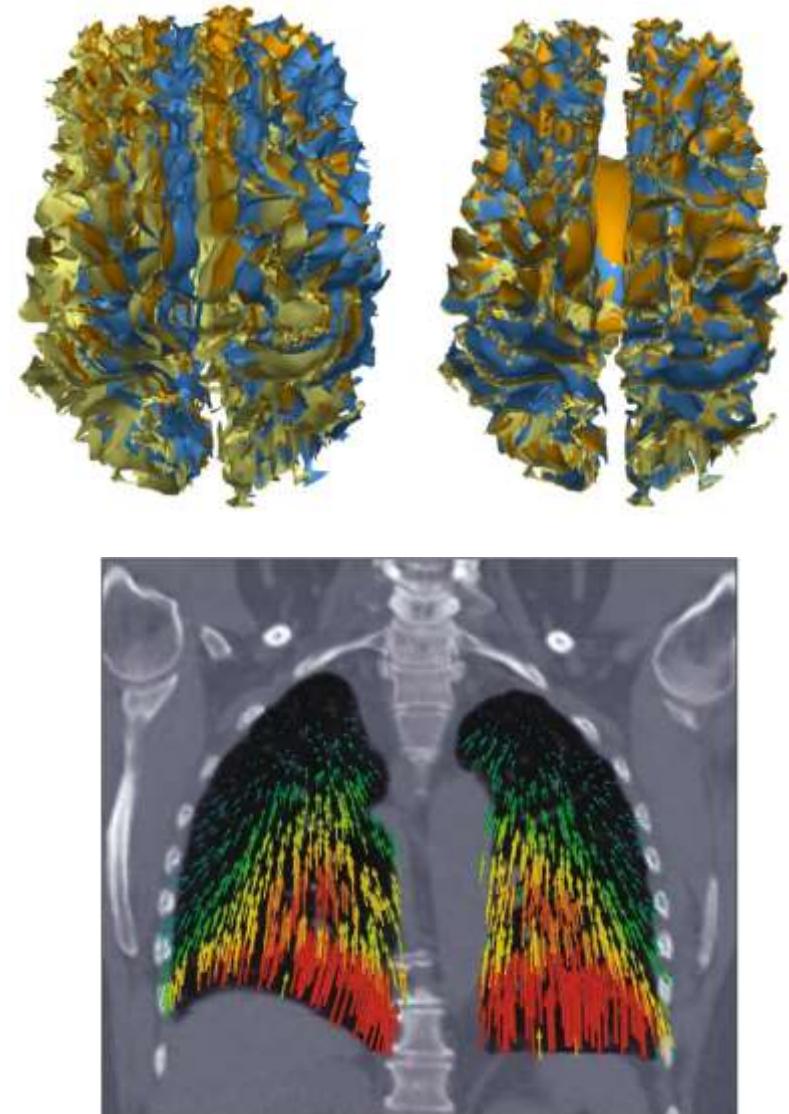
Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

16./23. Dezember 2024

## **5.1 Problemstellung und Evaluierung**

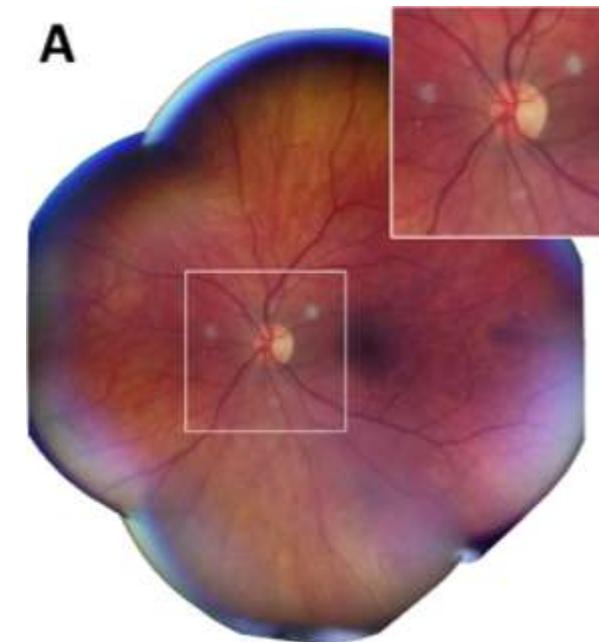
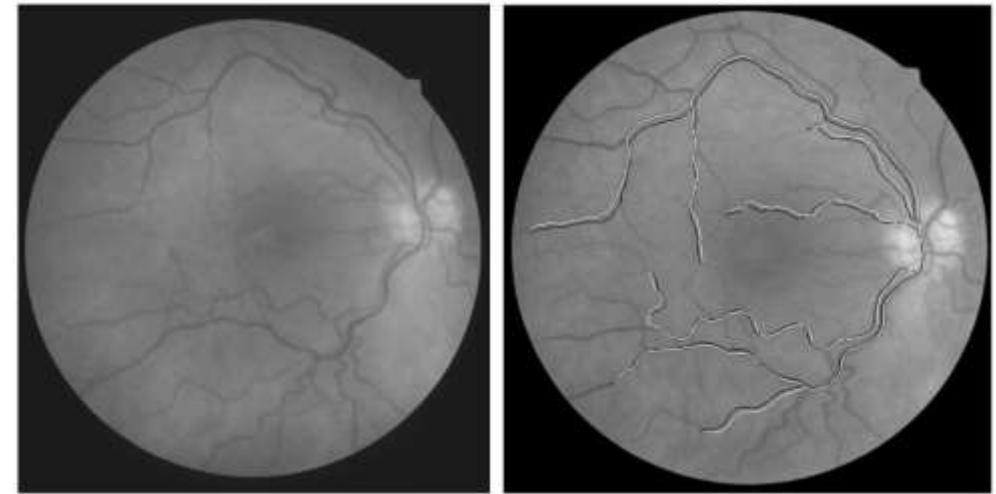
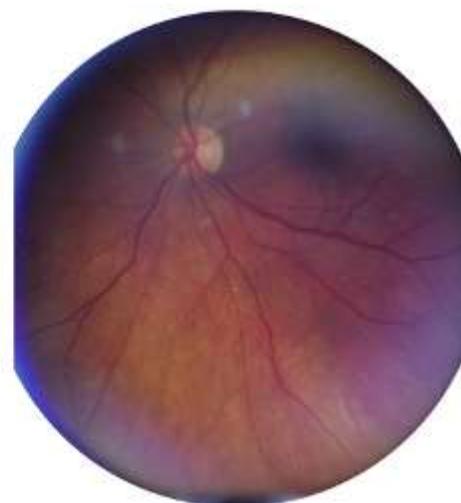
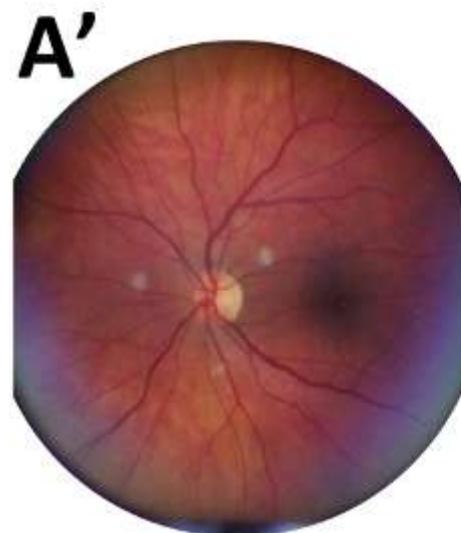
# Anwendungen der Bildregistrierung (Teil 1)

- **Registrierung** bringt bewegliche *Objektbilder (Templatebilder)* in Korrespondenz mit einem festen *Referenzbild*.  
Einsatzbereiche:
  - Korrektur von **Bewegungen** oder Lagerungsunterschieden bei wiederholten Aufnahmen
  - Direkter **Vergleich** verschiedener Patienten oder Zeitpunkte
    - Auf Basis der registrierten Bilder oder der benötigten Deformation



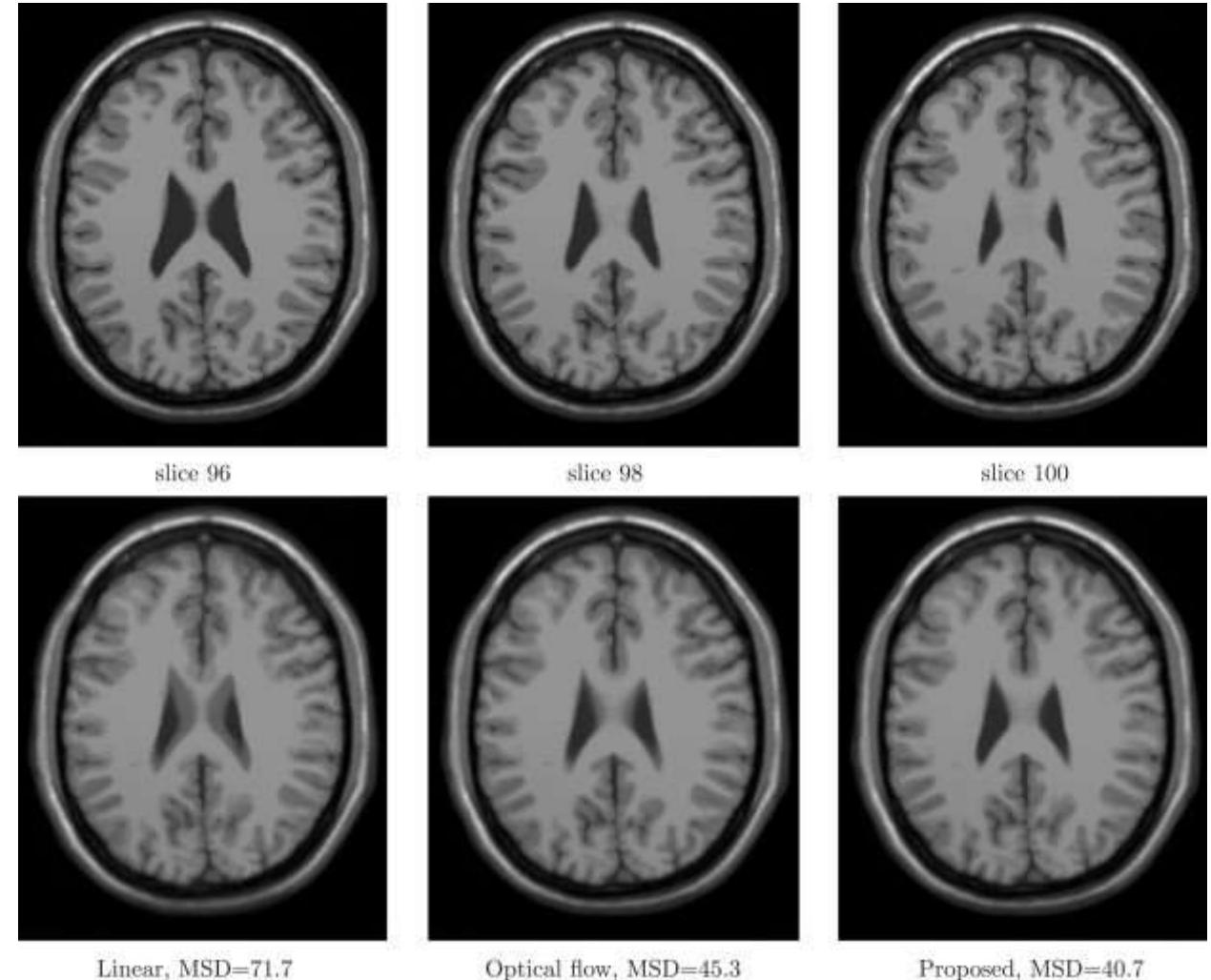
# Anwendungen der Bildregistrierung (Teil 2)

- Weitere Einsatzbereiche:
  - **Fusionierung** von Bildinhalten verschiedener Zeitpunkte oder komplementärer Modalitäten
  - **Montage** von Einzelbildern in Panoramas



# Anwendungen der Bildregistrierung (Teil 3)

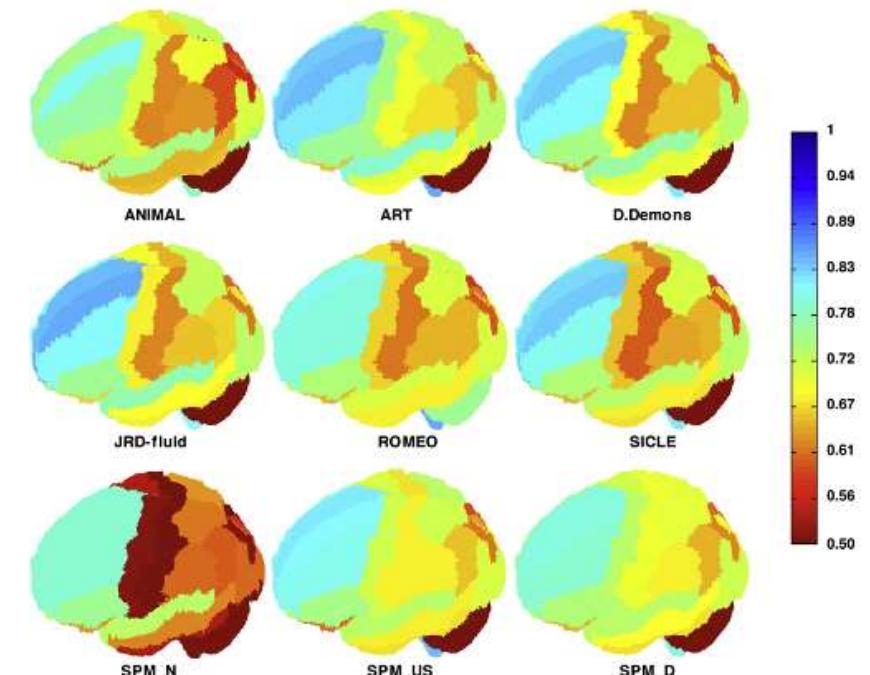
- Weiterer Einsatzbereich:
  - Strukturerhaltende **Interpolation** zwischen Bildern



# Registrierung vs. Segmentierung

In bestimmten Fällen besteht eine Verwandschaft zwischen Registrierungs- und Segmentierungsproblemen:

- **Registrierung via Segmentierung**
  - Segmentierung durch statistische Formmodelle bringt Stützpunkte in Korrespondenz
- **Segmentierung via Registrierung**
  - Registrierung auf ein korrekt segmentiertes Beispiel ermöglicht eine Übertragung der Labels



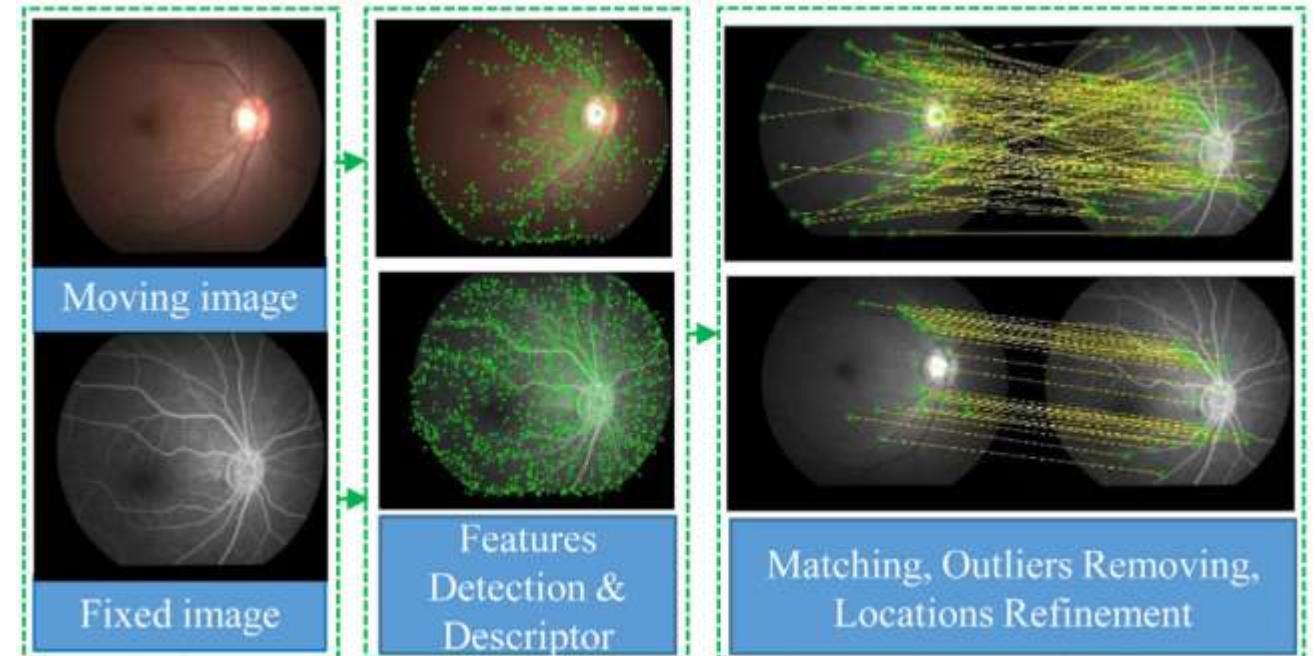
# Mono- vs Multimodale Registrierung

- **Monomodale (intramodale) Registrierung**
  - Annahme: Die korrekt registrierten Bilder haben im pixelweisen Vergleich ähnliche Intensitäten
  - Setzt i.d.R. voraus, dass die beteiligten Bilder vom selben bildgebenden Verfahren stammen und mit ähnlichen Parametern (z.B. T1 oder T2-Wichtung im MRT) aufgenommen wurden
- **Multimodale (intermodale) Registrierung**
  - Anatomische Strukturen sollen in Korrespondenz gebracht werden, obwohl sie i.d.R. unterschiedliche Kontraste erzeugen

# Zur Registrierung verwendete Information

Registrierungsverfahren unterscheiden sich in der **Art der verwendeten Bildinformation:**

- Manuell selektierte, durch Marker gegebene oder im Bild erkannte **Landmarken** (Punkte)
- **Kurven** oder **Oberflächen**
- **Voxel-** bzw. **intensitätsbasiert**



# Evaluierung mit vorgegebenen Landmarken

Sind für bestimmte **Landmarken** Positionen

- $\mathbf{r}_i$  ( $i = 1, \dots, m$ ) im Referenzbild
- $\mathbf{p}_i$  ( $i = 1, \dots, m$ ) im Objektbild

bekannt, quantifizieren der mittlere bzw. maximale *Target Registration Error*

$$\text{TRE}_{\text{mean}} = \frac{1}{m} \sum_{i=1}^m \|\mathbf{r}_i - T(\mathbf{p}_i)\|$$

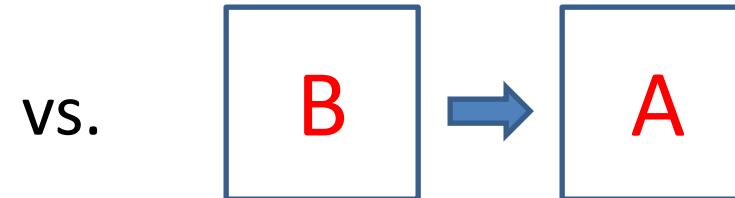
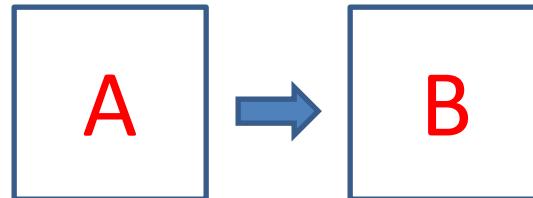
$$\text{TRE}_{\text{max}} = \max\{\|\mathbf{r}_i - T(\mathbf{p}_i)\| \mid i = 1, \dots, m\}$$

deren verbleibende Abweichung nach Anwendung der von der Registrierung bestimmten Transformation  $T$

- Mögliche Herkunft der Landmarken: Annotation durch Experten oder Anwendung einer bekannten Transformation zu Testzwecken

# Weitere wünschenswerte Eigenschaften

- **Vertauschbarkeit** von Referenz- und Objektbild
  - Sollte ungefähr die inverse Transformation ergeben



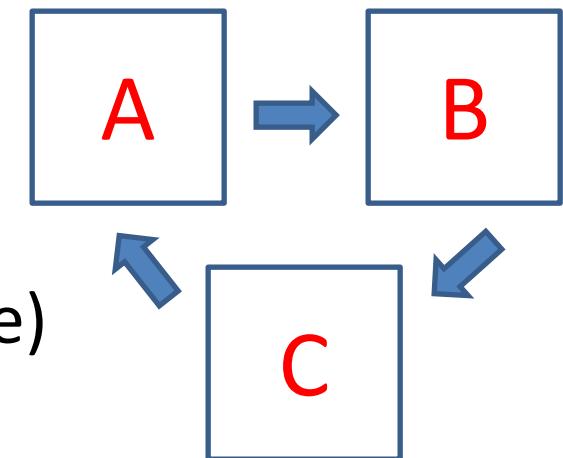
vs.

- **Zyklische Transformationen**

- Sollten ungefähr die Identität ergeben

- **Robustheit**

- Bildstörungen (z.B. künstliches Rauschen / Artefakte) sollten die geschätzte Transformation möglichst wenig verändern



# Evaluierung durch Visualisierung

- Zur **Einschätzung von Registrierungen** visualisiert man häufig
  - Differenz- / mittlere Bilder
  - Ausrichtung von Landmarken
  - Schachbrett-Muster
  - Separate Farbkanäle

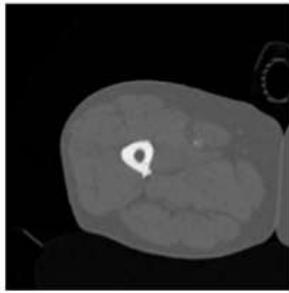


Fig 3.102: CT of a leg

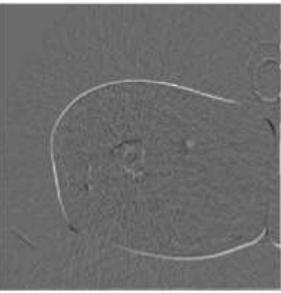


Fig 3.103: Pre-registration image subtraction

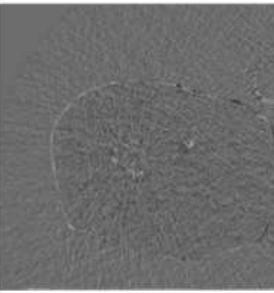
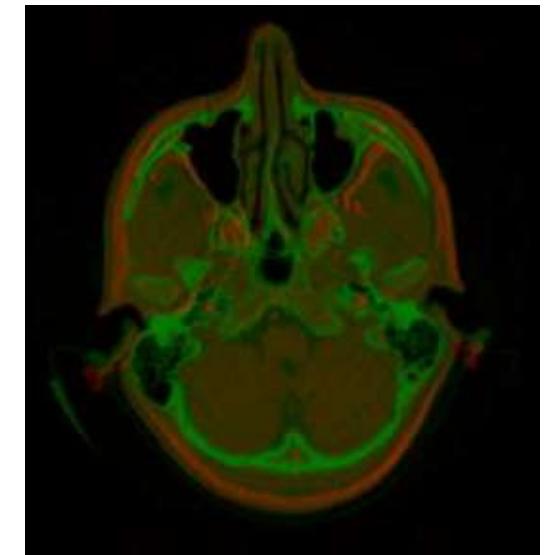
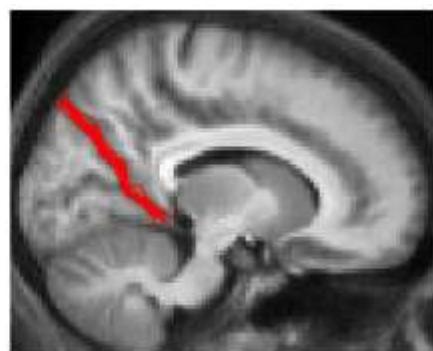
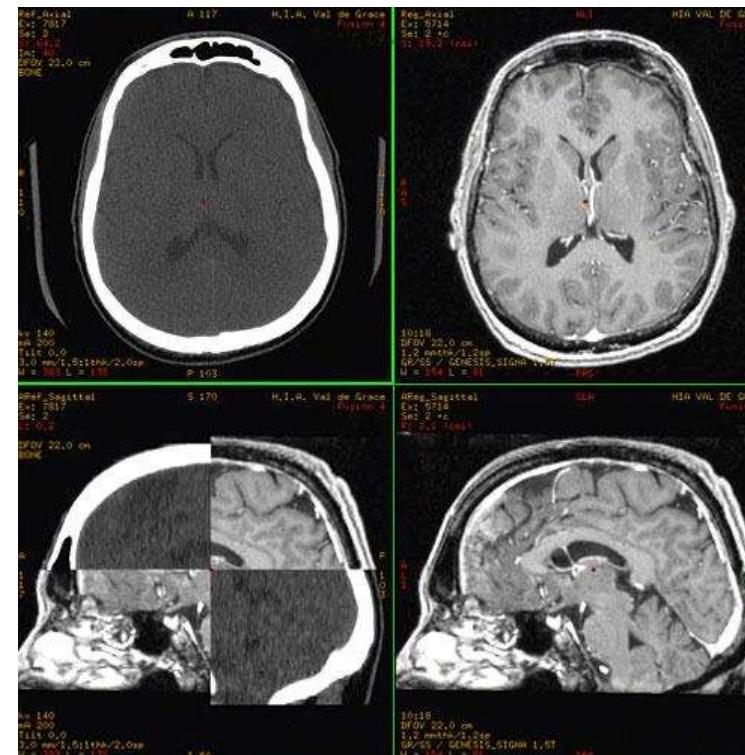


Fig 3.104: Post-registration image subtraction



# Bausteine eines typischen Registrierungsalgorithmus'

Registrierung erfordert in der Regel die Auswahl

- eines **Suchraums** erlaubter Transformationen
- eines **Interpolationsverfahren** um die Transformation anzuwenden
  - Siehe Kapitel 2
- einer **Kostenfunktion**, die Abweichungen der beiden Bilder oder ihrer relevanten Merkmale (z.B. Landmarken) quantifiziert
- eines **Optimierungsalgorithmus**, der die Kostenfunktion minimiert

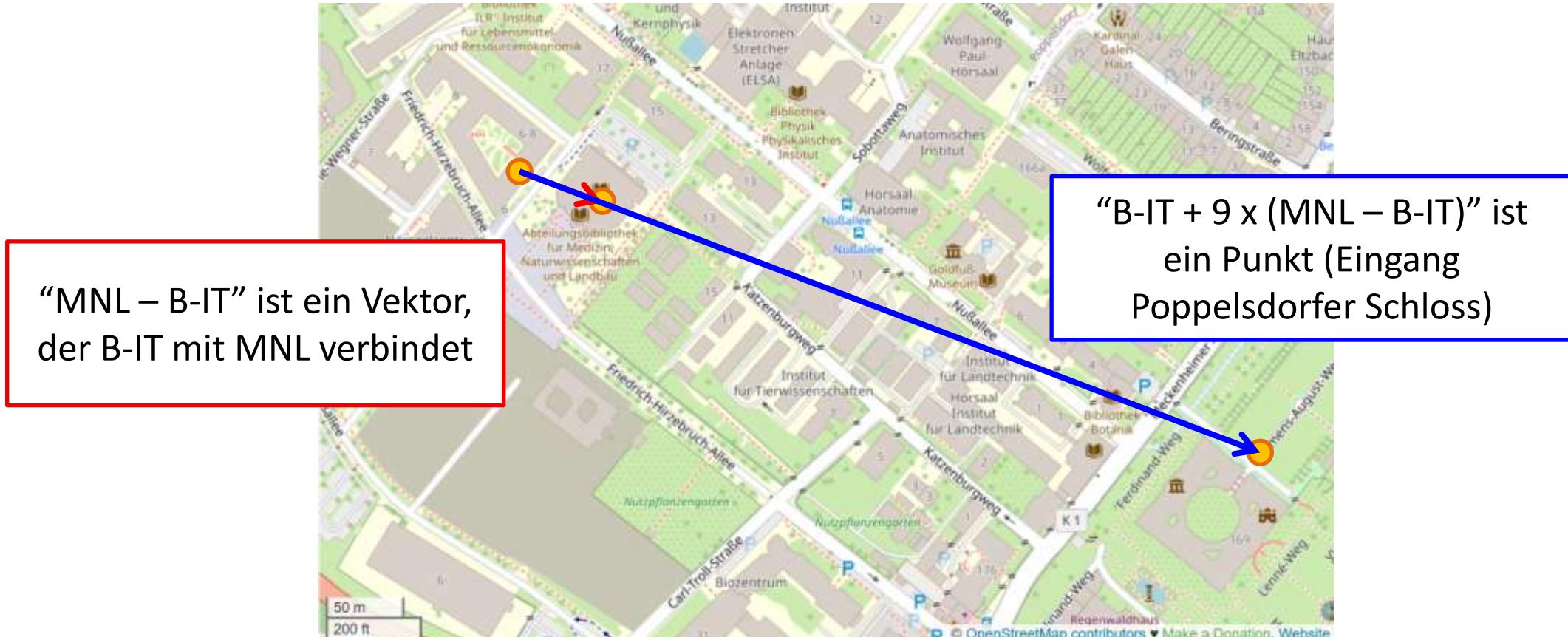
## **5.2 Lineare Bildtransformationen**

# In der Registrierung übliche Transformationen

- Je nach Art der angewandten Transformation unterscheiden wir
  - **Starre Registrierung** (*engl.* rigid registration) erlaubt nur Verschiebung und Rotation
  - **Affine Registrierung** (*engl.* affine registration) bildet parallele Linien auf parallele Linien ab
  - **Deformierbare Registrierung** (*engl.* deformable registration) erlaubt im Prinzip beliebige Deformationen, die jedoch in der Regel durch Regularisierung wieder eingeschränkt werden
    - z.B. keine gegenseitige Durchdringung von Organen, Begrenzung von Verzerrungen auf ein plausibles Maß

# Punkte

- **Punkte** in einem Bild bilden keinen sinnvollen Vektorraum
  - Was sollte “ $2 \times B\text{-IT}$ ” oder “ $MNL + B\text{-IT}$ ” sein?



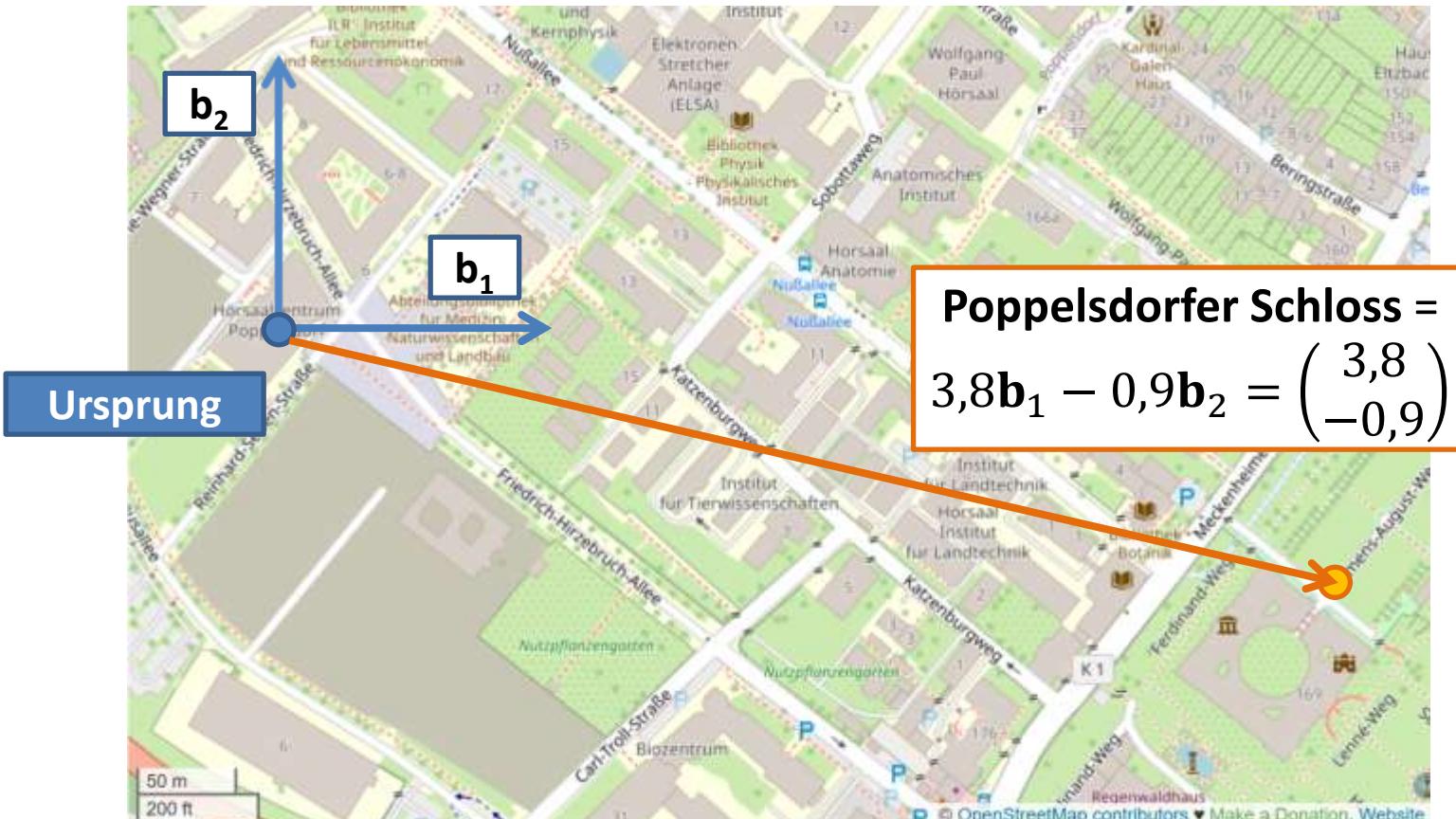
# Punkte als Vektoren

- Ein Referenzpunkt als **Ursprung** ermöglicht es mit Punkten zu arbeiten, als seien es Vektoren



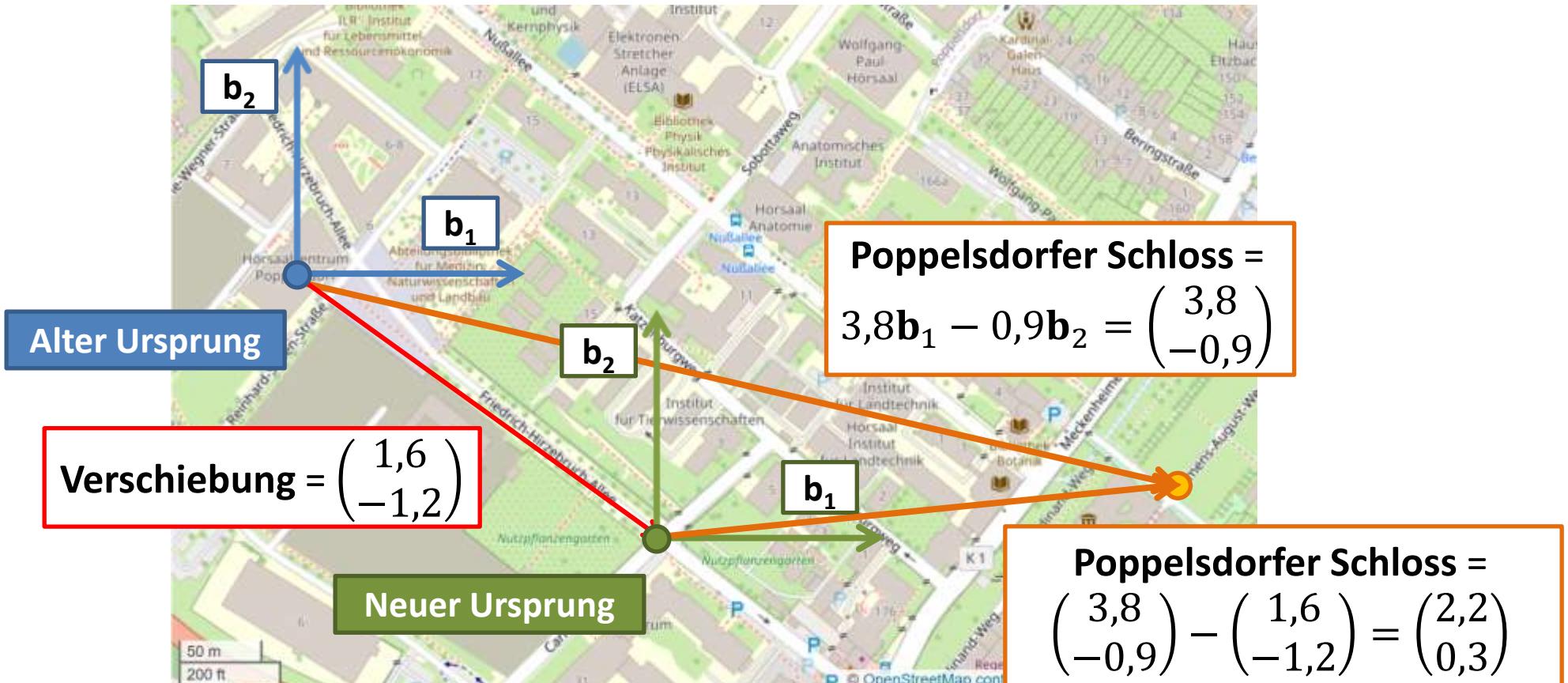
# Basen und Koordinaten

- Die Definition einer **Basis** ermöglicht die Darstellung von Vektoren durch Koeffizienten



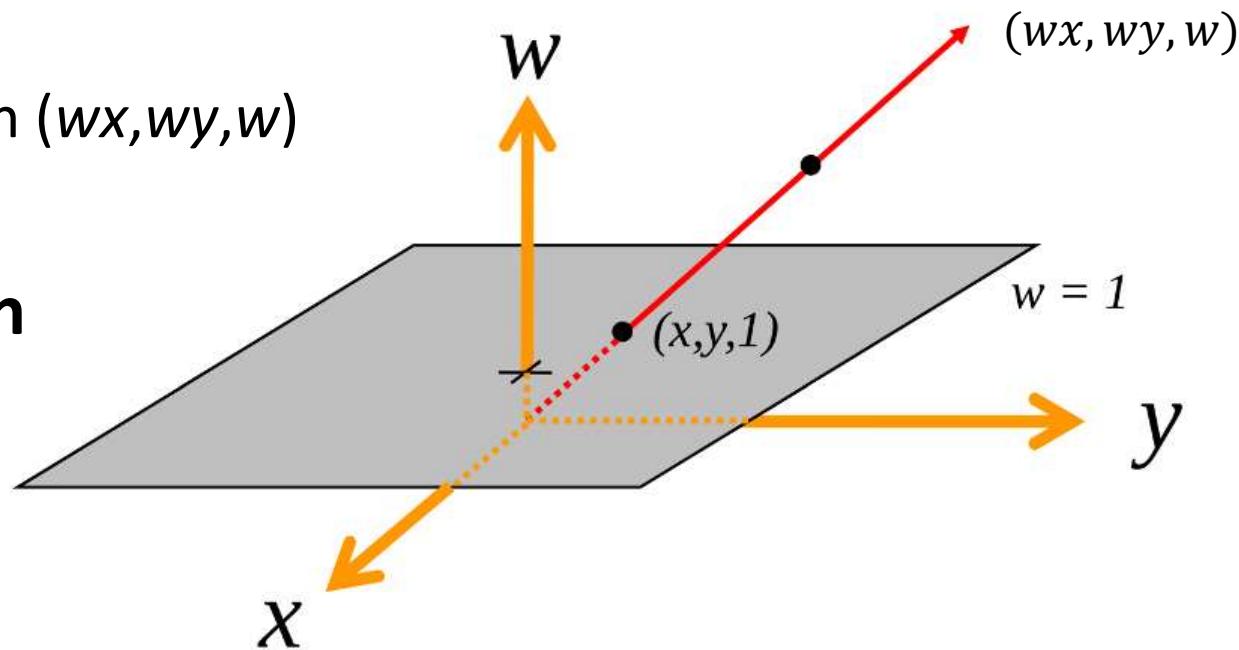
# Verschiebung des Ursprungs

- Eine **Verschiebung des Ursprungs** ändert die Koeffizienten von Punkten, nicht aber die von Vektoren zwischen Punkten



# Homogene Koordinaten

- **Homogene Koordinaten** ermöglichen uns die Unterscheidung von Punkten und Vektoren:
  - Repräsentation von  $n$ -dimensionalen **Punkten** durch Geraden in einem  $(n+1)$ -dimensionalen Raum:
    - Nenne die neue Koordinate  $w$
    - Repräsentiere den Punkt  $(x,y)$  durch  $(wx,wy,w)$
    - Kanonische Repräsentation:  $(x,y,1)$
  - Repräsentiere **Vektoren zwischen Punkten** mit  $w=0$



# Translation in Homogenen Koordinaten

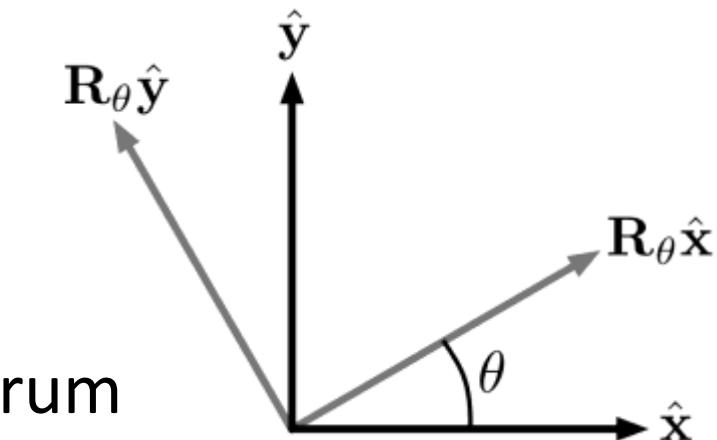
- **Translation** (Parallelverschiebung)  $f(\mathbf{x}) = \mathbf{x} + \mathbf{t}$  ist eine der grundlegendsten Operationen bei der Registrierung
  - Im ursprünglichen  $n$ -D-Raum ist sie jedoch nichtlinear. Für  $\mathbf{t} \neq 0$  ist
    - $f(\mathbf{x} + \mathbf{y}) = \mathbf{x} + \mathbf{y} + \mathbf{t} \neq f(\mathbf{x}) + f(\mathbf{y}) = \mathbf{x} + \mathbf{y} + 2\mathbf{t}$
    - $f(\alpha\mathbf{x}) = \alpha\mathbf{x} + \mathbf{t} \neq \alpha f(\mathbf{x}) = \alpha\mathbf{x} + \alpha\mathbf{t}$
- **Homogene Koordinaten** ermöglichen Translationen mittels Matrix-Vektor-Multiplikation:

$$f(\mathbf{x}) = \mathbf{x} + \mathbf{t} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_x \\ x_y \\ 1 \end{pmatrix}$$

- *Hinweis:* In Übereinstimmung mit der Anschauung bleiben Vektoren ( $w = 0$ ) von Translationen unberührt

# Rotationen: Grundidee

- **Rotationen** wirken immer in einer Ebene, der Orthogonalraum bleibt unberührt
  - in 2D: Rotation des ganzen Bildes
  - in 3D: Rotation um eine Rotationsachse
  - Rotation um den Ursprung. Vorgeschaltete Translation ermöglicht alternatives Rotationszentrum
- Skizze zeigt Auswirkung der Rotation um  $\theta$  in der von orthonormalen Vektoren  $\hat{\mathbf{x}}$  und  $\hat{\mathbf{y}}$  aufgespannten Ebene
  - *Konvention:* Positive  $\theta$  rotieren  $\hat{\mathbf{x}}$  auf  $\hat{\mathbf{y}}$  zu



$$\mathbf{R}_\theta \hat{\mathbf{x}} = \cos \theta \hat{\mathbf{x}} + \sin \theta \hat{\mathbf{y}}$$

$$\mathbf{R}_\theta \hat{\mathbf{y}} = -\sin \theta \hat{\mathbf{x}} + \cos \theta \hat{\mathbf{y}}$$

$$\mathbf{R}_\theta \mathbf{a} = \mathbf{a}.$$

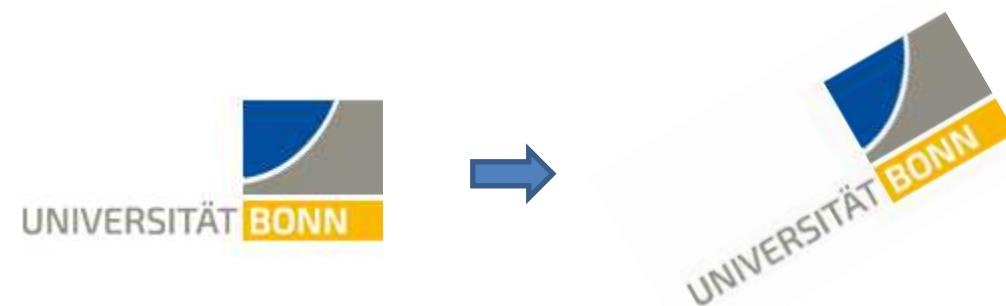
# Rotationen: Matrix-Schreibweise

- Aufgrund der Überlegungen der vorherigen Folie

$\mathbf{R}_\theta \hat{\mathbf{x}} = \cos \theta \hat{\mathbf{x}} + \sin \theta \hat{\mathbf{y}}$ ;  $\mathbf{R}_\theta \hat{\mathbf{y}} = -\sin \theta \hat{\mathbf{x}} + \cos \theta \hat{\mathbf{y}}$ ;  $\mathbf{R}_\theta \mathbf{a} = \mathbf{a}$   
erhalten wir:

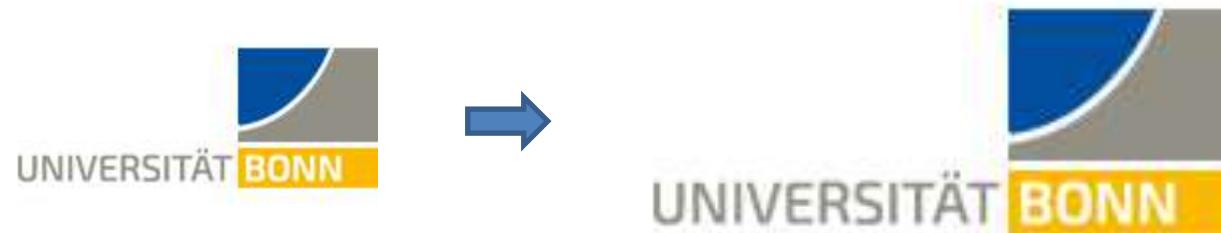
$$\mathbf{R}_\theta = \mathbf{I} + (\cos \theta - 1)(\hat{\mathbf{x}}\hat{\mathbf{x}}^T + \hat{\mathbf{y}}\hat{\mathbf{y}}^T) + \sin \theta (\hat{\mathbf{y}}\hat{\mathbf{x}}^T - \hat{\mathbf{x}}\hat{\mathbf{y}}^T)$$

- Beispiel mit  $\hat{\mathbf{x}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ ,  $\hat{\mathbf{y}} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ :  $\mathbf{R}_\theta = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$



# Ähnlichkeitsabbildungen

- Die skalierte Einheitsmatrix  $\alpha \mathbf{I}$  entspricht einer **uniformen Skalierung** um den Faktor  $\alpha$ 
  - Ähnlichkeitsabbildungen  $f(\mathbf{x}) = \alpha \mathbf{R}\mathbf{x} + \mathbf{t}$  erhalten Formen, d.h. Winkel und relative Abstände
  - In der Registrierung ermöglichen sie z.B. den Ausgleich von Auflösungsunterschieden



# Affine Abbildungen

- **Affine Abbildungen** haben die Form  $f(x)=Mx+t$ 
  - Die Matrix **M** darf beliebig sein
    - *Zusätzlich:* Projektion, nicht-uniforme Skalierung, Spiegelung, Scherung
  - Bildet Geraden auf Geraden oder Punkte ab, Parallelität bleibt erhalten, Teilverhältnisse auf Geraden bleiben erhalten
- Homogene Koordinaten ermöglichen die Schreibweise

$$\begin{pmatrix} a & b & c & tx \\ d & e & f & ty \\ g & h & i & tz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The matrix is shown with curly braces underlining the first three columns and the fourth column. Above the matrix, the label 'M' is connected by a curved arrow to the first three columns, and the label 't' is connected by a curved arrow to the fourth column.

- „Affine“ und „lineare“ Registrierung werden häufig synonym verwendet

# Orthogonalprojektionen

- **Projektionen** spielen in der Registrierung in der Regel nur eine mittelbare Rolle als Bausteine anderer Transformationen
- **Orthogonalprojektion** auf den von  $\mathbf{x}$  aufgespannten Unterraum:

$$\mathbf{P}_x = \frac{\mathbf{x}\mathbf{x}^T}{\mathbf{x}^T\mathbf{x}}$$

$$\mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{P}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- Orthogonalprojektion auf den **Komplementärraum**:  $\mathbf{P}_x^\perp = \mathbf{I} - \mathbf{P}_x$

$$\mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{P}_x^\perp = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

# Nicht-uniforme Skalierung

- **Nicht-uniforme Skalierung** um den Faktor  $\alpha$  entlang der Richtung  $\mathbf{x}$ 
  - basiert auf der Zerlegung

$$\mathbf{v} = \mathbf{I}\mathbf{v} = \mathbf{P}_x\mathbf{v} + \mathbf{P}_x^\perp\mathbf{v}$$

- Resultierende Matrix:

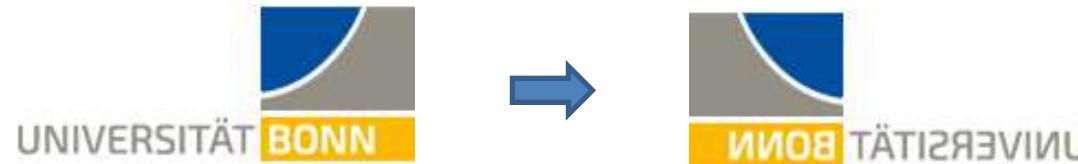
$$\mathbf{S}_{\alpha,x} = \alpha\mathbf{P}_x + \mathbf{P}_x^\perp$$

- In der Registrierung ermöglicht sie z.B. den Ausgleich von richtungsabhängiger (anisotroper) Auflösung



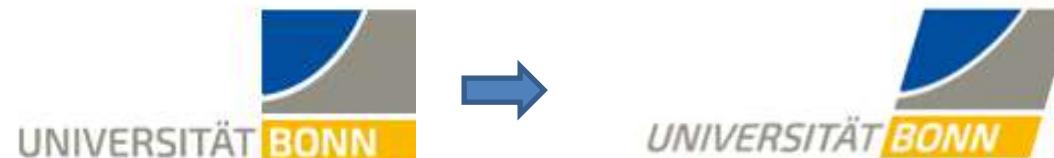
# Spiegelungen und Scherungen

- **Spiegelung** entspricht einer nicht-uniformen Skalierung mit Faktor  $\alpha = -1$ :  $M_{\hat{x}} = P_{\hat{x}}^\perp - P_{\hat{x}}$



- **Scherung (Transvektion)**: Verschiebung parallel zu einer festen Achse, proportional (mit Scherfaktor  $\beta$ ) zur Position entlang einer orthogonalen Achse

$$T = I + \beta \hat{x} \hat{y}^T$$



# Berechnung der Transformationen aus Korrespondenzen

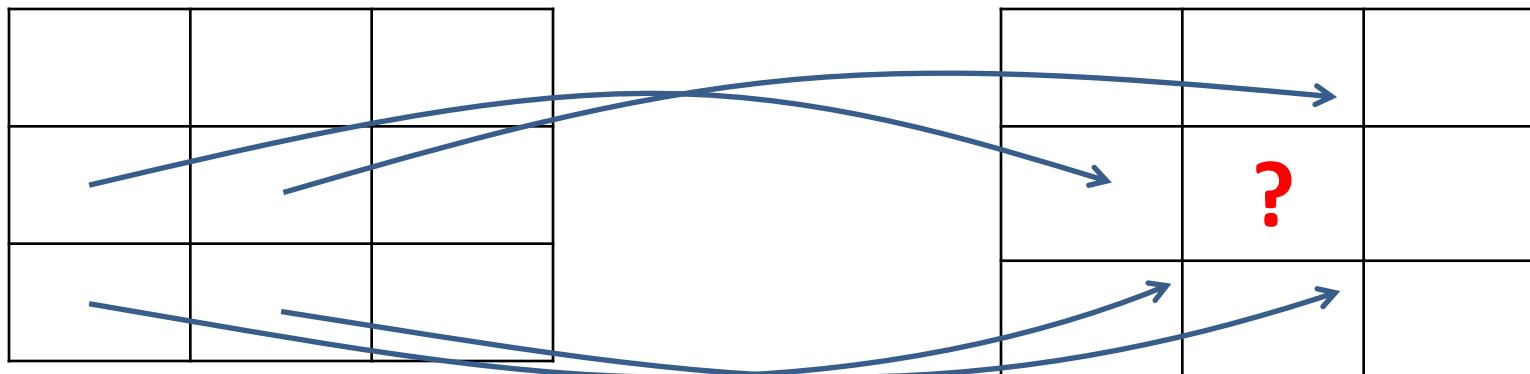
- Seien wieder  $\mathbf{p}_i$  ( $i = 1, \dots, m$ ) Punkte im Objektbild,  $\mathbf{r}_i$  ( $i = 1, \dots, m$ ) korrespondierende Punkte im Referenzbild
- Gesucht sei Transformation  $T$  die zu kleinsten Quadratsummen führt:

$$T = \arg \min_{T'} \sum_{i=1}^m \|T'(\mathbf{p}_i) - \mathbf{r}_i\|^2$$

- **Starre oder Ähnlichkeits-Transformationen**  $T$  erhalten wir mittels Prokrustes-Analyse (Kapitel 4.5)
- **Affine Transformationen**  $T$  ergeben sich durch lineares Ausgleichsproblem mit den Koeffizienten der Transformationsmatrix als Unbekannten (Übung)

# Nachteil der Vorwärts-Transformation

- Anwendung der **Vorwärts-Transformation**  $T$  erzeugt das transformierte Bild durch Iteration über das Ursprungsbild
  - Manche Pixel des erzeugten Bildes erhalten mehrere Farbwerte, andere keins
  - Hinterlässt ohne weiteres Auffüllen „Löcher“ im Resultat



# Lösung: Nutzen der Rückwärts-Transformation

- Berechnung der inversen (**Rückwärts**)-Transformation  $T^{-1}$  ermöglicht Berechnung des transformierten Bildes durch Iteration über die Ausgabepixel
  - Jedes Pixel erhält genau einen Farbwert
  - Interpolation im Ursprungsbild mit üblichen Verfahren
  - Bei affinen Abbildungen: Berechnung von  $T^{-1}$  als inverse Matrix



# Bild- vs. Weltkoordinaten

Bei der Arbeit mit medizinischen Bildern unterscheiden wir zwischen

- **Bildkoordinaten:** Indizieren Pixel/Voxel
  - Bild als Matrix: rc-Koordinaten (row/column)
    - Vertikale Position zuerst, Ursprung links oben, größere Zeilen weiter unten
  - Bild als Funktion: xy-Koordinaten
    - Horizontale Position zuerst, Ursprung links unten, größeres y weiter oben
- **Weltkoordinaten:** Physisches Koordinatensystem (XYZ), in dem der Patient verortet ist. Bei uniformen Voxelgittern:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X_0 \\ Y_0 \\ Z_0 \end{pmatrix} + x \begin{pmatrix} X_x \\ Y_x \\ Z_x \end{pmatrix} + y \begin{pmatrix} X_y \\ Y_y \\ Z_y \end{pmatrix} + z \begin{pmatrix} X_z \\ Y_z \\ Z_z \end{pmatrix}$$

# Bild- zu Weltkoordinaten als Affine Abbildung

- In homogenen Koordinaten lässt sich der Zusammenhang zwischen Bild- und Weltkoordinaten durch eine Matrix beschreiben:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} X_x & X_y & X_z & X_0 \\ Y_x & Y_y & Y_z & Y_0 \\ Z_x & Z_y & Z_z & Z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- Bilder aus Hybridgeräten (z.B. PET/CT) sind häufig intrinsisch registriert. Korrespondenzen sind durch die Bild-zu-Welt-Transformationen gegeben bzw. können durch Matrix-Invertierung und –Multiplikation leicht berechnet werden

# Zusammenfassung

- Anwendung „**linearer Bildtransformationen**“ geht von der Nutzung **homogener Koordinaten** aus
  - Hierdurch werden auch Translationen abgedeckt
  - **Starre Abbildungen:** Translation und Rotation
  - **Ähnlichkeits-Abbildungen:** Starr+uniforme Skalierung
  - **Affine Abbildungen:** Ähnlichkeit+nicht uniforme Skalierung, Scherung, Spiegelung
- Erzeugung registrierter Bilder durch **Rückwärts-Transformation**
- Unterscheidung von **Bild- und Weltkoordinaten**

## **5.3 Registrierung als Optimierungsproblem**

# Registrierung als Optimierungsproblem

- Gegeben seien
  - Referenzbild  $y$
  - Objektbild  $x$
  - Kostenfunktion  $C$
  - Suchraum  $S_T$  von Transformationen, parametrisiert durch  $w$
- Folgendes Optimierungsproblem bestimmt die Parameter der optimalen Transformation  $T$ :

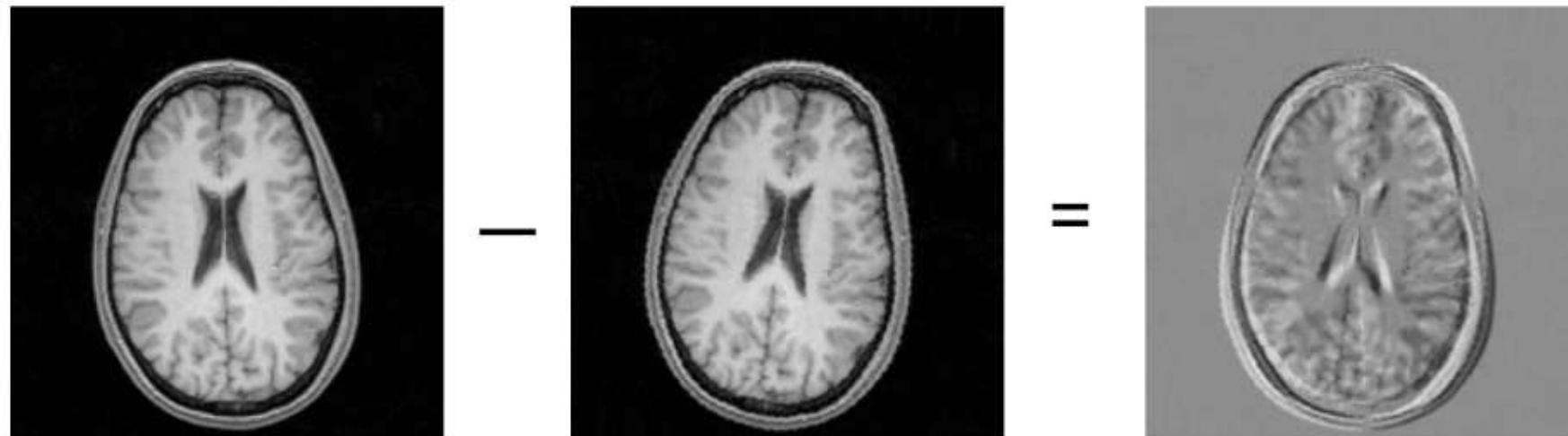
$$w = \arg \min_{w'} C(y, T(x|w'))$$

# Kleinste Quadrate (L2-Norm) als Kostenfunktion

- Eine einfache Kostenfunktion zur voxelbasierten Registrierung ist durch die **mittlere quadratische Abweichung** über die N überlappenden Pixel gegeben:

$$C = \frac{1}{N} \sum_{i=1}^N (x_i - y_i)^2$$

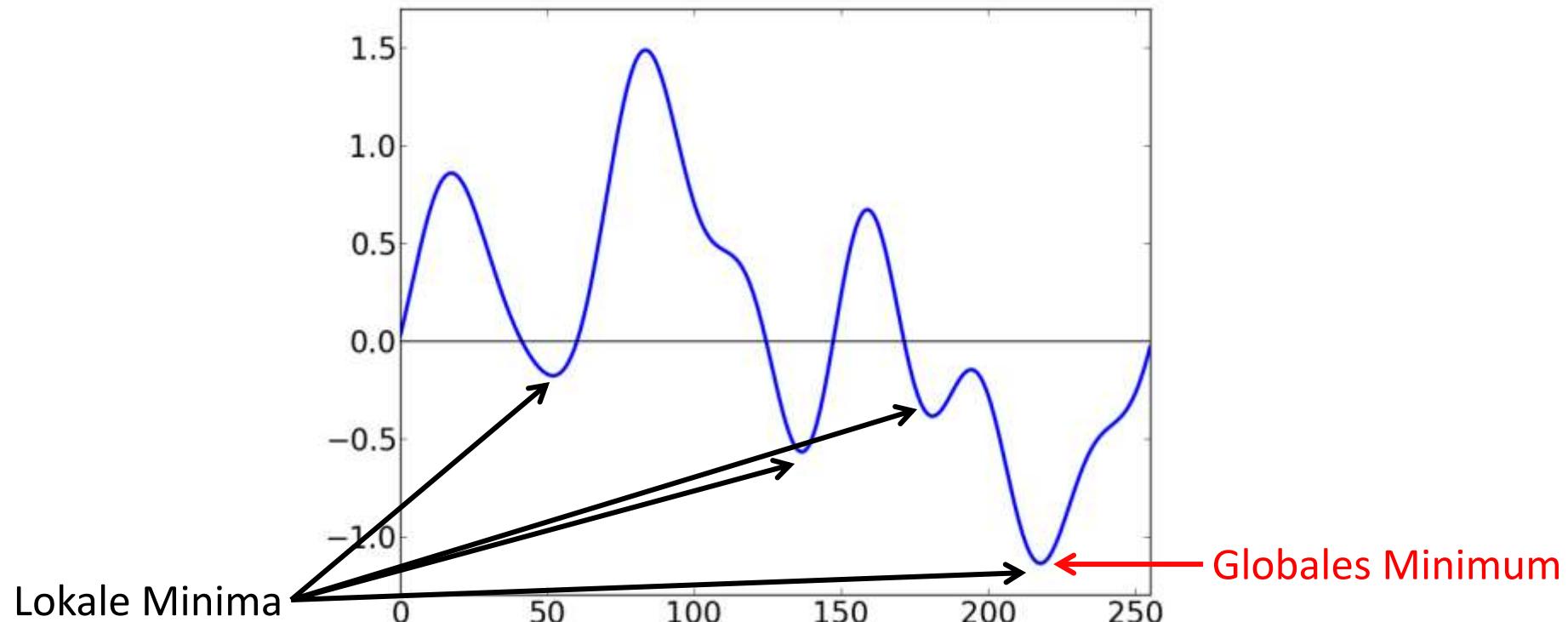
- *Quiz:* Was ist eine Einschränkung dieser Kostenfunktion?



Bildquelle: Oxford FMRIB

# Ziel von Optimierungsverfahren

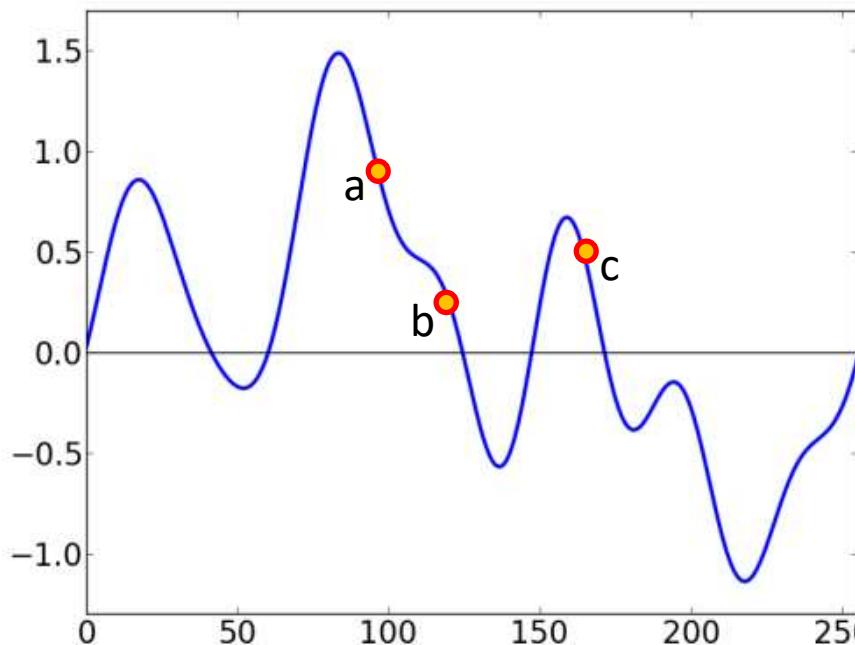
- *Angestrebt:* algorithmische Minimierung einer Funktion  $f(x)$ 
  - Annahme:  $f(x)$  ist stetig



# 1D-Fall: „Einfangen“ von Minima durch Intervalle

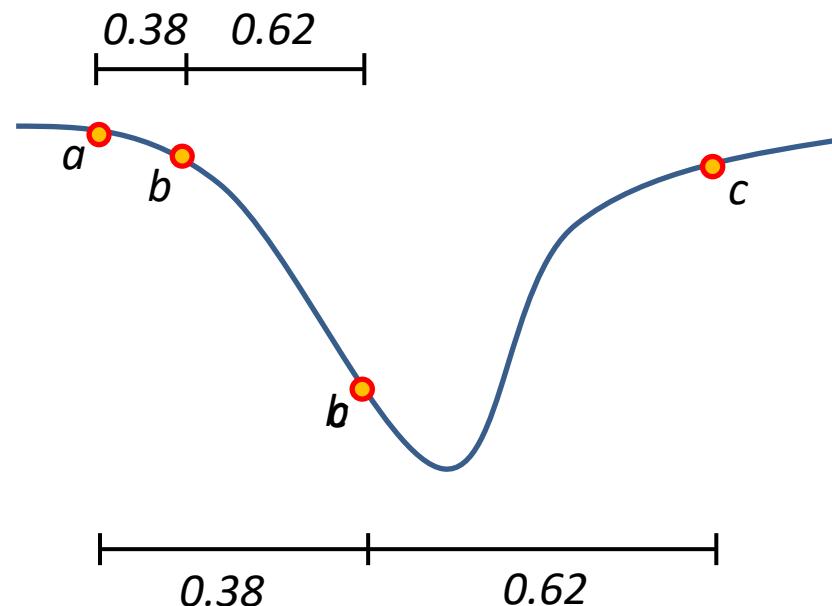
- Eine kontinuierliche Funktion  $f(x)$  hat im Intervall  $(a,c)$  mindestens ein lokales Minimum, falls es darin einen inneren Punkt  $b$  gibt, für den gilt:

$$f(a) > f(b) \text{ und } f(c) > f(b)$$



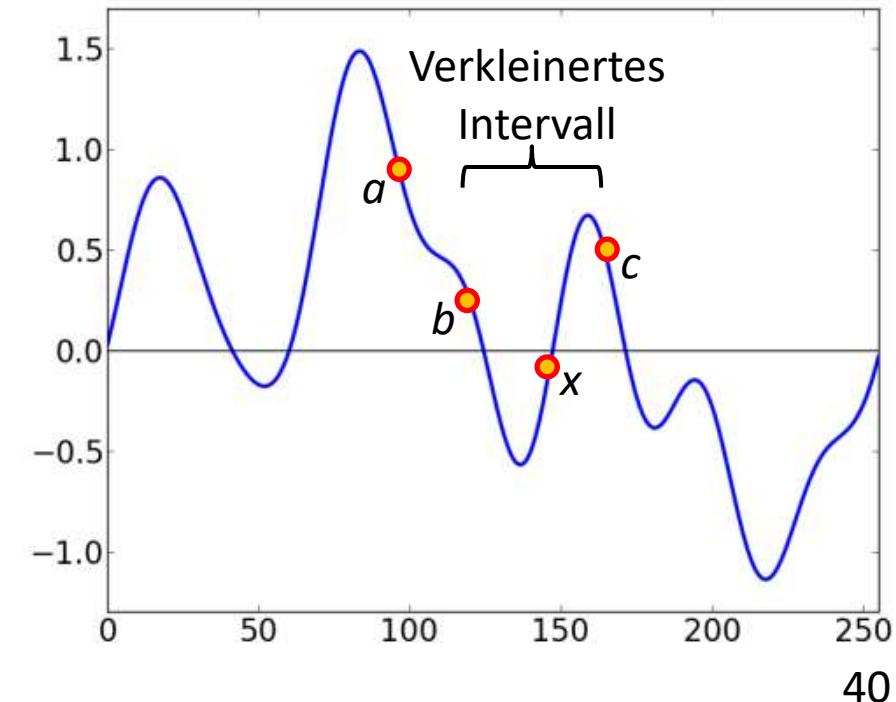
# Finden eines initialen Intervalls

- Ausgehend von einem ersten Intervall  $(a,b)$  mit  $f(a) > f(b)$  vergrößern wir es schrittweise über den kleineren Wert hinaus, bis die Bedingung der letzten Folie erfüllt ist
  - Minimierung scheitert, wenn das Intervall zu groß wird oder die Grenze der zulässigen Werte erreicht



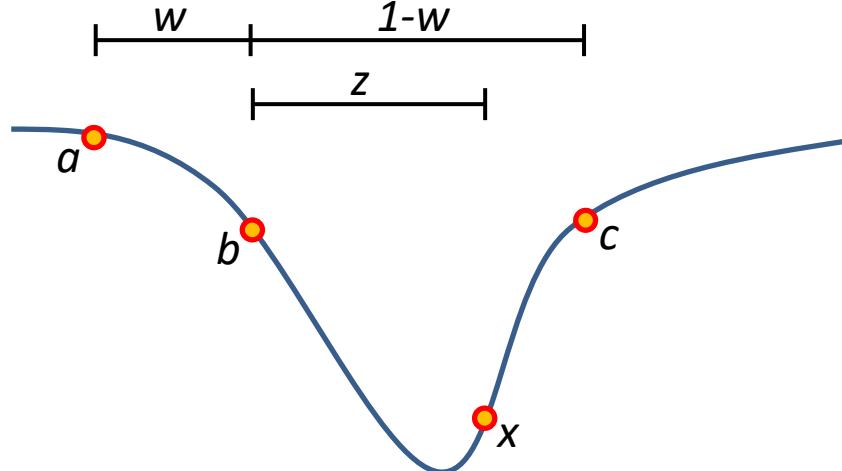
# Iterative Verkleinerung des Intervalls

- **Grundidee:** Werte  $f(x)$  im längeren der beiden Intervalle  $(a, b)$  und  $(b, c)$  aus. O.B.d.A. sei dies  $(b, c)$ 
  - Falls  $f(x) < f(b)$  befindet sich mindestens ein lokales Minimum in  $(b, c)$
  - Falls  $f(x) > f(b)$  befindet sich mindestens ein lokales Maximum in  $(a, x)$
  - In jedem Fall haben wir das Intervall verkleinert. Benenne die neuen drei Punkte in  $(a, b, c)$  um und iteriere so lange, bis  $|c-a| < \theta$



# An welchem Punkt verfeinern?

- **Frage:** Welcher neue Punkt  $x$  führt für vorgegebene  $(a, b, c)$  im nächsten Schritt zum kürzesten Intervall?



$$w = \frac{b - a}{c - a} \quad z = \frac{x - b}{c - a}$$

Abhängig von  $f(x)$  ist das neue Intervall  $1-w$  oder  $w+z$  breit. Um in beiden Fällen denselben Fortschritt zu erzielen setzen wir:

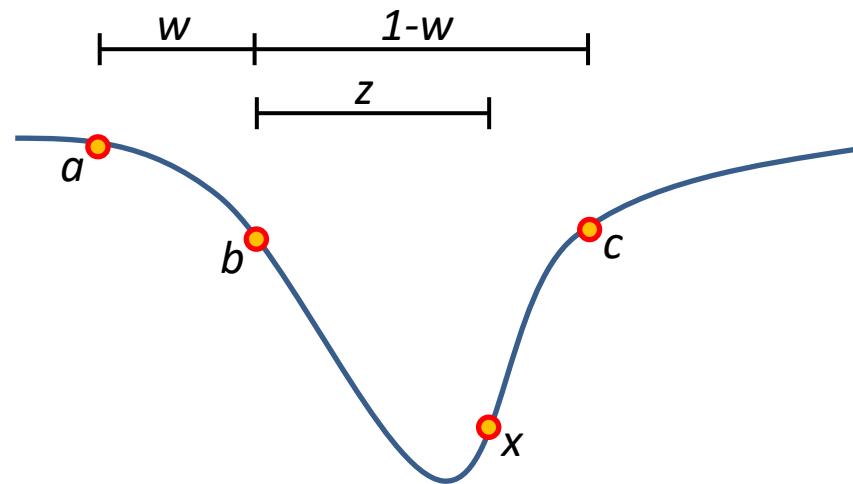
$$1 - w = w + z$$

$$\Rightarrow z = 1 - 2w \quad (\text{Gl. 1})$$

- **Antwort:**  $x$  sollte von  $c$  denselben Abstand haben wie  $b$  von  $a$ .

# Wo liegt der optimale innere Punkt?

- **Frage:** Mit welchem inneren Punkt  $b$  macht die Regel auf der letzten Folie für gegebenes  $(a,c)$  konstanten Fortschritt?



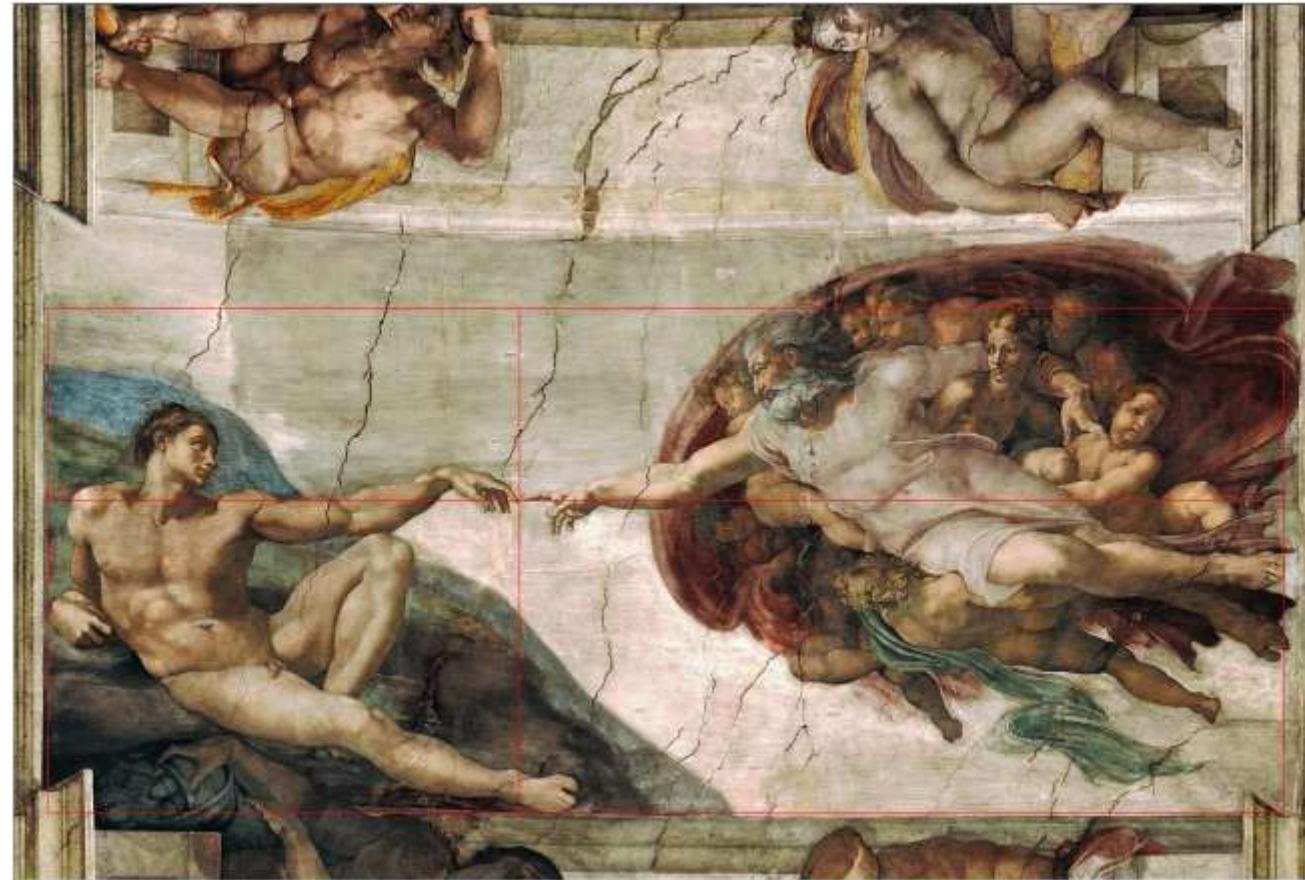
**Idee:** Um konstanten Fortschritt zu machen, muss die relative Position von  $x$  im neuen Intervall der von  $b$  im ursprünglichen entsprechen

$$\frac{z}{1-w} = w \quad (\text{Gl. 2})$$

- Kombination mit Gl. 1 ( $z = 1 - 2w$ ) ergibt:  $w^2 - 3w + 1 = 0$
- Die einzige Lösung in  $(0,1)$  ist  $w = \frac{3-\sqrt{5}}{2} \approx 0.38197$
- **Antwort:**  $b$  sollte dem „goldenen Schnitt“ von  $(a,c)$  entsprechen

# Randbemerkung: Der goldene Schnitt in der Kunst

In der Renaissance war der „goldene Schnitt“  $\phi := \frac{1}{1-w} = \frac{1-w}{w} = \frac{1+\sqrt{5}}{2}$  auch als „göttliche Proportion“ bekannt



# Algorithmus: Suche mit dem goldenen Schnitt

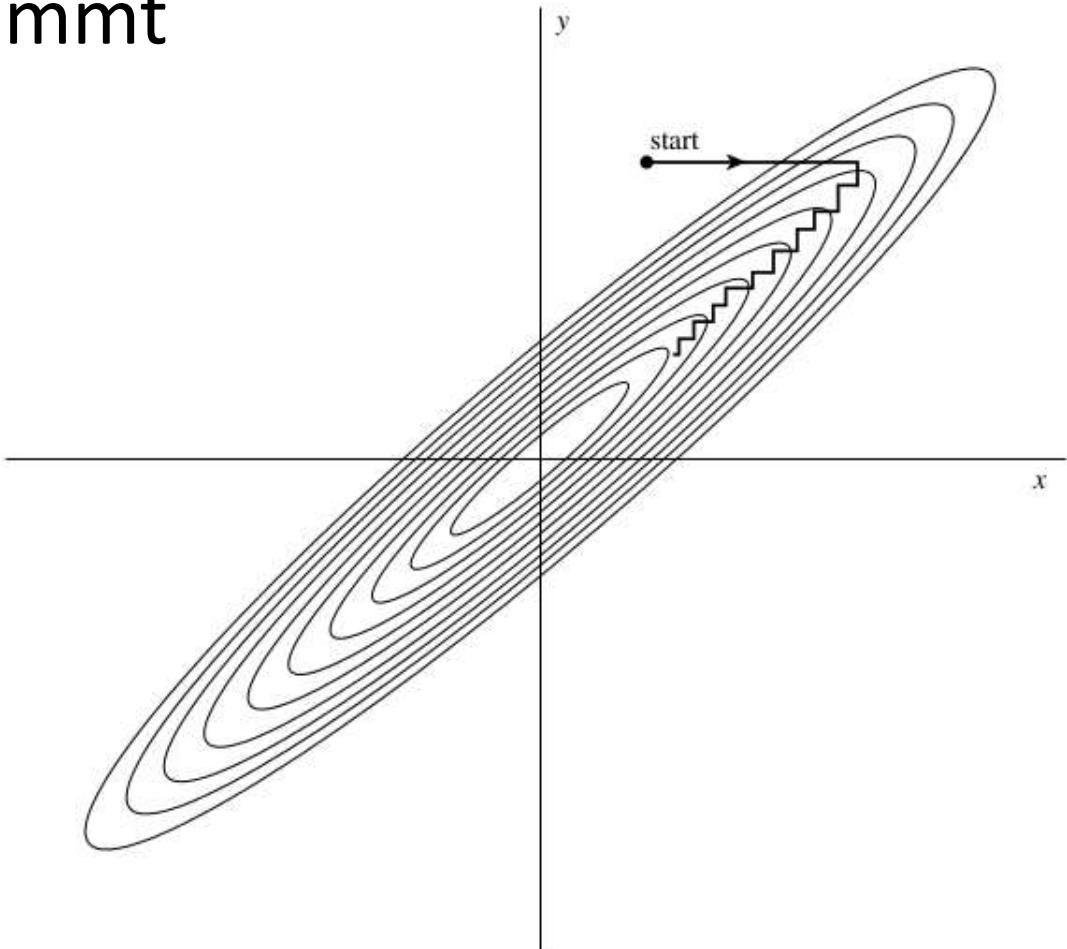
- **Gegeben:** Intervall  $(a,c)$  mit innerem Punkt  $b$
- **Iteration:**
  - Berechne  $f(x)$  für einen Punkt  $x$  der das größere der Intervalle  $(a,b)$  oder  $(b,c)$  im Verhältnis 0.38:0.62 teilt (zu  $b$  hin)
  - Wähle in Abhängigkeit von  $f(x)$  das linke oder rechte Intervall
    - Das, wo der innere Punkt unter den Rändern liegt
- **Beobachtung:** Selbst wenn  $b$  ursprünglich  $(a,c)$  nicht im goldenen Schnitt teilt, erzeugt diese Vorgehensweise i.d.R. innerhalb weniger Iterationen Tripel im goldenen Schnitt
  - Ab diesem Punkt verkürzt jede Iteration das Intervall um den Faktor 0.62
  - „Lineare Konvergenz“: Zahl der korrekten Nachkommastellen wächst linear mit der Zahl der Iterationen

# Optimierung in höheren Dimensionen

- Lineare Registrierung in 3D hat einen 6- (starr) bis 12-dimensionalen (affin) Suchraum
  - Die Suche mit dem goldenen Schnitt lässt sich nicht in höhere Dimensionen verallgemeinern
  - *Statt dessen:* Alternierende 1D-Optimierungen entlang einer Menge von Suchrichtungen
    - *Beispiel:* N Basis-Vektoren des N-D-Suchraums
    - *In der Registrierung:* Translation, Rotation, Skalierung, Scherung
  - Iteriere wiederholt über die  $N$  Richtungen, bis zur Konvergenz
  - Funktioniert dann gut, wenn Richtungen „entkoppelt“ sind

# Ein Problem für alternierende 1D-Ansätze

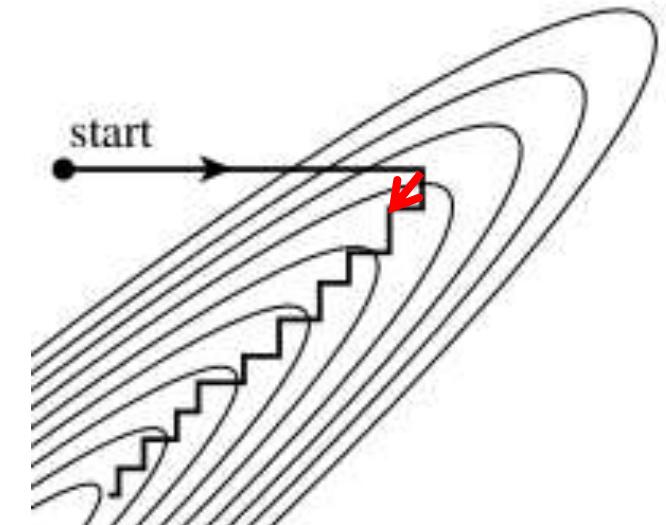
- **Problem:** Alternierende 1D-Optimierung wird sehr langsam, wenn die Zielfunktion ein schmales Tal aufweist, das mit den Suchrichtungen nicht übereinstimmt
  - Zwingt die Methode zu einer großen Zahl kleiner Schritte



# Verfahren von Powell

- **Grundidee:**

- Iteriere vom Startpunkt  $P_0$  aus  $N$  1D-Optimierungen
- Jede davon ende in einem neuen Punkt  $P_i$
- Nutze nach  $N$  1D-Optimierungen  $P_N - P_0$  als neue Suchrichtung
  - Richtung des Gesamtfortschritts
  - Hoffnung: „weist entlang des Tals“



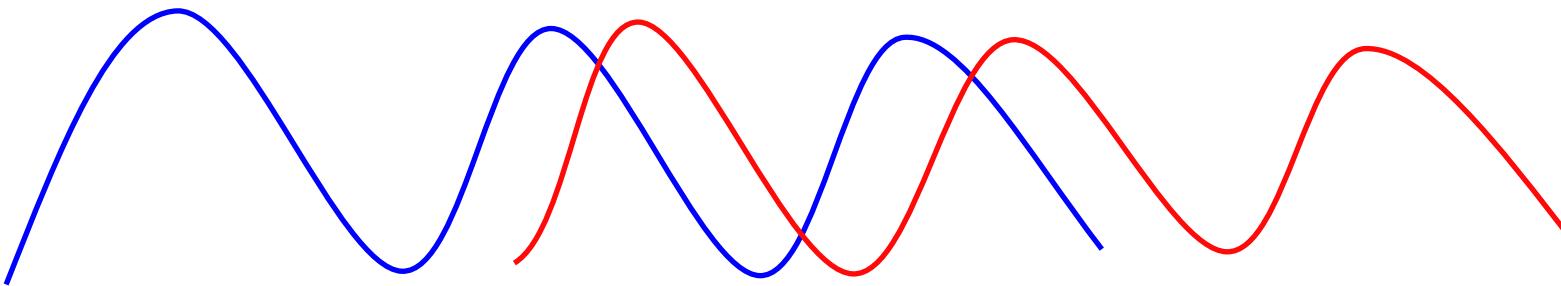
- **Varianten:**

- Ersetze die bislang erste Suchrichtung durch die neue Richtung
  - *Aber:* Richtungen können degenerieren (lineare Abhangigkeit)
- Ersetze die Suchrichtung mit dem groten Beitrag
  - *Idee:* Ersetzen einer moglich kollinearen Richtung
- Periodische Orthogonalisierung der Suchrichtungen

# Lokale und Globale Optimierung

- **Lokale Optimierung:**

- Findet ein lokales Minimum in der Nachbarschaft des Punkts  $x$
- *Aber:* Suboptima sind aufgrund von Bildinhalten oft unvermeidlich



- **Globale Optimierung:**

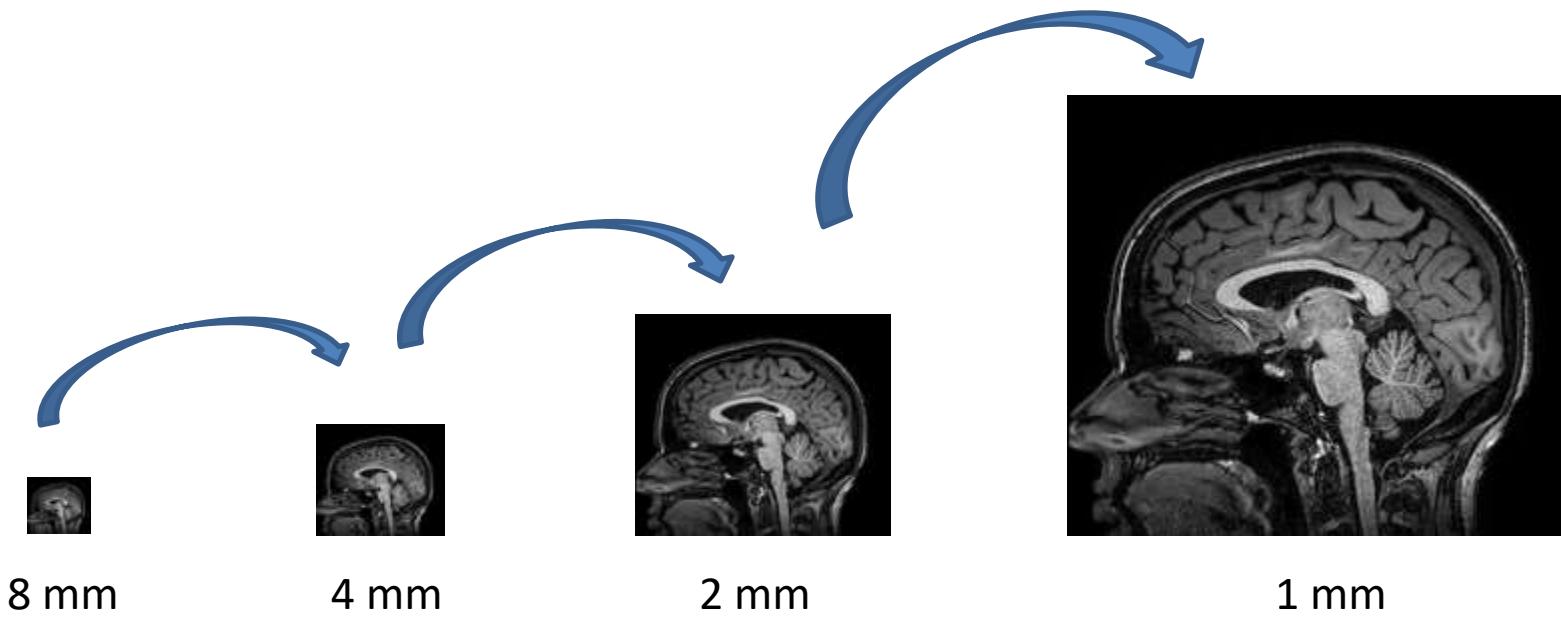
- Ohne weitere Einschränkungen von  $f(x)$  können wir idR nicht darauf hoffen garantiert das globale Optimum zu finden
- Heuristiken sind oft ausreichend um sinnvolle Optima zu finden
- Manuelle Initialisierung / Eingreifen bleibt manchmal notwendig

# Rastersuche

- **Idee:** Systematisches Ausprobieren verschiedener Initialisierungen
  - *Beispiel:* Je 10 Werte innerhalb eines sinnvollen Bereichs
- **Vorsicht:** Kombinatorische Explosion in höheren Dimensionen
  - *Beispiel:* Affine Transformation in 3D:  $10^{12}$  Kombinationen
  - Selbst wenn wir die Kostenfunktion in einer Zehntelsekunde auswerten können verbringen wir damit 3000 Jahre!
- **Dennoch:** Nützliche Idee für leicht zu berechnende Kostenfunktionen und niedrigdimensionale Suchräume

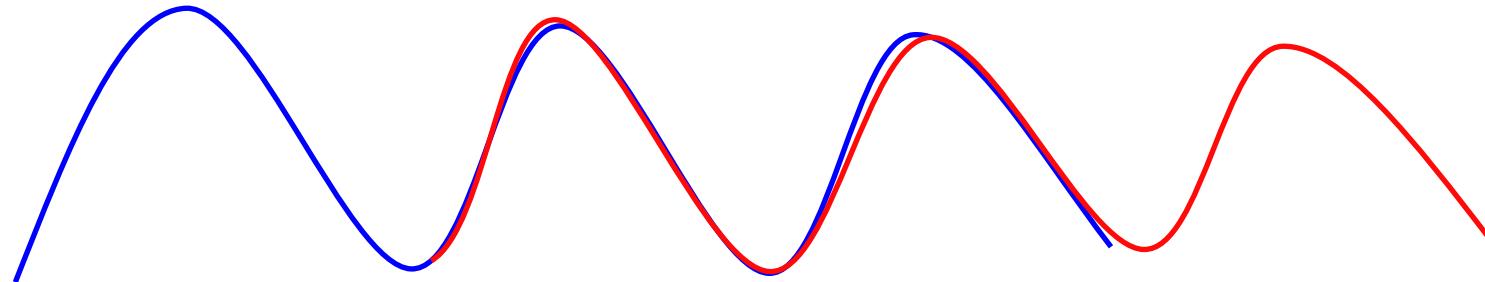
# Optimierung auf mehreren Skalen

- **Idee:** Registriere zunächst auf einer groben Skala, verfeinere dann iterativ auf detaillierteren Skalen. *Vorteile:*
  - Kostenfunktion schnell zu berechnen
  - Konvergenz in wenigen Schritten
  - Eliminierung feiner Strukturen reduziert Probleme mit Suboptima



# Perturbation und Neustart

- Zufällige Perturbation einer Lösung und Neustart der Optimierung kann aus lokalen Optima heraushelfen
  - *Aber:* Könnte natürlich auch in einem noch schlechteren Suboptimum enden
  - *Daher:* Vergleich zu vorherigem Optimum, Auswahl des besseren



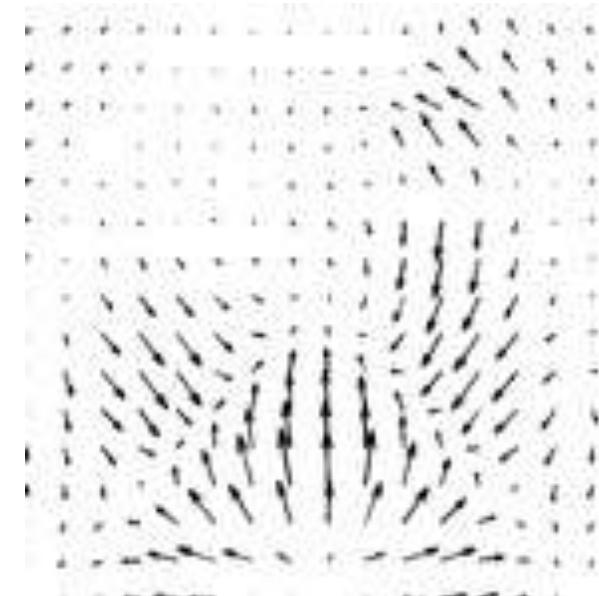
# Zusammenfassung: Registrierung per Optimierung

- Die Norm des Differenzbildes ist eine einfache **Kostenfunktion** zur intensitätsbasierten Bildregistrierung
- Optimierung stetiger 1D-Funktionen per **Suche mit dem goldenen Schnitt**
  - *Grundidee*: Iterative Verkürzung eines Suchintervalls
- **Alternierende 1D-Optimierung** als Grundidee für höhere Dimensionen
  - Verfahren von Powell findet geeignete Suchrichtungen
- Bessere Ergebnisse in der Praxis durch **Heuristiken** wie Rastersuche, Mehrskalen-Optimierung, Perturbation und Neustart

## **5.4 Nichtlineare Registrierung**

# Deformierbare Registrierung

- **Affine Transformationen** sind nicht flexibel genug, um
  - Bewegungen der meisten Organe auszugleichen (z.B. Atmung, Herzschlag, Brain Shift)
  - Anatomie verschiedener Patienten aufeinander abzubilden
- **Nichtlineare Registrierung** ermöglicht die hierzu benötigten lokalen Deformationen
  - *Grundidee:* Darstellung der Deformation durch ein Verschiebungsvektorfeld
  - Dieses muss **regularisiert** werden, da sonst
    - nicht genug Bildinformation zur Verfügung steht, um es zu schätzen
    - unplausible Deformationen erzeugt werden



# Verwandtes Problem: Optischer Fluss

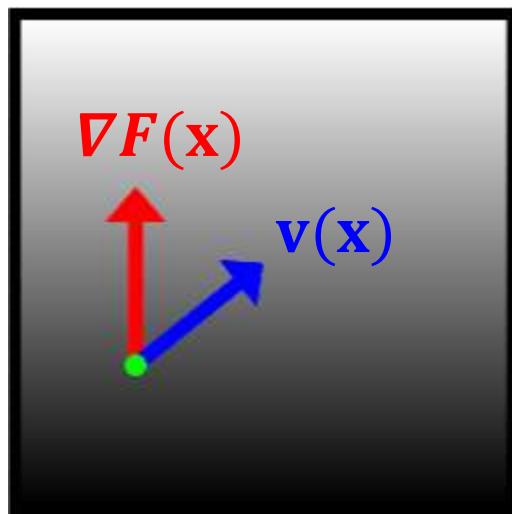


Bildquelle: Blender Foundation / [bigbuckbunny.org](http://bigbuckbunny.org)

# Optischer Fluss

- **Ziel:** Finde in einer Bildsequenz die Verschiebungsvektoren  $\mathbf{v}(\mathbf{x})$  jeden Pixels  $\mathbf{x}$  von einem Bild ins nächste
- **Zugrundeliegende Annahme:** Pixel bewegen sich, ändern jedoch nicht ihre Intensität
- Approximation erster Ordnung:

$$M(\mathbf{x}) = F(\mathbf{x}) - \langle \mathbf{v}(\mathbf{x}), \nabla F(\mathbf{x}) \rangle$$



**Hinweis:** Nur die Bewegung entlang des Gradienten wirkt sich auf die Intensität aus. Dies bezeichnet man als *Apertureproblem*.

# Lösung des Aperturproblems

- Um das **Aperturproblem** zu lösen, wählen wir den Verschiebungsvektor  $\mathbf{v}(\mathbf{x})$  mit der kleinsten möglichen Norm:  $\mathbf{v}(\mathbf{x}) = \lambda \nabla F(\mathbf{x})$
- Aus  $M(\mathbf{x}) = F(\mathbf{x}) - \langle \mathbf{v}(\mathbf{x}), \nabla F(\mathbf{x}) \rangle$  folgt dann

$$M(\mathbf{x}) = F(\mathbf{x}) - \lambda \|\nabla F(\mathbf{x})\|_2^2$$

$$\Rightarrow \lambda = \frac{F(\mathbf{x}) - M(\mathbf{x})}{\|\nabla F(\mathbf{x})\|_2^2}$$

$$\Rightarrow \mathbf{v}(\mathbf{x}) = \frac{F(\mathbf{x}) - M(\mathbf{x})}{\|\nabla F(\mathbf{x})\|_2^2} \nabla F(\mathbf{x})$$

– Quiz: Gibt es mit dieser Formel noch ein Problem?

# Dämonen-Regularisierung

- Dämonen-Regularisierung für  $\|\nabla F(\mathbf{x})\|^2 \approx 0$ :

$$\mathbf{v}_D(\mathbf{x}) = \frac{F(\mathbf{x}) - M(\mathbf{x})}{\|\nabla F(\mathbf{x})\|_2^2 + (F(\mathbf{x}) - M(\mathbf{x}))^2} \nabla F(\mathbf{x})$$

– wenn  $\|\nabla F(\mathbf{x})\|_2^2 + (F(\mathbf{x}) - M(\mathbf{x}))^2 < \epsilon : \mathbf{v}_D(\mathbf{x}) := \mathbf{0}$

- Hält Korrespondenzen **in der Nachbarschaft** ( $\|\mathbf{v}_D\| \leq 0.5$ )

$$\begin{aligned}\|\mathbf{v}_D(\mathbf{x})\| &= \frac{|F(\mathbf{x}) - M(\mathbf{x})| \|\nabla F(\mathbf{x})\|}{\|\nabla F(\mathbf{x})\|_2^2 + (F(\mathbf{x}) - M(\mathbf{x}))^2} \\ &= \frac{\text{GM}^2(|F(\mathbf{x}) - M(\mathbf{x})|, \|\nabla F(\mathbf{x})\|)}{2 \text{QM}^2(|F(\mathbf{x}) - M(\mathbf{x})|, \|\nabla F(\mathbf{x})\|)}\end{aligned}$$

– mit GM=geometrisches Mittel, QM=quadratisches Mittel  
–  $\|\mathbf{v}_D\| \leq 0.5$  folgt aus der Ungleichung  $\text{GM} \leq \text{QM}$

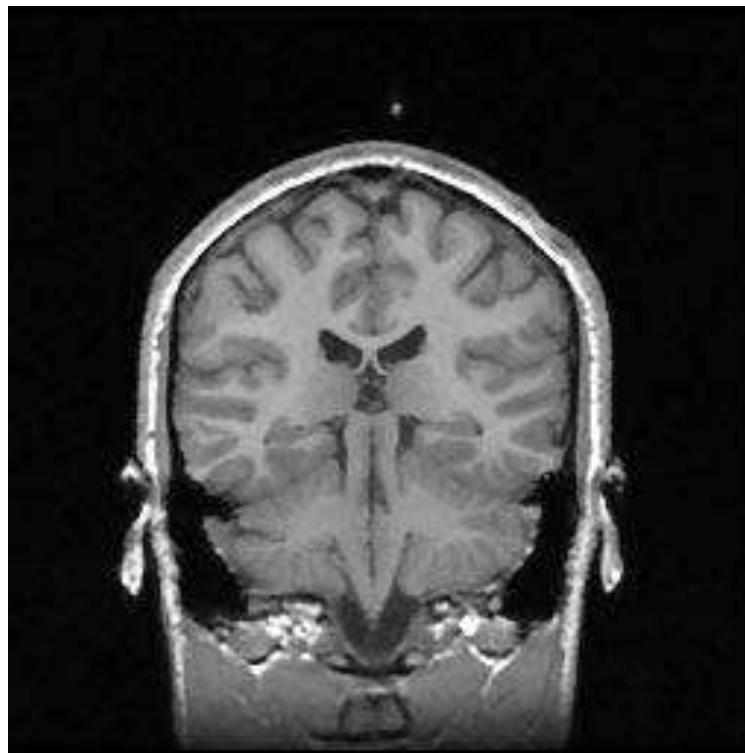
# Dämonen-Algorithmus zur Bildregistrierung

- Der **Dämonen-Algorithmus** zur Bildregistrierung (Thirion 1998) iteriert folgende Schritte:
  1. Berechne den regularisierten optischen Fluss  $\bar{\psi}(x)$  zwischen dem festen Bild  $F(x)$  und dem deformierten Objektbild  $M(\psi(x))$ .
  2. Addiere die Flussvektoren auf das aktuelle Verschiebungsfeld:  
$$\hat{\psi}(x) = \psi(x) + \bar{\psi}(x)$$
  3. Regularisiere durch Gauss-Glättung:  $\varphi(x) = G_\sigma * \hat{\psi}(x)$ .  
Das ergibt das Verschiebungsfeld  $\psi(x)$  der nächsten Iteration

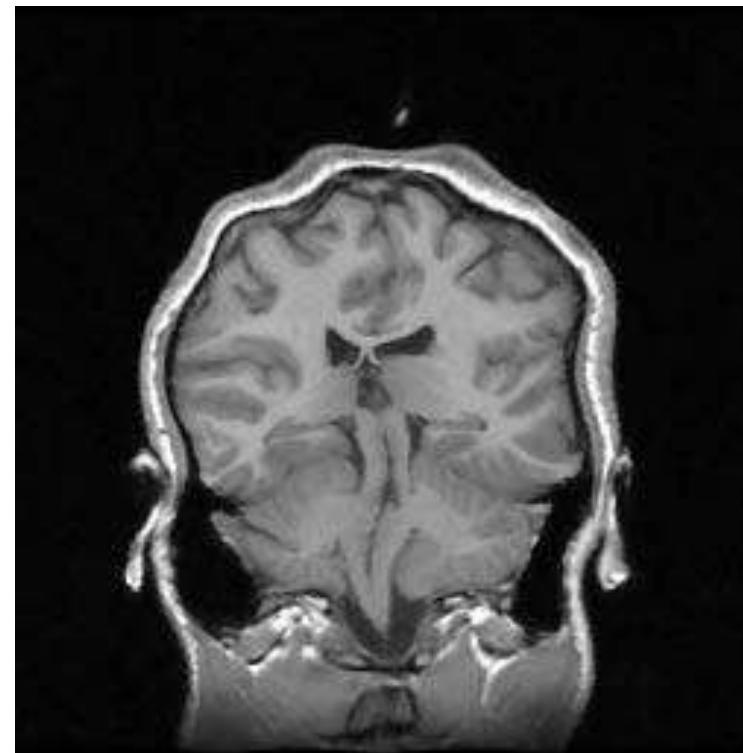
**Vorstellung:** “Maxwellsche Dämonen” besetzen die Pixel und verschieben das Bild aufgrund der Intensitätsunterschiede.

# Beispiel-Ergebnis des Dämonen-Algorithmus

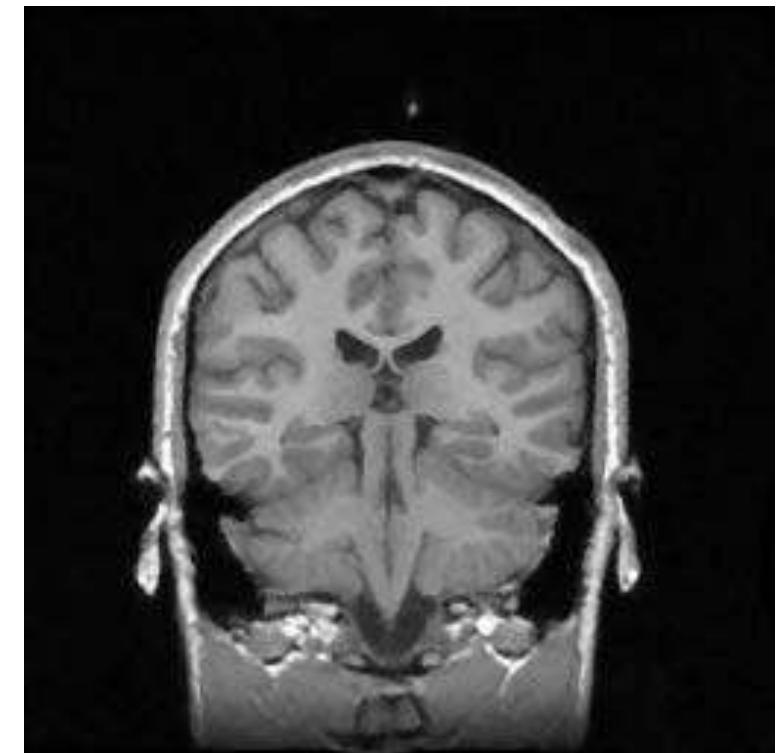
- Ergebnis von (Thirion 1998)



(a) Ursprüngliches MRT-Bild



(b) Deformierte Variante von (a)



(c) Registrierung von (b) auf (a)

# Zusammenfassung: Nichtlineare Registrierung

- **Grundidee** der nichtlinearen Registrierung:
  - Transformation durch ein **Verschiebungsvektorfeld** bietet genügend Flexibilität für lokale Deformationen
  - **Regularisierung** beschränkt Flexibilität auf ein sinnvolles Maß
- Klassischer Ansatz: **Dämonen-Algorithmus**
  - Basiert auf iterativer Berechnung des **optischen Flusses** zwischen Referenz und deformiertem Objektbild
    - Regularisiert optischen Fluss um sehr lange Vektoren zu vermeiden
  - Regularisiert das Verschiebungsvektorfeld zusätzlich durch **Gauss-Glättung** in jeder Iteration

# Zum Nach- und Weiterlesen

- Heinz Handels: *Medizinische Bildverarbeitung*. Vieweg+Teubner, 2. Auflage, 2009
- Isaac N. Bankman: *Handbook of Medical Imaging. Processing and Analysis*. Academic Press, 2000
- W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery: *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 3<sup>rd</sup> edition, 2007
- J.-P. Thirion: *Image Matching as a Diffusion Process: An Analogy with Maxwell's Demons*. Medical Image Analysis 2(3):243-260, 1998

# Kapitel 6a: Grundlagen Neuronaler Netze

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

Per Video: 9. Januar 2025



# Ehrungen für Pioniere des Tiefen Lernens

- **Turing-Preis 2018** an Yoshua Bengio, Geoffrey Hinton und Yann LeCun, für Durchbrüche zur Etablierung tiefer neuronaler Netze
- **Physik-Nobelpreis 2024** an John Hopfield und Geoffrey Hinton, für „grundlegende Entdeckungen und Erfindungen die maschinelles Lernen mit neuronalen Netzen ermöglichen“



Yoshua Bengio

Geoffrey Hinton

Yann LeCun



III. Niklas Elmehed © Nobel Prize Outreach  
John J. Hopfield

III. Niklas Elmehed © Nobel Prize Outreach  
Geoffrey Hinton



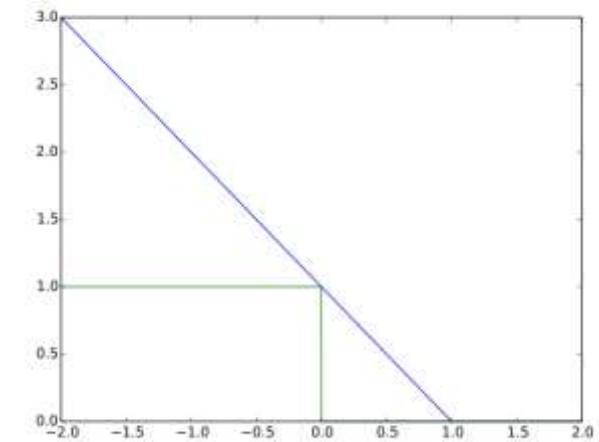
## **6a.1 Grundkonzepte des Maschinellen Lernens**

# Grundidee: Überwachtes Maschinelles Lernen

- Ziel des **überwachten maschinellen Lernens** ist es, aus **Trainingsbeispielen**  $\{\mathbf{x}_i, y_i\}$  eine sinnvolle Funktion  $y = f(\mathbf{x})$  zu lernen
  - **Klassifikation:**  $y$  ist diskret
    - Spezialfall: binär, z.B.  $y = \pm 1$
    - *Beispiele:* Zeigt das Bild  $\mathbf{x}$  Hautkrebs? Gehört Pixel  $\mathbf{x}$  zu einer Zellmembran?
  - **Regression:**  $y$  ist kontinuierlich
    - *Beispiel:* Wie alt ist der Proband, von dem Hirnscan  $\mathbf{x}$  stammt?
- *Grundannahme:* **Merkmalsvektoren**  $\mathbf{x}_i$  und **Labels**  $y_i$  sind unabhängige Stichproben einer festen Wahrscheinlichkeitsverteilung  $P(\mathbf{x}, y)$ 
  - **Tiefes Lernen** ermöglicht es hochdimensionale Daten (z.B. Bilder) direkt als Eingabe  $\mathbf{x}$  zu verwenden, statt sie auf einen Merkmalsvektor zu reduzieren

# Verlust und Risiko

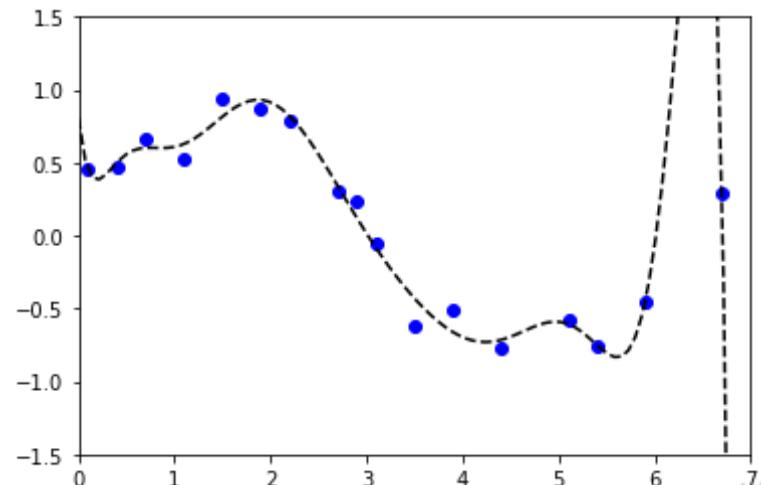
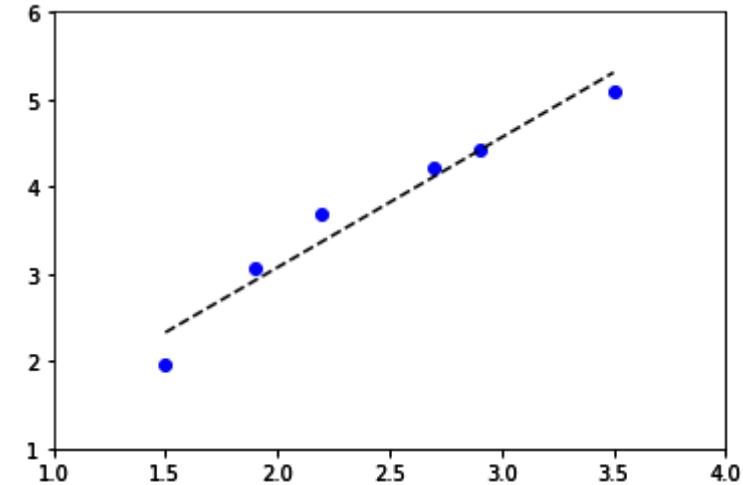
- Eine **Verlustfunktion**  $L(y, y')$  quantifiziert den Schaden der Vorhersage  $y' = f(\mathbf{x})$ , wenn  $y$  korrekt ist
  - *Binäre Klassifikation:*
    - Null-Eins-Verlust: Null wenn  $y = y'$ , sonst eins
    - Scharnier-Verlust (*engl.* hinge loss):  $L=\max(0,1-yy')$
  - *Regression:*
    - Quadratischer Verlust (Gauß-Verlust):  $L=(y-y')^2$
- Ziel des Trainings ist Minimierung des **Risikos**, d.h. des erwarteten Verlusts von  $f(\mathbf{x})$  unter der Verteilung  $P(\mathbf{x}, y)$ 
  - *Grundannahme:*  $f(\mathbf{x})$  soll auf Daten angewandt werden, die aus derselben Verteilung stammen wie die Trainingsdaten



Hinge loss mit  $y=1$

# Training als Optimierungsproblem

- Die zu lernende Funktion  $f(\mathbf{x})$  wird häufig durch **Parameter** bestimmt
  - *Beispiel:* Anpassung einer Ausgleichsgeraden an Messpunkte
    - Parameter: Achsenabschnitt, Steigung
- Das **Training** optimiert die Parameter im Hinblick auf eine Zielfunktion
  - Reine Fokussierung auf das **empirische Risiko**, d.h. den mittleren Verlust auf den Trainingsdaten  $\{\mathbf{x}_i, y_i\}$ , führt bei flexiblen Parametern leicht zu unplausiblen Funktionen
  - Regularisierung bevorzugt „einfache“  $f(\mathbf{x})$



# Trainings- / Validierungs- / Testdaten

Bei der Entwicklung von Methoden mittels überwachten Lernens müssen die verfügbaren Daten partitioniert werden:

- Mittels der **Trainingsdaten** werden die Parameter optimiert
  - *Merke*: Geringer Trainingsfehler (empirisches Risiko) garantiert noch keine geeignete Generalisierung auf neue Daten!
- Mittels **Validierungsdaten** werden Hyperparameter eingestellt, z.B. Stärke der Regularisierung, Art der Optimierung
- Mittels **Testdaten** kann das Risiko von  $f(x)$  geschätzt werden
  - *Wichtig*: Trainings-, Validierungs- und Testdaten müssen disjunkt sein.  
Vorsicht auch bei wiederholten Messungen desselben Patienten!
  - Bestenfalls werden Testdaten erst nach der Entwicklung verfügbar

# Evaluierung von Klassifikatoren

- Für binäre Klassifikatoren ergeben sich in der **Konfusionsmatrix** folgende Fälle
- Daraus leitet man ab:
  - **Korrektklassifikationsrate**

$$\text{(engl. accuracy)} \quad ACC = \frac{RP+RN}{RP+RN+FP+FN}$$

		Vorhersage $f(\mathbf{x})$	
		Positiv	Negativ
Label $y$	Positiv	Richtig Positiv (RP)	Falsch Negativ (FN)
	Negativ	Falsch Positiv (FP)	Richtig Negativ (RN)

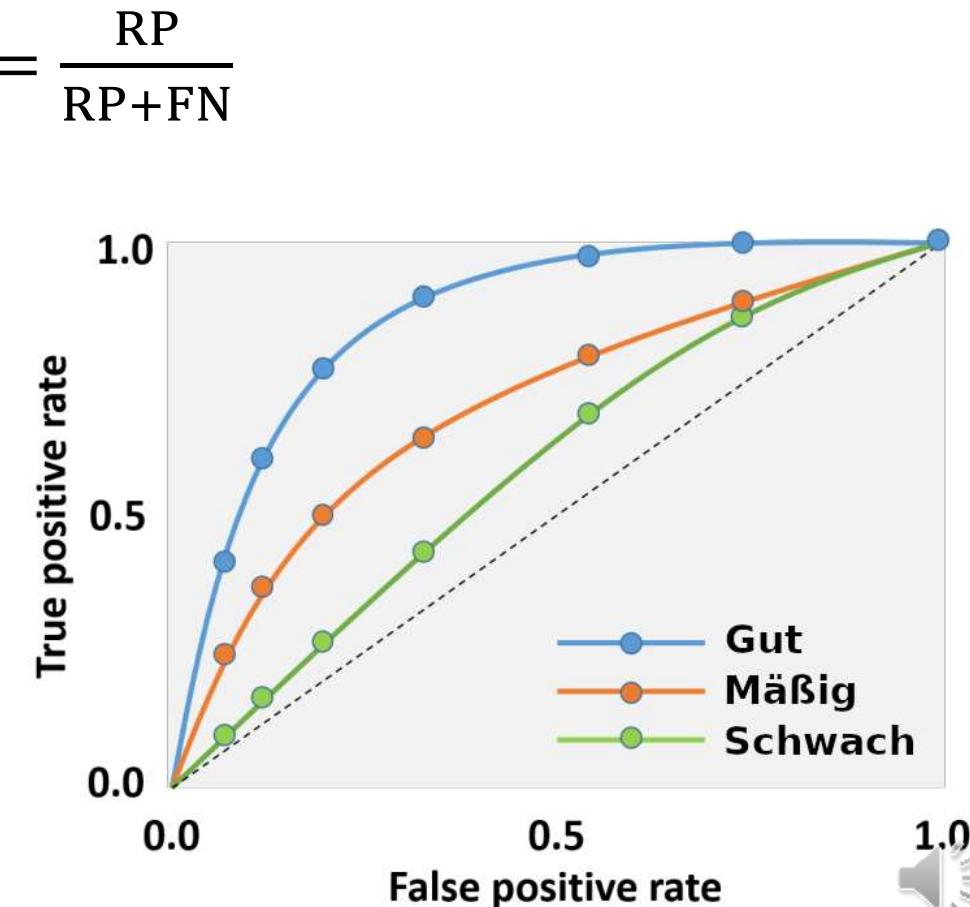
$$\text{– Positiver Vorhersagewert (engl. precision)} \quad P = \frac{RP}{RP+FP}$$

$$\text{– Sensitivität / Trefferquote (engl. recall)} \quad R = \frac{RP}{RP+FN}$$

$$\text{– F-Maß} \quad F = 2 \frac{P \cdot R}{P + R}$$

# Evaluierung mittels ROC-Kurven

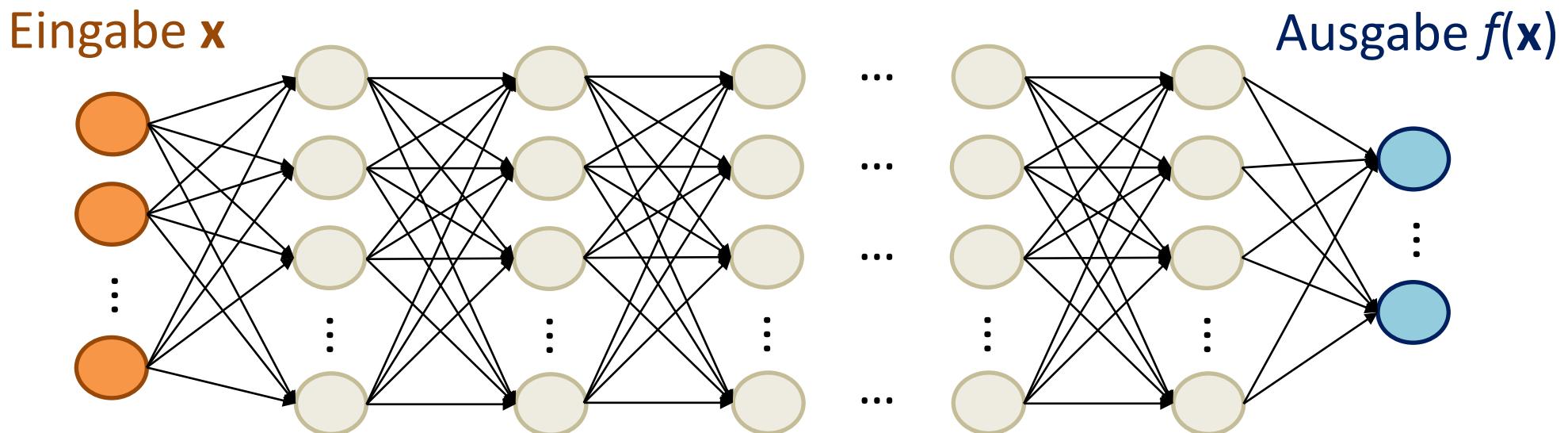
- Viele binäre Klassifikatoren basieren auf einer kontinuierlichen Entscheidungsfunktion. Durch Wahl verschiedener Schwellenwerte kann man die Balance einstellen zwischen
  - Sensitivität = **Richtig-Positiv-Rate**  $TPR = R = \frac{RP}{RP+FN}$
  - **Falsch-Positiv-Rate**  $FPR = \frac{FP}{RN+FP}$
- Die **ROC-Kurve** (*engl. receiver operating characteristic*) zeigt die Auswirkungen verschiedener Einstellungen
  - ROC-Kurven nahe der Diagonalen entsprechen zufälliger Klassifikation
  - Die **Fläche unter der ROC-Kurve** (*engl. Area under the curve, AUC*) ist ein weiteres Maß für die Güte eines Klassifikators



## **6a.2 Grundlagen Neuronaler Netze**

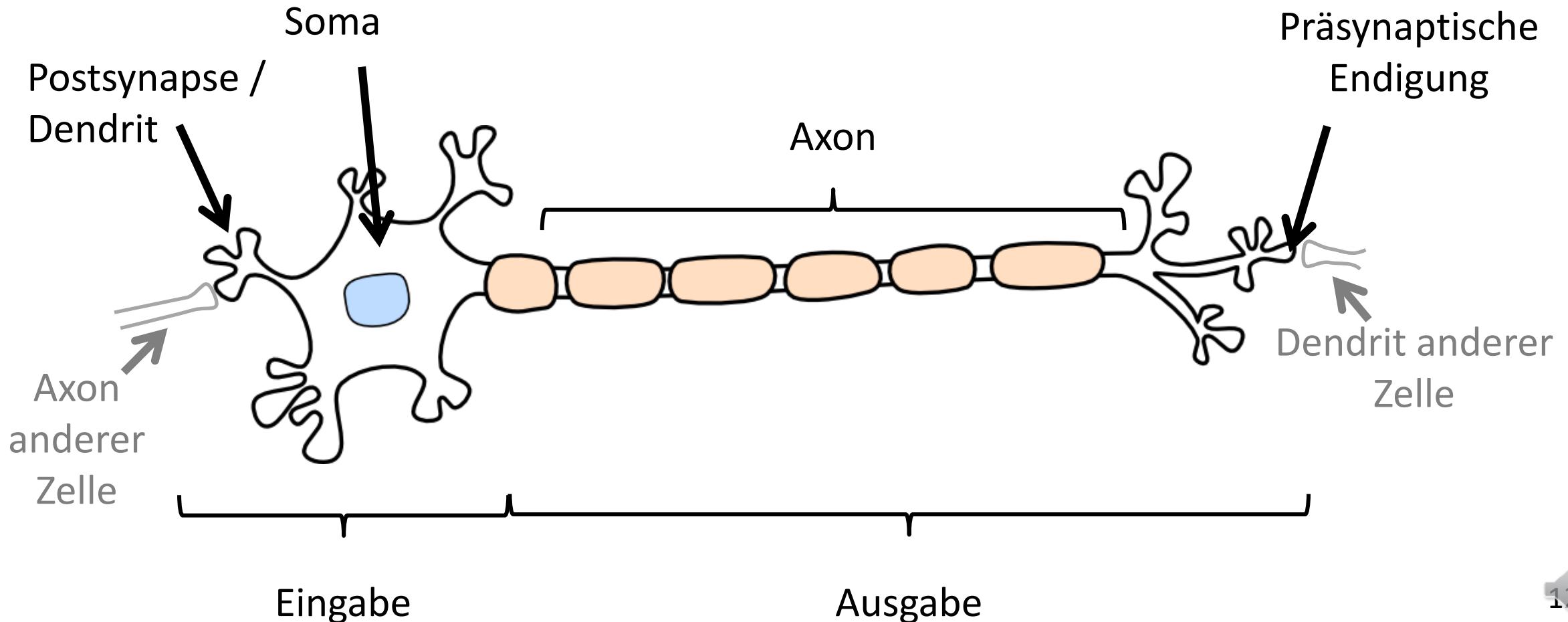
# Grundidee Neuronaler Netze

- **Neuronale Netze** setzen die zu lernende Funktion  $f(\mathbf{x})$  aus vielen einfachen Bausteinen zusammen, den künstlichen Neuronen
- Im **vorwärtsgerichteten** Fall (*engl. feed forward*) lassen sich die Neuronen so in Schichten anordnen, dass die Ausgaben jeder Schicht nur von späteren Schichten verarbeitet werden



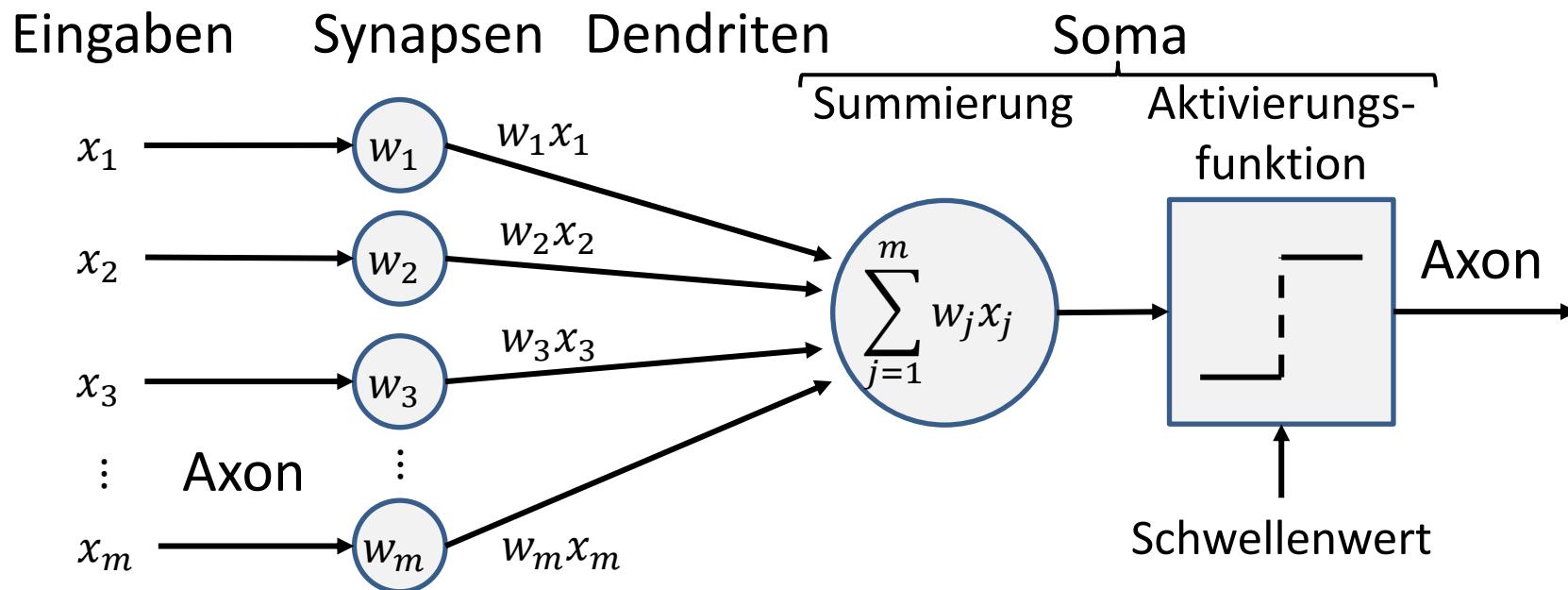
# Inspiration: Das Biologische Neuron

- Neuronale Netze sind von der Informationsverarbeitung im menschlichen Gehirn inspiriert



# Vom Biologischen zum Künstlichen Neuron

Biologisches Neuron	Künstliches Neuron
Axon	Verbindung zwischen den Schichten
Synapse	Gewichte
Dendrit	Gewichtete Eingaben
Soma	Summierung und Aktivierungsfunktion



# Künstliche Neurone: Formel

Die Ausgabe  $y_k$  des  $k$ -ten künstlichen Neurons einer Schicht ist durch folgende Formel gegeben:

$$y_k = f \left( \sum_{j=1}^m w_{kj} x_j + b_k \right)$$

- Eingaben  $x_j$  ( $j = 1, \dots, m$ )
- Gewichte  $w_{kj}$
- Negativer Schwellenwert (engl. bias)  $b_k$
- Aktivierungsfunktion  $f$

Hinweis: Diese Formel ist von biologischen Neuronen inspiriert, aber kein realistisches Modell!

# Künstliche Neurone: Matrix-Notation

Häufig werden die Parameter aller Neurone einer Schicht in einer Matrix dargestellt. Hierzu führen wir ein immer-an-Neuron ein ( $x_0 := 1$ ,  $w_{k0} := b_k$ ), so dass

$$y_k = f \left( \sum_{j=0}^m w_{kj} x_j \right)$$

Dies ermöglicht die Schreibweise

$$\mathbf{y} = f(\mathbf{W}\mathbf{x})$$

wobei  $f$  komponentenweise angewandt wird

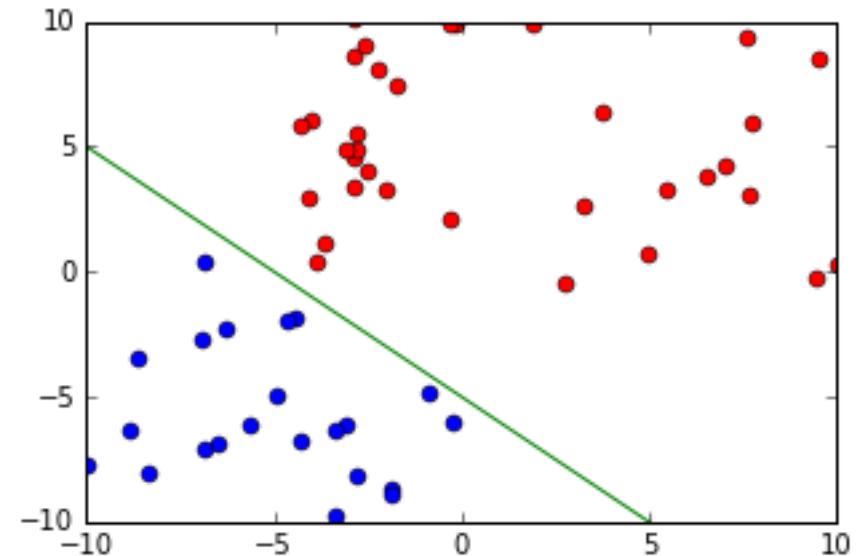
*Quiz: An was erinnert Sie dieser Trick?*

# Aktivierungsfunktionen

- Biologische Neurone erzeugen ein **Aktionspotential**, wenn die gewichtete Summe der Eingaben eine Schwelle überschreitet
- Definieren wir analog  $\alpha_k := \sum_{j=1}^m w_{kj}x_j + b_k$  und eine binäre **Aktivierungsfunktion**

$$f(\alpha_k) = \begin{cases} 1 & \text{wenn } \alpha_k > 0 \\ 0 & \text{wenn } \alpha_k \leq 0 \end{cases}$$

erhalten wir einen  
**linearen Klassifikator**



# Beispiel: Lineare Bildklassifikation

Eine **Bildklassifikation** mit einer einzigen Schicht von Neuronen entspräche einer affinen Funktion, die jeder Klasse einen Wert zuweist, der mit der Wahrscheinlichkeit der Klasse zunehmen soll

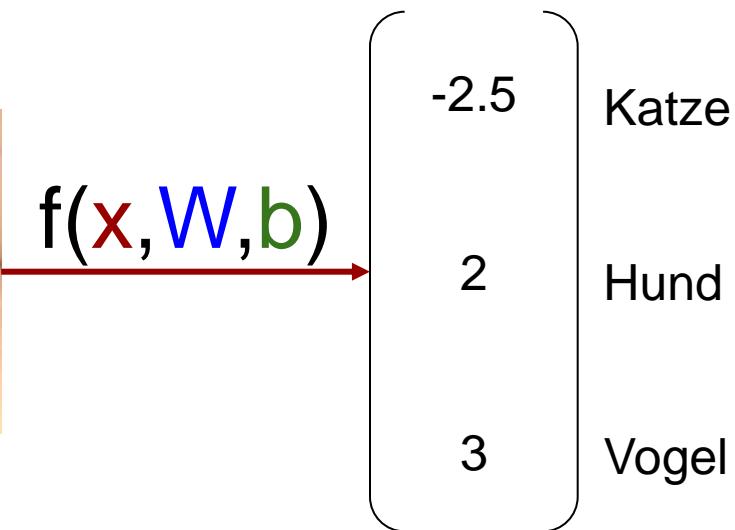
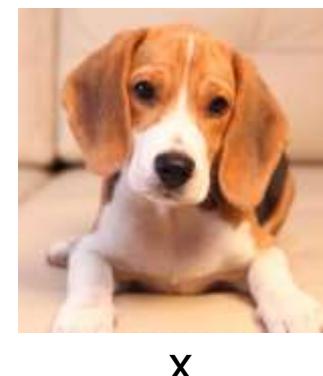
$$f(x, W, b) = Wx + b = s$$

**x**: Eingabe (Bild)

**W**: Parameter (Gewichte)

**b**: Bias (Versatz)

**s**: Bewertung (*engl.* Score) jeder Klasse



# Training einer Linearen Bildklassifikation

Ziel des **Trainings** einer linearen Bildklassifikation wäre es, die Parameter **W** und **b** so zu wählen, dass bei möglichst vielen Trainingsbildern  $x_i$  die korrekte Klasse den höchsten Wert hat

- Hierzu werden wir eine Verlustfunktion nach **W** und **b** ableiten



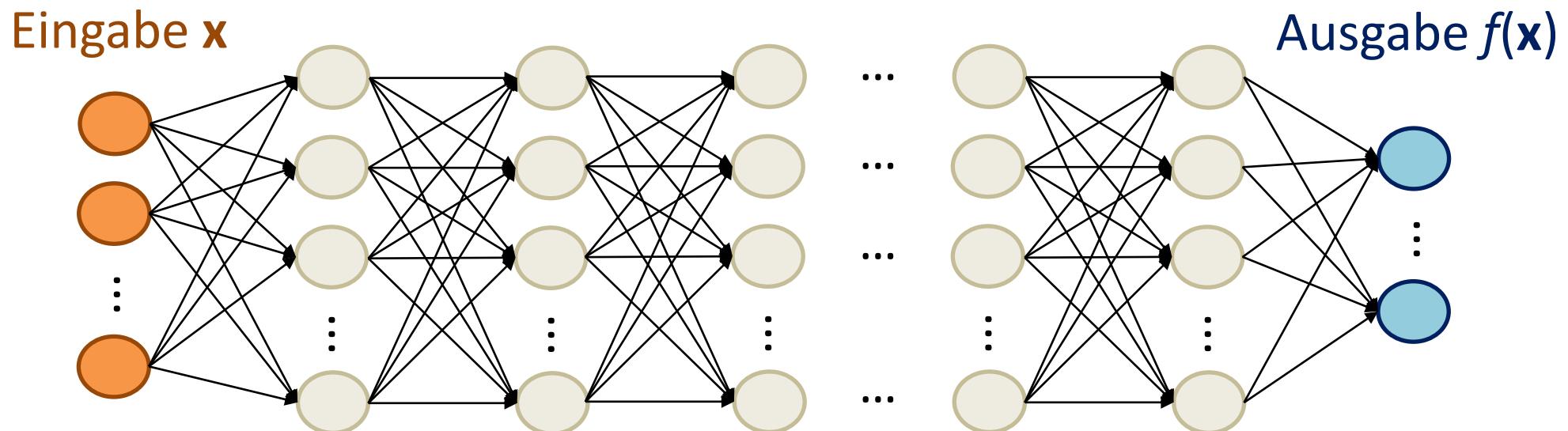
64x64  
RGB-Bild

$$\begin{array}{c} W \\ \hline \begin{matrix} 2.3 & 5 & 0 & \dots \\ -1 & -3 & 1 & \dots \\ 0.5 & 2 & -1 & \dots \end{matrix} \end{array} \quad \begin{array}{c} x_i \\ \hline \begin{matrix} 3 \\ 231 \\ 21 \\ \vdots \end{matrix} \end{array} \quad + \quad \begin{array}{c} b \\ \hline \begin{matrix} -1.5 \\ 2 \\ 1 \end{matrix} \end{array} = \begin{array}{c} f(x_i, W, b) = Wx_i + b \\ \hline \begin{matrix} -2.5 \\ 2 \\ 1 \end{matrix} \end{array}$$

The diagram illustrates the linear classification process. On the left, a 64x64 RGB image of a Beagle puppy is shown. To its right, the input vector  $x_i$  is represented as a column of feature values: 3, 231, 21, and a vertical ellipsis. In the center, a large black plus sign (+) indicates the addition of the weight matrix  $W$  and the bias vector  $b$ . To the right of the plus sign, the bias vector  $b$  is shown as a column with values -1.5, 2, and 1. Finally, the result of the addition, the function  $f(x_i, W, b) = Wx_i + b$ , is shown as a column with values -2.5, 2, and 1. This result is then mapped to three categories: Katze (Cat), Hund (Dog), and Vogel (Bird). The Hund category is highlighted with a large black equals sign (=).

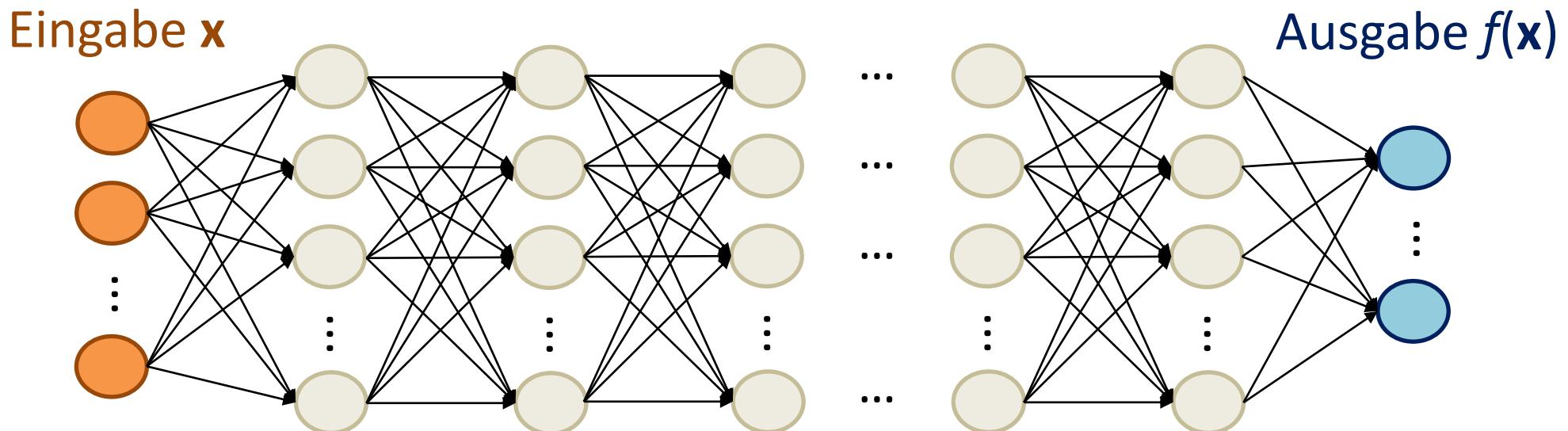
# Tiefe Neuronale Netze

- Komplexe Aufgaben wie Bildklassifikation erfordern komplizierte nichtlineare Funktionen  $f(\mathbf{x})$
- **Tiefe neuronale Netze** ermöglichen dies durch die Anordnung künstlicher Neurone in einer hohen Zahl von Schichten
  - Für die Bildklassifikation können das mehr als 100 sein
  - „Tiefe“ Netze haben mindestens zwei verborgene Schichten



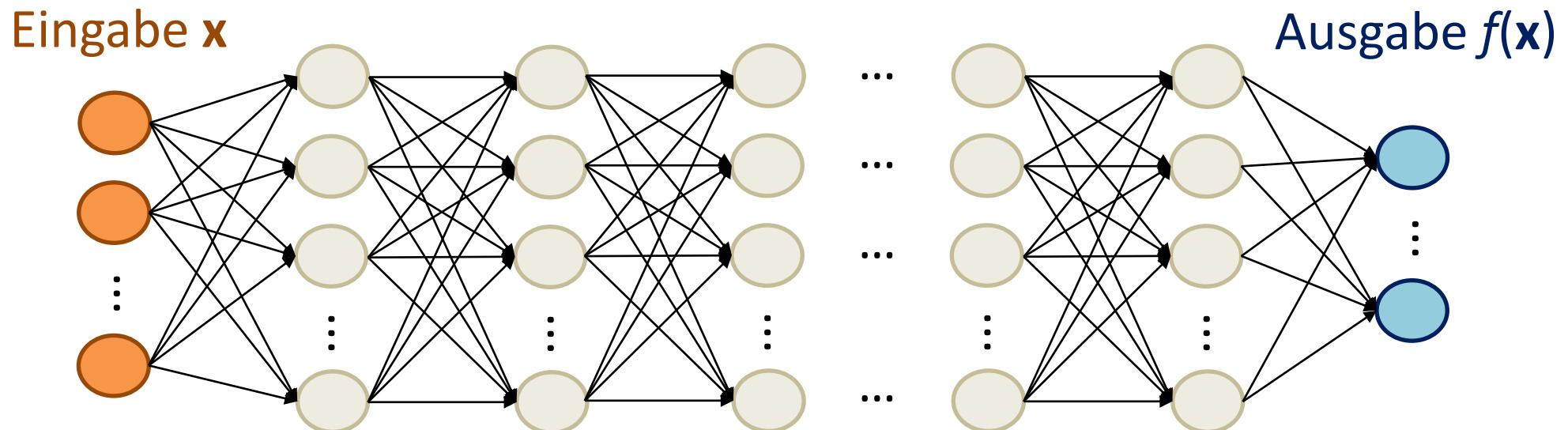
# Verborgene Schichten

- Ein- und Ausgaben sind aus den Trainingsdaten bekannt
  - Da die Eingabe keine Gewichte hat, zählt sie nicht als eigene Schicht
- Schichten zwischen Ein- und Ausgabe werden als „**verborgen**“ bezeichnet (*engl.* hidden layers)
  - Ermöglichen hierarchischen Aufbau abstrakter Repräsentationen
  - Sind für Menschen häufig nicht klar interpretierbar



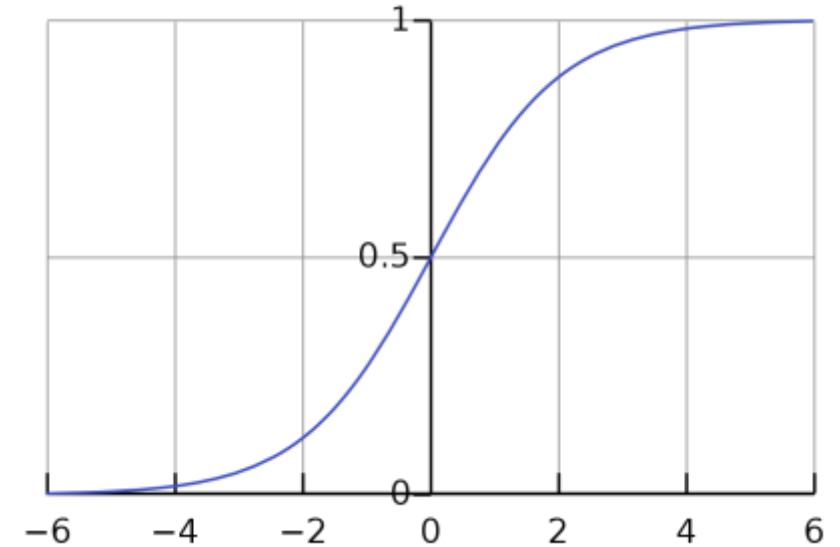
# Vollständig verbundene Schichten

- Zwei Schichten sind **vollständig verbunden**, wenn jedes Neuron der ersten mit jedem der zweiten verbunden ist
- Eine einzige Schicht ergibt einen linearen Klassifikator
  - Quiz: Warum bieten tiefere Netze uns nur dann einen Vorteil, wenn wir eine nichtlineare Aktivierungsfunktion  $f(\mathbf{Wx})$  nutzen?



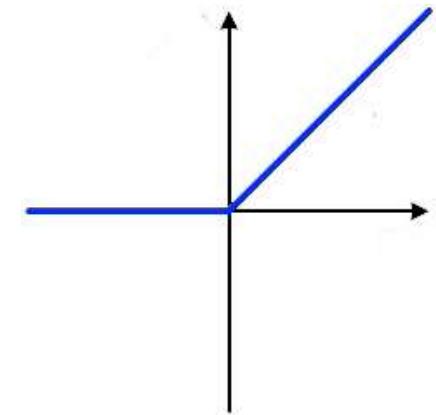
# Sigmoide Aktivierungsfunktion

- **Logistische Funktion**  $f(\alpha) = \frac{1}{1+e^{-\alpha}}$ 
  - Ausgabe in (0,1) als Wahrscheinlichkeit interpretierbar
  - Ähnliches Verhalten wie Sprungfunktion, aber sinnvoll differenzierbar
    - Voraussetzung für Training, siehe 6a.3
  - Ableitung einfach zu berechnen:  $f'(\alpha) = f(\alpha)(1 - f(\alpha))$
  - *Nachteil:* Ableitung für große/kleine  $\alpha$  nahe Null
  - Wird manchmal für Ausgabe-Schichten verwendet, selten für innere Schichten



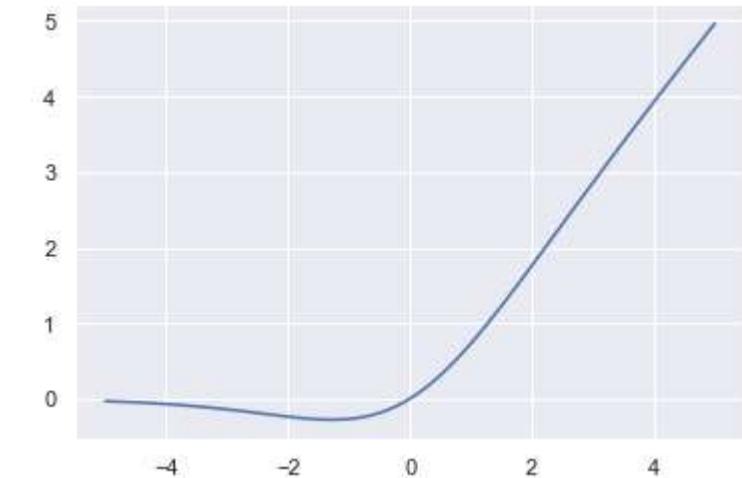
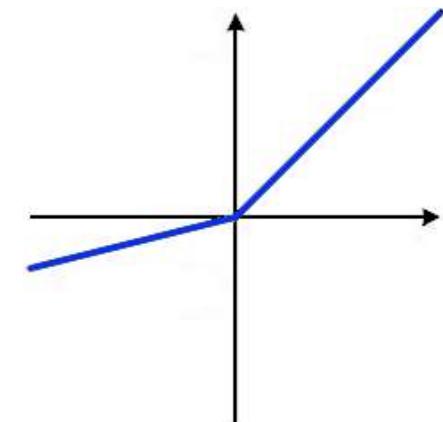
# Rectified Linear Unit (ReLU)

- **Rectified Linear Unit (ReLU)**  $f(\alpha) = \max(\alpha, 0)$ 
  - Lineare Einheit mit Rectifier (“Gleichrichter”)
  - *Biologische Motivation*: Neurone feuern unterhalb der Schwelle nicht, bei darüber hinaus zunehmender Erregung mit höherer Frequenz
  - $f(\alpha)$  und  $f'(\alpha)$  einfach und effizient zu berechnen
  - Verringert Probleme mit verschwindenden Gradienten
  - Sehr beliebt in tiefen Netzwerken
  - *Nachteil*: ReLUs können „sterben“ – wenn keine Eingabe sie aktiviert, gibt es keinen Gradienten um die Gewichte zu ändern



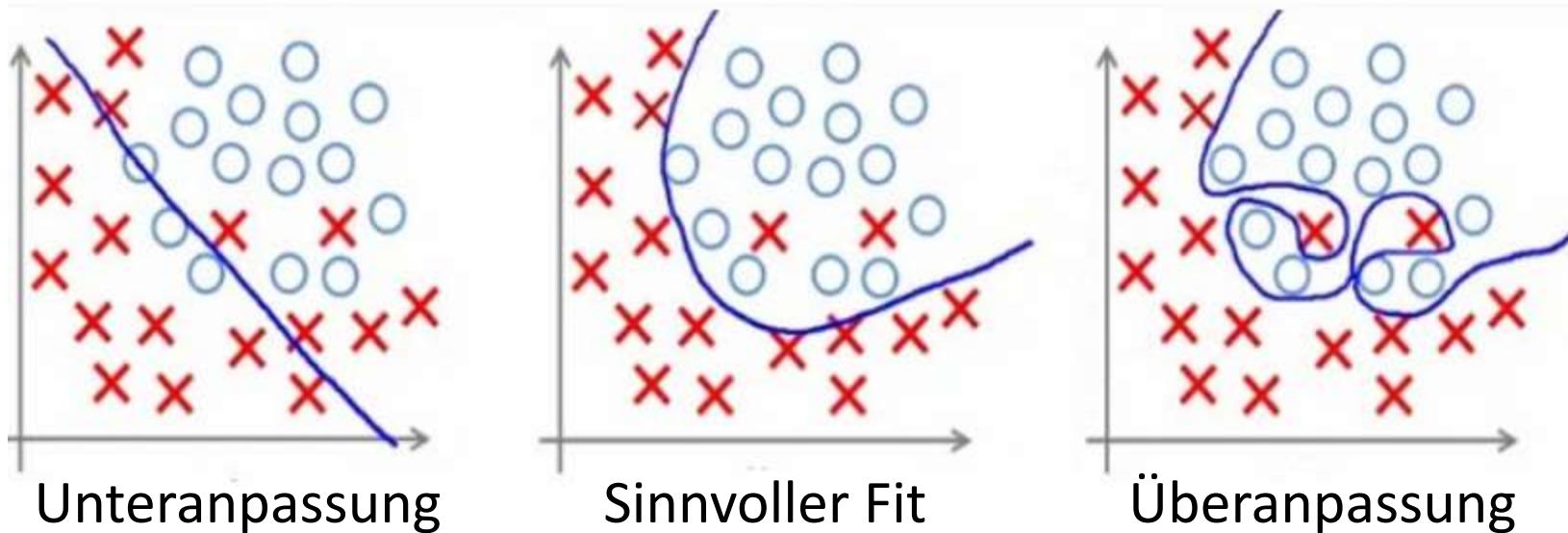
# Varianten der ReLU

- **Leaky ReLU<sup>1</sup>**  $f(\alpha) = \max(0.01\alpha, \alpha)$ 
  - Ziel: “Sterben” der ReLUs vermeiden
  - Parametrischer ReLU<sup>2</sup>: Statt 0.01 festzulegen wird dieser Faktor ebenfalls gelernt
- **Swish<sup>3</sup>**  $f(\alpha) = f(\alpha) = \alpha/(1 + e^{-\alpha})$ 
  - Überall differenzierbare Variante des ReLU
  - Nicht monoton
  - Führt bei tiefen Netzen häufig zu etwas besseren Ergebnissen
  - Erzeugt jedoch höheren Rechenaufwand



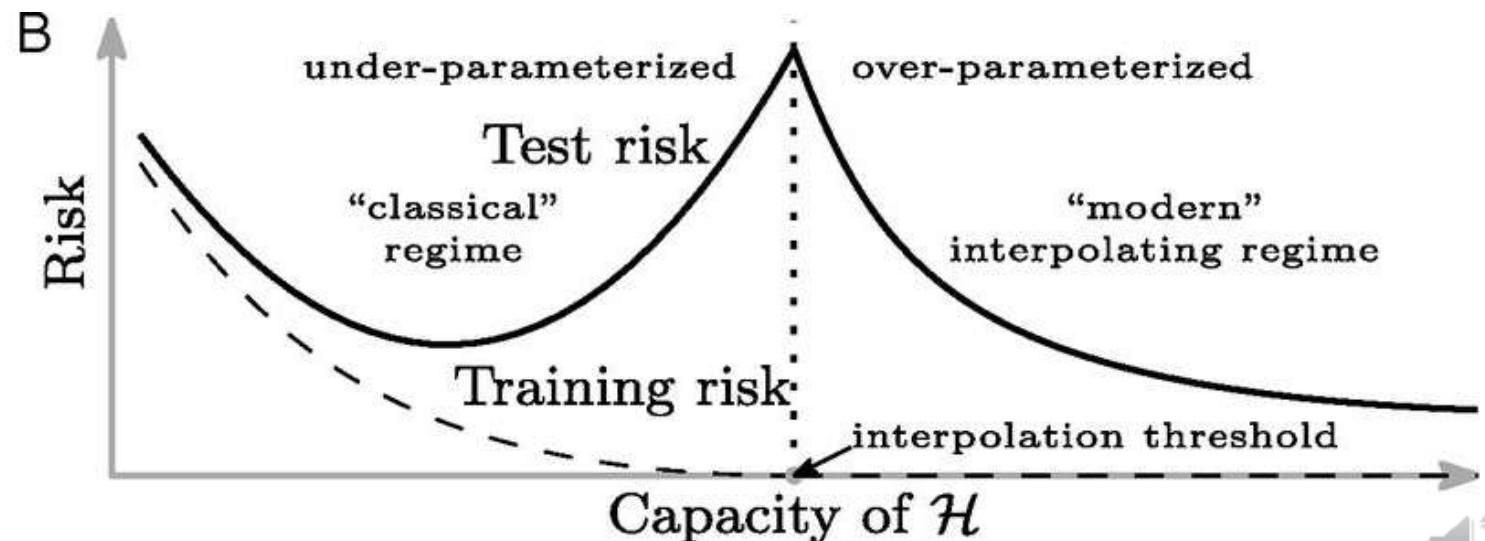
# Kapazität eines Neuronalen Netzwerks

- Mit zunehmender Zahl von Neuronen pro Schicht und zunehmender Tiefe stellen neuronale Netzwerke immer flexiblere Familien von Funktionen  $f(\mathbf{x})$  dar
- Zu hohe bzw. geringe Kapazität kann zu **Überanpassung** (engl. overfitting) bzw. **Unteranpassung** (engl. underfitting) führen



# Phänomen des „Doppelten Abstiegs“

- In der Praxis erreicht man die besten Ergebnisse häufig mit neuronalen Netzen, deren Kapazität *höher* ist, als es zur Interpolation der Trainingsdaten nötig wäre
- *Erklärungsansatz:* Solche Netze können die Trainingsdaten auf unterschiedliche Art interpolieren – auch mit solchen Funktionen  $f(x)$ , die sinnvoll auf neue Daten generalisieren
  - Algorithmen zum Training der Netze (siehe 6a.3) scheinen „gute“ Lösungen zu bevorzugen
  - Es ist nicht völlig geklärt, warum das so ist



# Zusammenfassung: Neuronale Netze

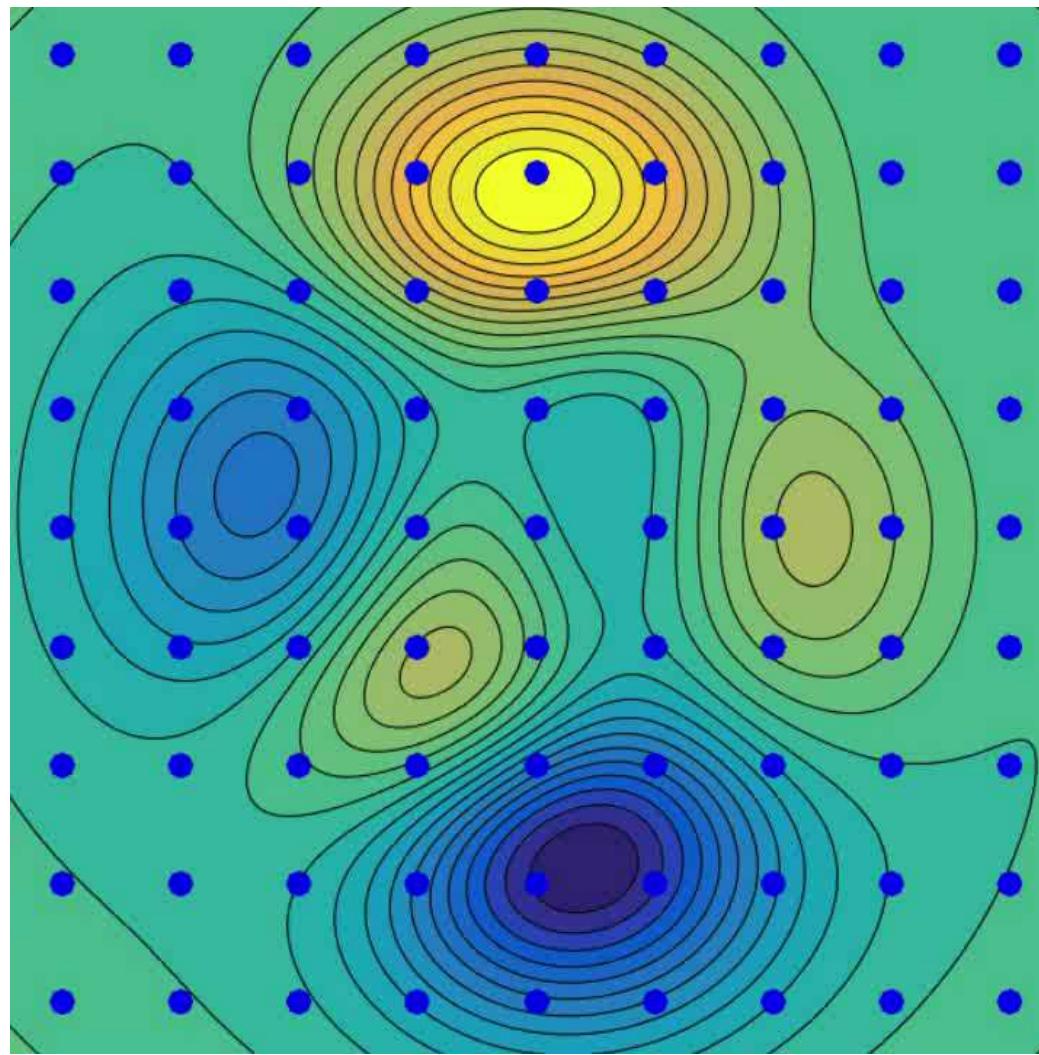
- **Neuronale Netze** sind eine beliebte Repräsentation lernbarer Funktionen  $f(\mathbf{x})$  hochdimensionaler Eingaben
- **Künstliche Neurone** sind ihre Bausteine
  - Affine Abbildung mit nichtlinearer Aktivierungsfunktion  $y = f(\mathbf{Wx})$
  - Neuronale Netze mit **einer einzigen Schicht** ergeben lineare Klassifikatoren
  - **Tiefe neuronale Netze** ermöglichen nichtlineare Funktionen
    - Benötigen geeignete **Aktivierungsfunktionen** wie z.B. ReLU
- Die **Kapazität** eines Netzwerks wird durch die Zahl der Schichten und Neurone bestimmt
  - Die genauen Werte sind **Hyperparameter**, die z.B. mittels Validierungsdaten eingestellt werden können

## **6a.3 Training Neuronaler Netze**

# Grundidee: Training Neuronaler Netze

- Aufgabe des **Trainings** ist es, die Parameter des neuronalen Netzes (insb. Gewichte und Bias-Terme) so einzustellen, dass die Funktion  $f(\mathbf{x})$  die gewünschte Aufgabe erfüllt
  - z.B. Bild  $\mathbf{x}$  einer Hautveränderung als gutartig oder bösartig einstufen
- Wie in 6a.1 erfolgt dies durch Minimierung einer **Verlustfunktion**  $L$ , die auf den Trainingsdaten Abweichungen zwischen Ausgaben  $f(\mathbf{x}_i)$  und Labeln  $y_i$  bestraft
  - *Grundlegende Strategie:* Gradientenabstieg, d.h. Berechnung der Ableitungen von  $L$  nach den Parametern und Veränderung der Parameter entgegen dieser Richtung
    - *Quiz:* In welchem Kontext haben wir bereits dieselbe Grundidee genutzt?

# Illustration: Gradientenabstieg



# Beispiel: Hinge-Loss für mehrere Klassen

- Der binäre Hinge-Loss aus 6a.1 lässt sich wie folgt auf mehrere Klassen verallgemeinern:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

- $y_i$  ist das korrekte Label für Trainingsbeispiel  $i$
- $s_j$  ist der Wert für Klasse  $j$
- Für alle  $N$  Trainingsdatenpunkte ergibt sich die Verlustfunktion

$$L = \frac{1}{N} \sum_i L_i$$



Katze	3.0	<b>3.0</b>	1.0
Hund	<b>1.0</b>	-5.0	3.2
Vogel	-2.5	1.5	<b>-5.0</b>
Verlust	3.0	0	16.2

# Probabilistische Vorhersagen mittels Softmax

- **Softmax** rechnet die Bewertungen der Klassen in bedingte Wahrscheinlichkeiten um:

$$P(y = k | \mathbf{x} = \mathbf{x}_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

– *Interpretation:* vom neuronalen Netz geschätzte Wahrscheinlichkeit, dass  $k$  das passende Label für  $\mathbf{x}_i$  ist



Katze	3.0	3.0	1.0
Hund	1.0	-5.0	3.2
Vogel	-2.5	1.5	-5.0
	exp ↓		
	20.08		0.87
	2.7	Normierung →	0.12
	0.08		0.00

# Kreuzentropie als Verlustfunktion

- Die Kreuzentropie eignet sich als Verlustfunktion für die Abweichung zweier Wahrscheinlichkeitsverteilungen
- Hat die Klasse  $y_i$  Wahrscheinlichkeit 1, entspricht die Kreuzentropie deren negativer Log-Likelihood. Zusammen mit der Softmax-Transformation ergibt sich

$$L_i = -\ln \left( \frac{e^{s_{y_i}}}{\sum_j e^{s_j}} \right)$$

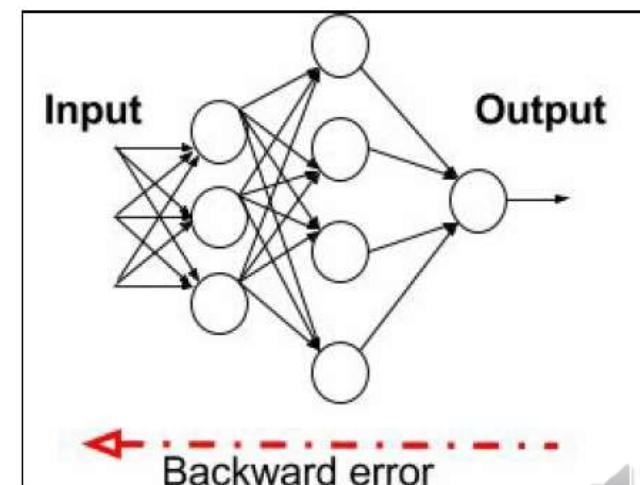
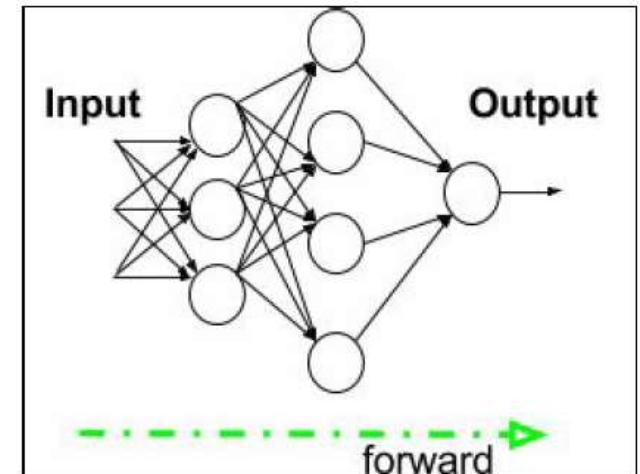


Katze	3.0	3.0	1.0
Hund	1.0	-5.0	3.2
Vogel	-2.5	1.5	-5.0
	exp ↓		
	20.08	0.87	
	2.7	Normierung → 0.12	
	0.08	0.00	
	L <sub>i</sub> =2.12		

# Grundidee der Backpropagation

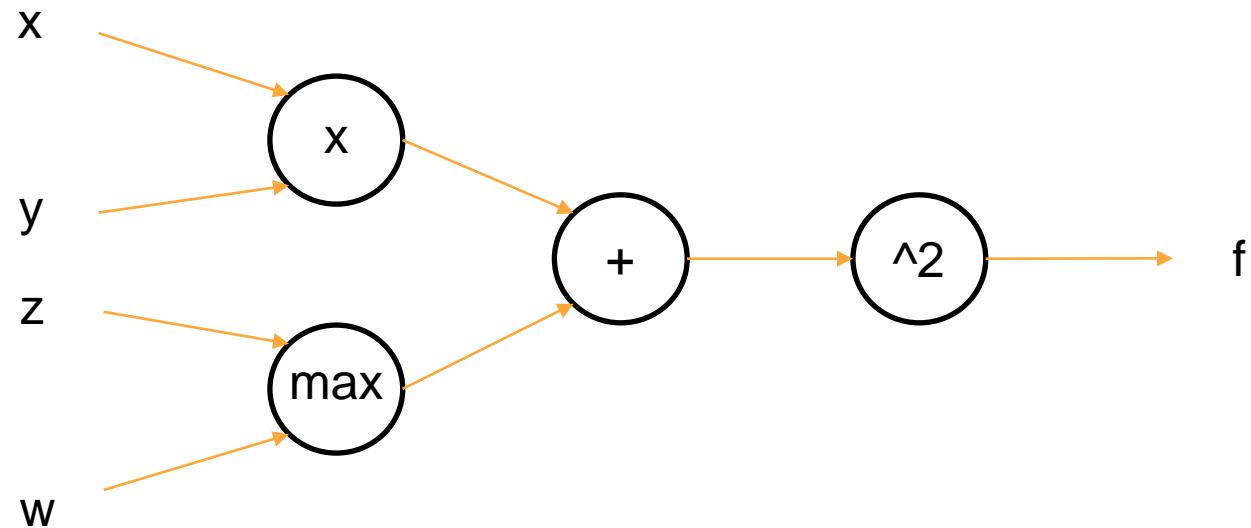
Ziel: Gradientenabstieg erfordert Ableitung der Verlustfunktion nach den Netzwerkparametern

1. **Vorwärtspropagierung** berechnet die Ausgabe des Netzwerks und ermöglicht Berechnung der Verlustfunktion
  - Alle Zwischenergebnisse werden gespeichert
2. **Backpropagation** (Rückpropagierung) berechnet schrittweise, von hinten nach vorn, die gewünschten partiellen Ableitungen
  - Nutzt mittels Kettenregel die Zwischenergebnisse



# Beispiel: Vorwärtspropagierung

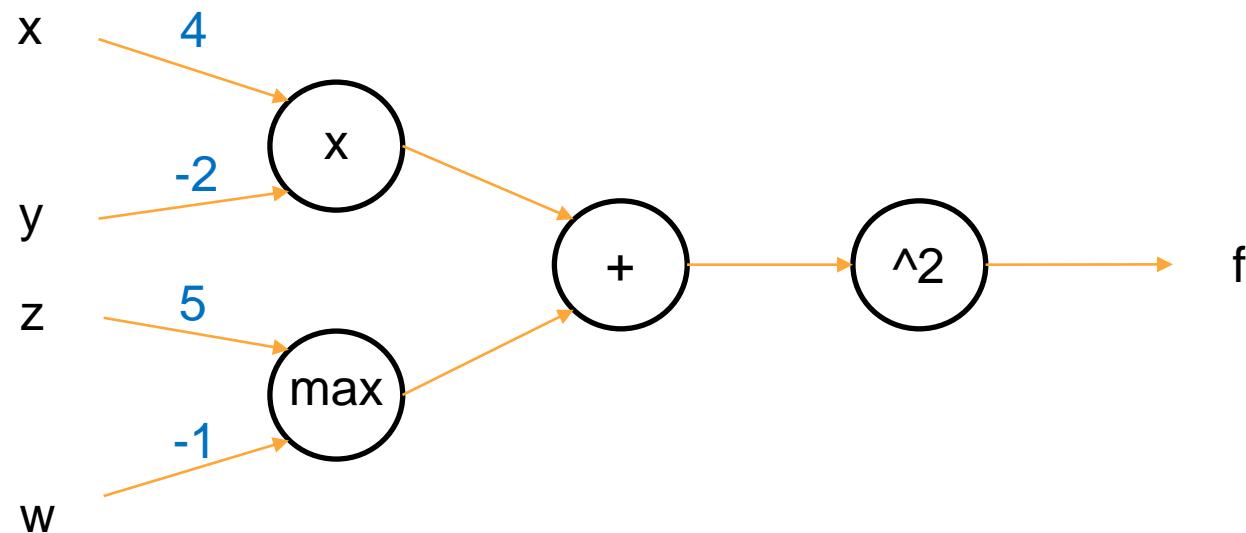
$$f(x, y, z, w) = (xy + \max(z, w))^2$$



# Beispiel: Vorwärtspropagierung

$$f(x, y, z, w) = (xy + \max(z, w))^2$$

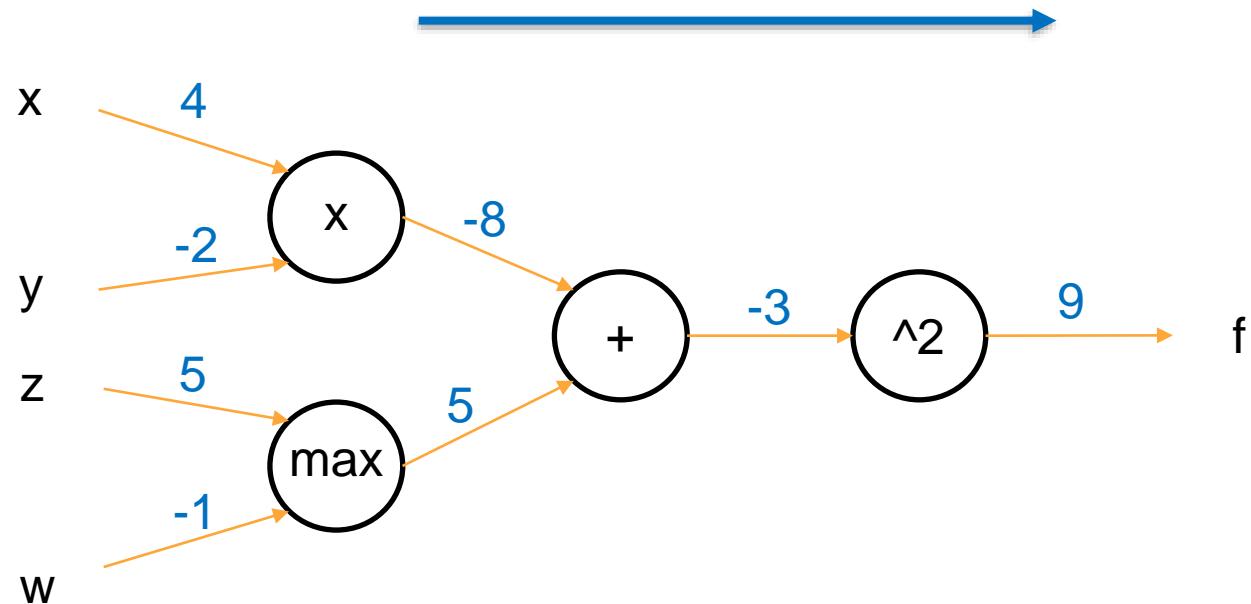
Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$



# Beispiel: Vorwärtspropagierung

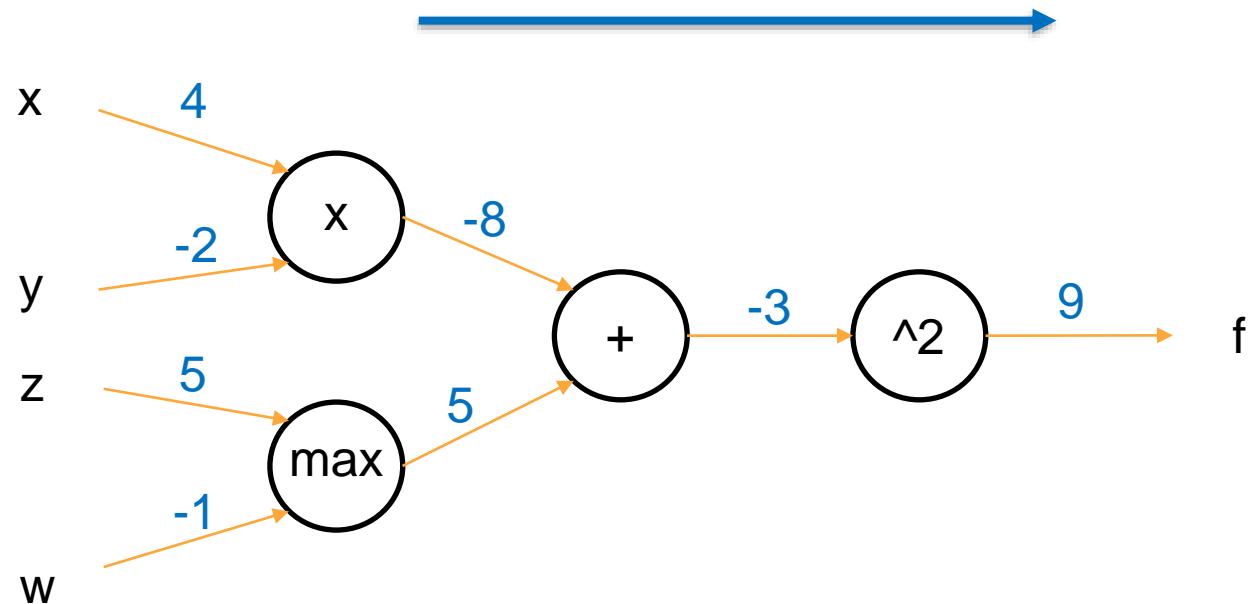
$$f(x, y, z, w) = (xy + \max(z, w))^2$$

Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$



# Beispiel: Backpropagation

$$f(x, y, z, w) = (xy + \max(z, w))^2$$



Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$

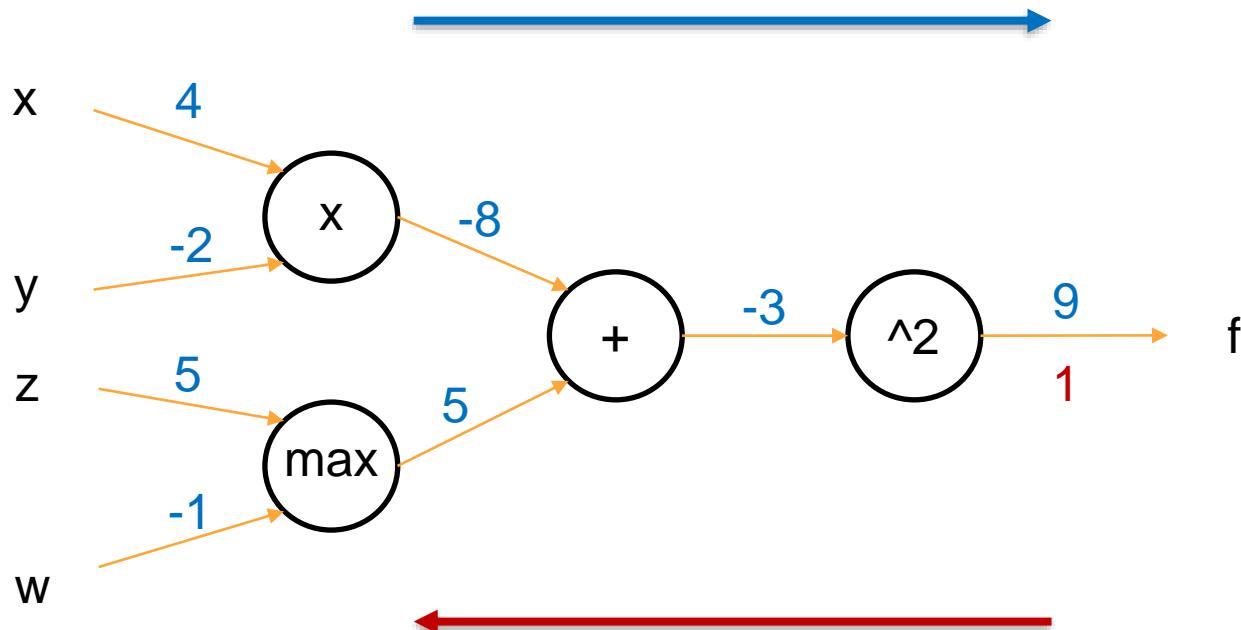
Backpropagation:  
Berechnet die Ableitung von f  
nach x, y, z, w

**Hinweis:** Beim Training  
neuronaler Netze nutzen wir das  
hier illustrierte Prinzip, um die  
Verlustfunktion nach den zu  
lernenden Parametern abzuleiten,  
insb. den Gewichten jeder Schicht

# Beispiel: Backpropagation

$$f(x, y, z, w) = (xy + \max(z, w))^2$$

Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$



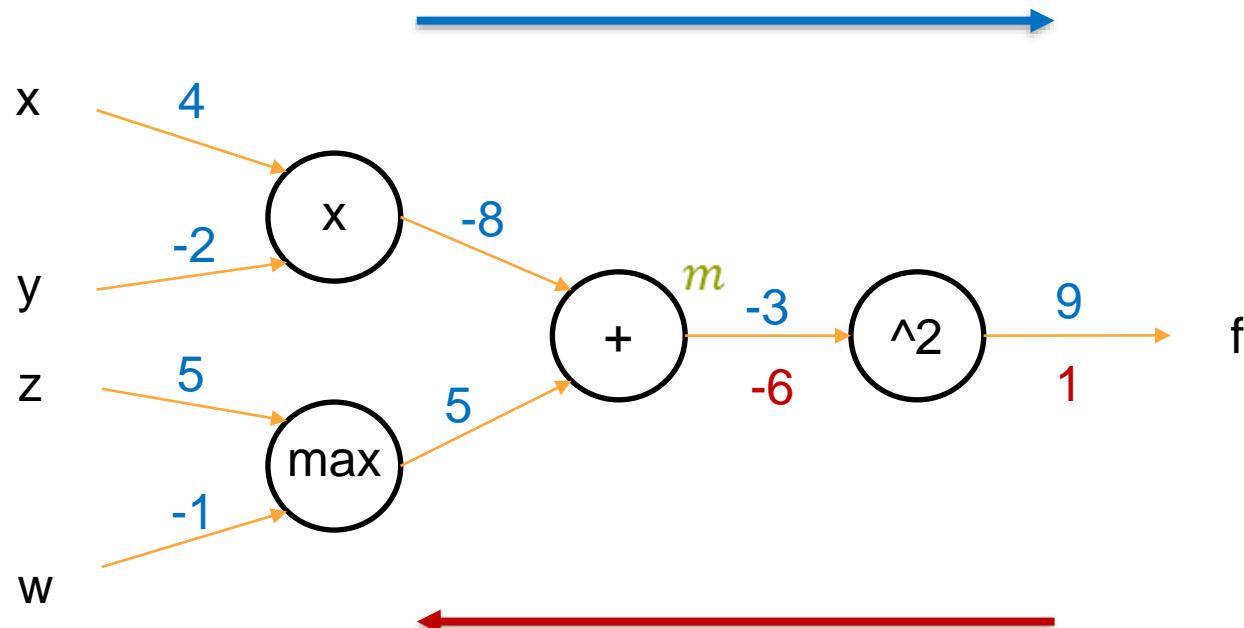
Backpropagation:  
Berechnet die Ableitung von  $f$   
nach  $x, y, z, w$

$$\frac{\partial f}{\partial f} = 1$$

# Beispiel: Backpropagation

$$f(x, y, z, w) = (xy + \max(z, w))^2$$

Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$



Backpropagation:  
Berechnet die Ableitung von  $f$   
nach  $x, y, z, w$

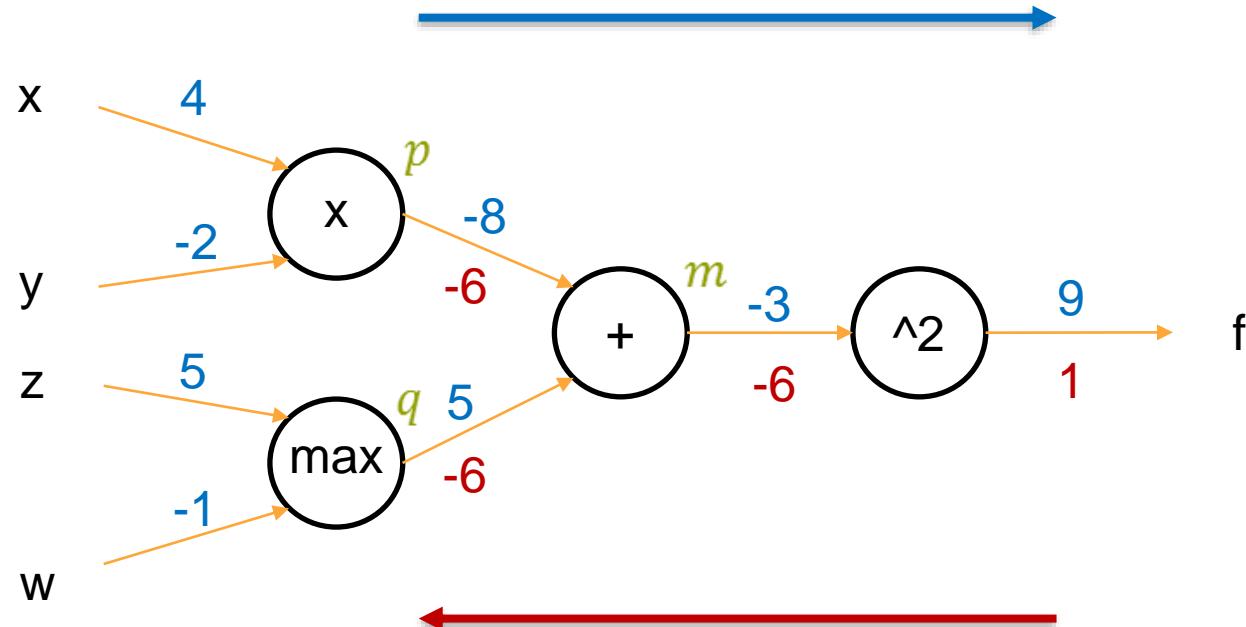
$$f = m^2$$

$$\frac{\partial f}{\partial m} = 2m$$

# Beispiel: Backpropagation

$$f(x, y, z, w) = (xy + \max(z, w))^2$$

Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$



Backpropagation:  
Berechnet die Ableitung von  $f$   
nach  $x, y, z, w$

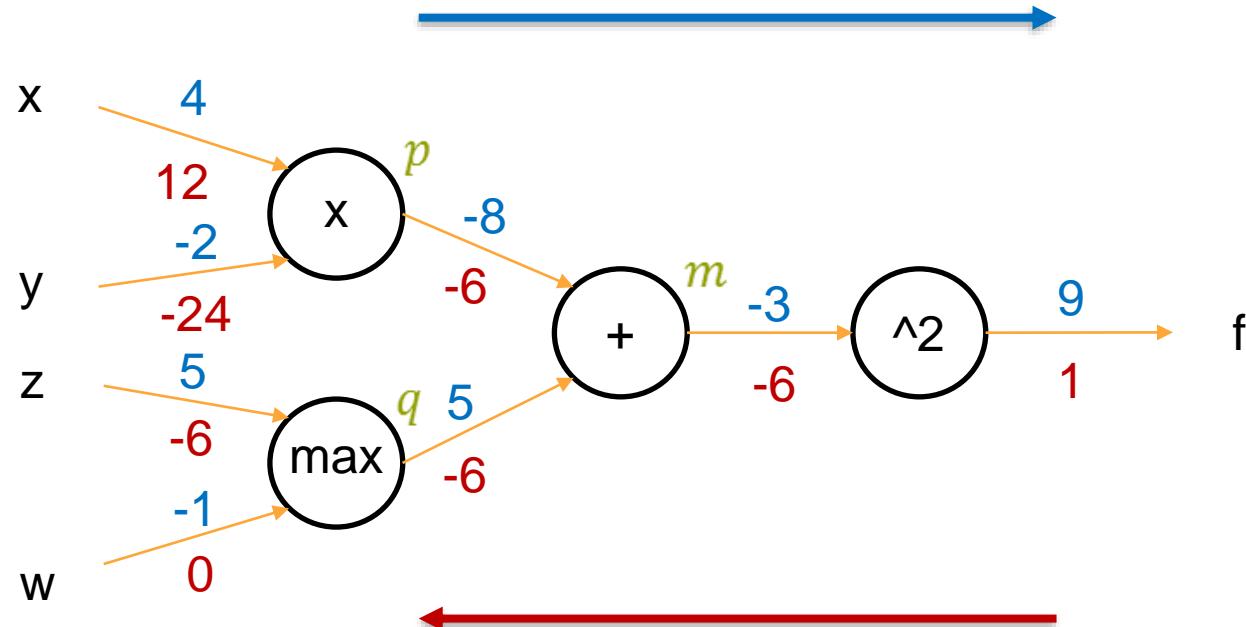
$$m = p + q$$

$$\frac{\partial f}{\partial p} = \frac{\partial f}{\partial m} \cdot \frac{\partial m}{\partial p}$$

Kettenregel

# Beispiel: Backpropagation

$$f(x, y, z, w) = (xy + \max(z, w))^2$$



Vorwärtspropagierung von  
 $x=4, y=-2, z=5, w=-1$

Backpropagation:  
Berechnet die Ableitung von  $f$   
nach  $x, y, z, w$

$$p = xy$$
$$q = \max(z, w)$$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial p} \cdot \frac{\partial p}{\partial x}, \quad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial p} \cdot \frac{\partial p}{\partial y}$$
$$\frac{\partial f}{\partial z} = \frac{\partial f}{\partial q} \cdot \frac{\partial q}{\partial z}, \quad \frac{\partial f}{\partial w} = \frac{\partial f}{\partial q} \cdot \frac{\partial q}{\partial w}$$

# Gradientenabstieg

- Ein **Gradientenabstieg** nutzt den per Backpropagation berechneten Gradienten  $\nabla_w L$  um die Gewichte  $w$  mit Lernrate  $\lambda$  in der Richtung anzupassen, die den Verlust  $L$  möglichst schnell verringert:

$$w += -\lambda \nabla_w L$$

- *Praktisches Problem:*
  - Die Berechnung der Verlustfunktion und ihrer Ableitungen ist sehr rechenaufwändig, da sie von allen Trainingsdaten abhängt
  - Gradientenabstieg macht relativ kleine Schritte und erfordert daher sehr häufige Auswertung

# Stochastischer Gradientenabstieg

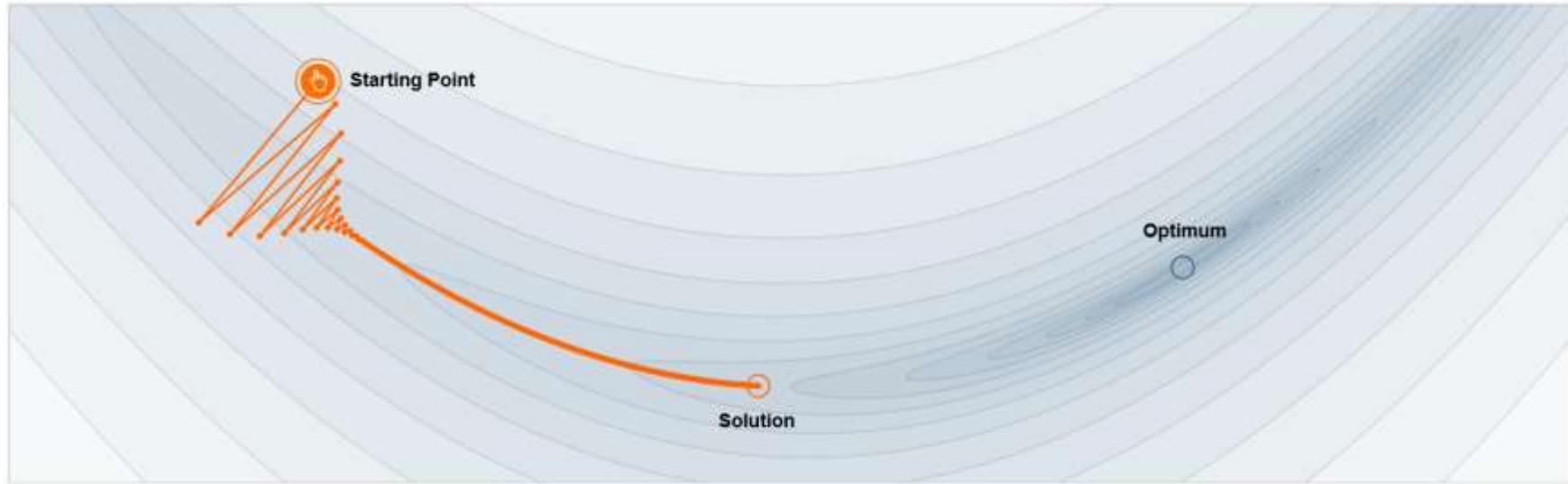
- **Stochastischer Gradientenabstieg** (SGD, engl. stochastic gradient descent) schätzt  $\nabla_w L$  in jedem Schritt mit einer zufälligen Teilmenge (“mini batch”) der Trainingsdaten
  - Richtung weniger zuverlässig, aber sehr viel schneller zu berechnen
- Eine **Epoche** bezeichnet eine bestimmte Zahl von Updates
  - *Traditionell*: Zahl der Trainingsdaten durch Minibatch-Größe, d.h. nach einer Epoche wurde (bei Ziehen ohne Zurücklegen) jedes Trainingsdatum einmal verarbeitet
  - *Manchmal auch*: Willkürlich festgelegte Zahl, z.B. 250 Updates

# Gradientenabstieg mit Trägheit / Momentum

- *Problem:* Unzuverlässige Gradientenrichtungen wegen SGD und/oder starker Krümmung der Verlustfunktion
- *Idee:* Glättung durch Berechnung eines gleitenden Mittels der letzten Gradientenrichtungen
  - Schritte werden größer, wenn Gradienten in dieselbe Richtung zeigen
  - Ermöglicht außerdem Überwinden von Regionen schwacher Gradienten
  - *Analogie:* Rollen eines Balls über die „Landschaft“ der Verlustfunktion
    - Impuls (*engl. momentum*) ist proportional zur Geschwindigkeit
    - Impuls ändert sich durch Reibung und Schwerkraft
- **Momentum-Update**  $\mathbf{w} += -\lambda \mathbf{v}$  mit Geschwindigkeit
$$\mathbf{v}^{(k)} := \mu \mathbf{v}^{(k-1)} + \nabla_{\mathbf{w}} L$$
  - Initialisierung mit  $\mathbf{v}^{(0)} = \mathbf{0}$ , Änderung bei jedem Update  $k$
  - Abschwächung  $0 \leq \mu < 1$  neben Lernrate  $\lambda$  weiterer Hyperparameter

# Illustration: Gradienten-Abstieg vs Momentum

$$\mathbf{w} += -0.003 \nabla_{\mathbf{w}} L$$



$$\mathbf{w} += -0.003 \mathbf{v}$$
$$\mathbf{v}^{(k)} := 0.8 \mathbf{v}^{(k-1)} + \nabla_{\mathbf{w}} L$$

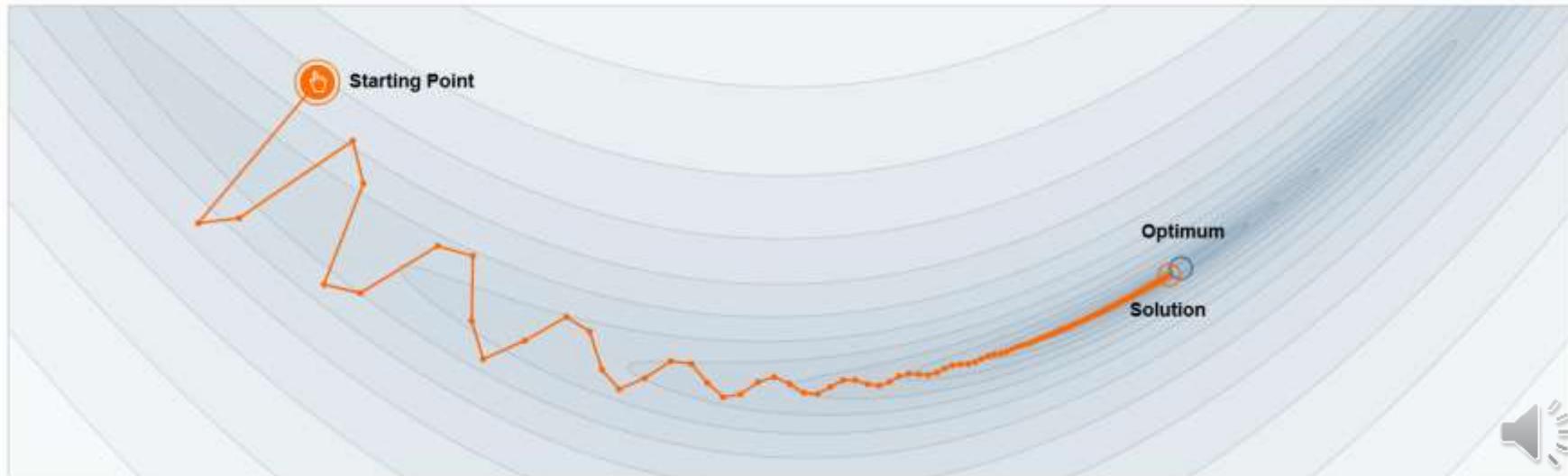
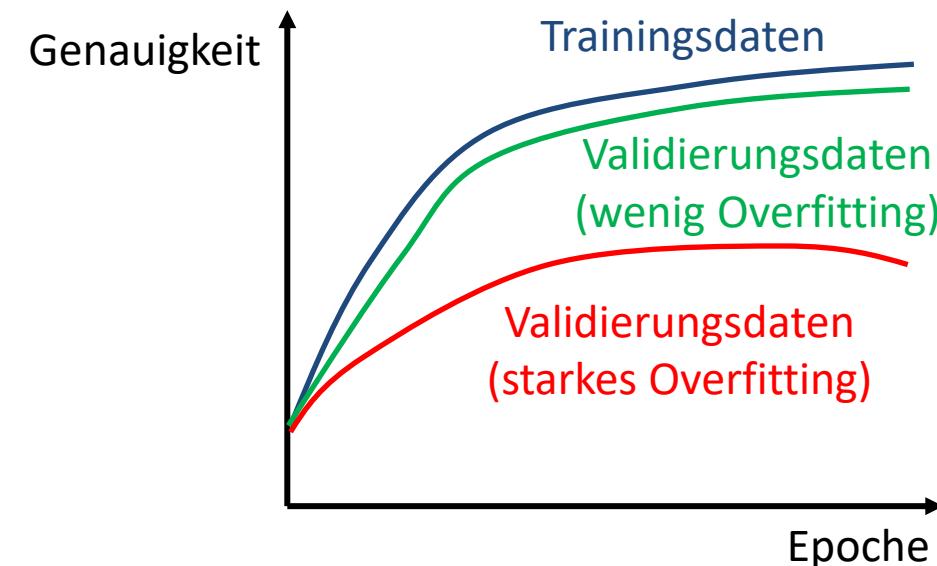
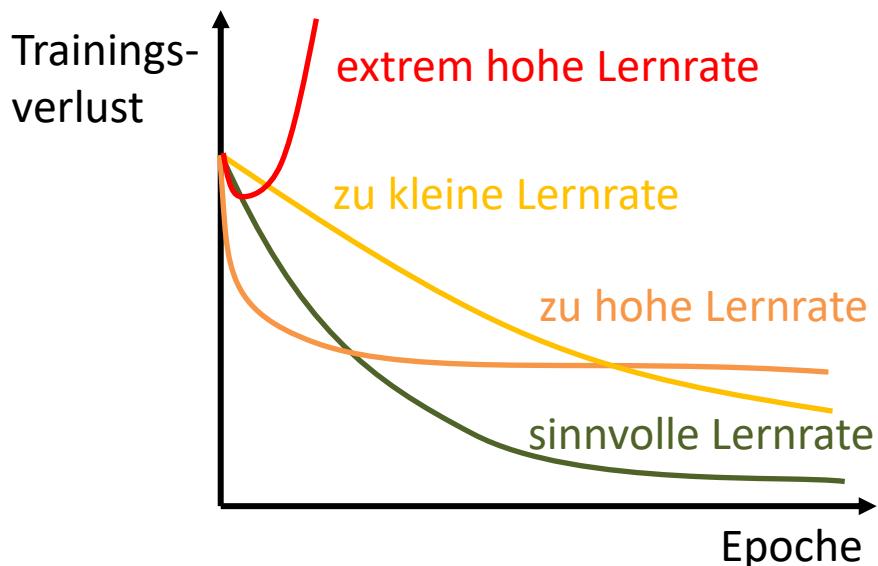


Illustration:  
<https://distill.pub/2017/momentum/>



# Überwachung des Lernprozesses

- Plotten des **Trainingsverlusts** zeigt, ob die Optimierung konvergiert oder Hyperparameter des Optimierers (z.B. Lernrate) angepasst werden müssen
  - extrem hohe Lernrate
  - zu kleine Lernrate
  - zu hohe Lernrate
  - sinnvolle Lernrate
- Ein Vergleich der Genauigkeit auf **Trainings-** und **Validierungsdaten** zeigt eine mögliche Überanpassung (**overfitting**)



# Anpassung der Lernrate

- Die anfängliche Lernrate  $\lambda$  wird im Laufe des Trainings üblicherweise immer weiter verringert
- Verbreitete **Lernratenpläne** (engl. learning rate schedule):
  - **Schrittweise** Reduzierung um einen bestimmten Faktor, nach einer festen Zahl von Epochen oder wenn der Validierungsverlust stagniert
  - **Lineare** Reduzierung von einer anfänglichen Lernrate  $\lambda_0$  auf eine ab der Iteration  $k=T$  gültige finale Lernrate  $\lambda_T$

$$\lambda_k = (1 - \alpha)\lambda_0 + \alpha\lambda_T \text{ mit } \alpha = \min\left(\frac{k}{T}, 1\right)$$

- **Exponentielle** Reduzierung mit Zerfallsparameter  $\gamma$

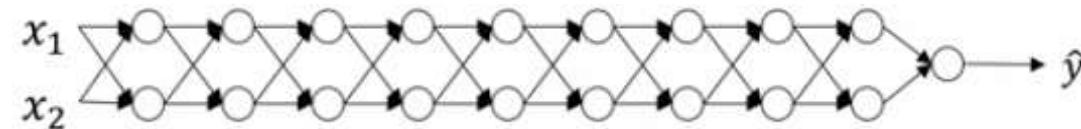
$$\lambda_k = \lambda_0 e^{-\gamma k}$$

# Adaptive Lernraten: AdaGrad und Adam

- *Beobachtung:* Das Update  $\mathbf{w} += -\lambda \nabla_{\mathbf{w}} L$  verändert Parameter  $w_i$  mit einem höheren Einfluss auf die Verlustfunktion stärker
  - *Konsequenz:* Höchste Lernrate  $\lambda$  mit stabilem Training wird durch die „empfindlichsten“ Neuronen beschränkt. Andere Neuronen benötigen für angemessenen Fortschritt evtl. ein höheres  $\lambda$
- **AdaGrad** berechnet Parameter-spezifische Lernraten
- **Adam** ist ein beliebtes Optimierungsverfahren für tiefe neuronale Netze. Es kombiniert adaptive Lernraten mit Momentum-Updates

# Initialisierung der Gewichte flacher Netze

- **Mit Null:** Würde zu identischen Gradienten und Updates führen und ermöglicht daher kein sinnvolles Training!
  - Aber: Bias-Parameter  $b_k$  werden häufig auf Null initialisiert
- **Kleine Zufallszahlen**, z.B.  $\mathcal{N}(0,0.01^2)$ . Funktioniert für flache Netze, aber problematisch in tiefen Netzen
  - Vernachlässigen wir die Aktivierungsfunktion, multiplizieren sich die Gewichtsmatrizen aller Schichten:



$$\hat{y} = W_{l-1}W_{l-2} \dots W_2W_1X$$

- Bei zu kleinen Gewichten „sterben“ die Aktivierungen aus
- Erhöhung der Varianz birgt die Gefahr „explodierender“ Aktivierungen

# Initialisierung der Gewichte tiefer Netze

- **Xavier et al.** initialisieren Gewichte eines Neurons mit  $\mathcal{N}(0,1/m)$ , wobei  $m$  die Zahl der Eingaben ist
  - *Idee:* Das Neuron berechnet  $f\left(\sum_{j=1}^m w_{kj}x_j + b_k\right)$
  - Die Varianz sollte von Schicht zu Schicht erhalten bleiben. Bei der Addition unkorrelierter Zufallsvariablen addieren sich die Varianzen
- **He et al.** berücksichtigen außerdem den Effekt der Aktivierungsfunktion
  - Bei Verwendung von ReLU empfehlen sie daher Initialisierung mit  $\mathcal{N}(0,2/m)$

# Batch-Normalisierung

- Optimierung ändert alle Schichten des Netzwerks gleichzeitig
  - *Problem:* Durch das Update früherer Schichten ändern sich die Eingaben späterer Schichten. Deren Updates sind dadurch nicht mehr optimal.
  - Batch-Normalisierung **entkoppelt** die einzelnen Schichten
    - Aktivierungen  $\mathbf{H}$  jeder Schicht werden zunächst standardisiert (Mittelwert 0, Varianz 1), dann mit zusätzlichen Parametern  $\beta$  und  $\gamma$  verschoben und skaliert
    - Mittelwert und Varianz der Aktivierungen hängen direkt von  $\beta$  und  $\gamma$  ab, nicht mehr vom komplexen Zusammenspiel aller vorherigen Gewichte
- *Vorteile:* Ermöglicht höhere Lernraten, reduzierte Abhängigkeit von der Initialisierung, häufig bessere Ergebnisse
- *Hinweis:* Zusätzlich zur Batch-Normalisierung ist es üblich die Eingaben des Netzwerks zu standardisieren

# Regularisierung

Um tiefe neuronale Netze zu **regularisieren** und Überanpassung (overfitting) zu reduzieren nutzt man u.a.

- **Bestrafung großer Parameterwerte** durch modifizierte Zielfunktion:

$$L = \underbrace{\frac{1}{N} \sum_i L_i}_{\text{Datenterm}} + \underbrace{\nu R(W)}_{\text{Regularisierer}}$$

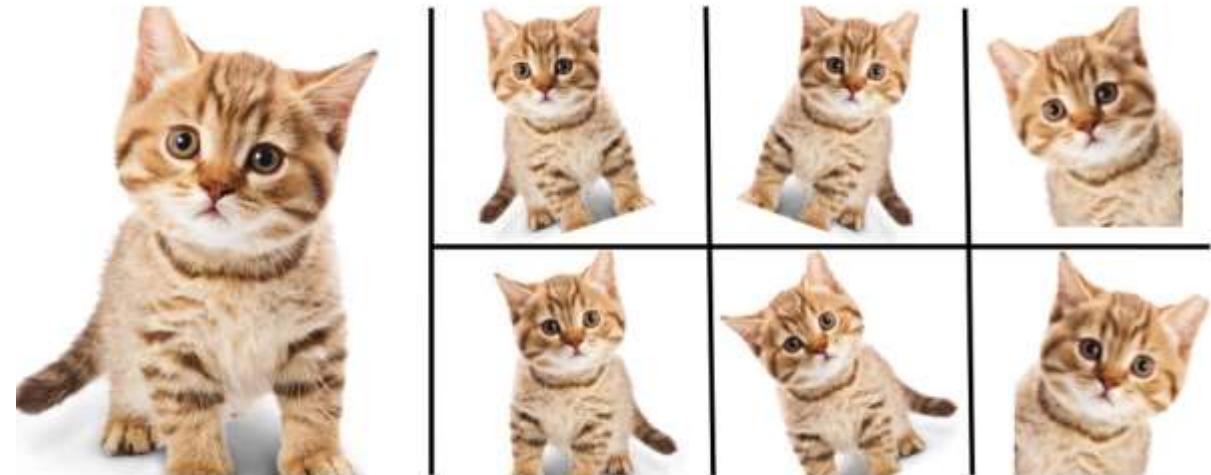
z.B. mit  $R(W) = \sum_l \sum_k \sum_j w_{lkj}^2$

–  $w_{lkj}$  ist das  $j$ -te Gewicht des  $k$ -ten Neurons der  $l$ -ten Schicht

- **Early Stopping:** Nutzt die Parameter der Epoche, nach der der Validierungsfehler am geringsten war, selbst wenn der Trainingsfehler danach weiter gesunken ist
- **Dropout:** Zufälliges Auslassen von Neuronen während des Trainings mit einstellbarer Wahrscheinlichkeit  $p$ 
  - Wird bei der Anwendung auf Testdaten kompensiert, indem die Ausgaben verborgener Schichten mit  $p$  skaliert werden

# Datenaugmentierung

- Ein **größerer Trainingsdatensatz** ist häufig das erfolgreichste Mittel um eine bessere Generalisierung zu erreichen
  - ...aber die Aufnahme/Annotation ist leider meist sehr aufwändig
- **Datenaugmentierung** erzeugt künstlich zusätzliche Daten
  - modifiziert vorhandene Bilder so, dass die Labels gültig bleiben oder automatisch angepasst werden können
  - Beispiele:
    - Verschiebung, Rotation
    - Ausschnitte, Spiegelung
    - (Leichte) Deformationen
    - Farb- und Kontraständerungen



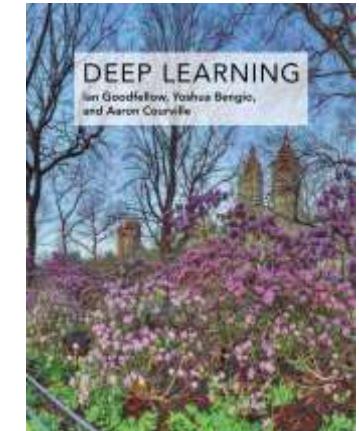
# Zusammenfassung

Das **Training neuronaler Netze** erfordert

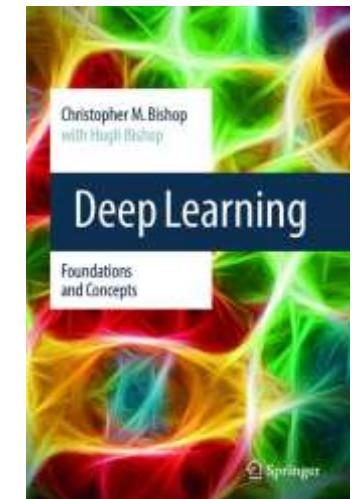
- die Wahl einer geeigneten **Verlustfunktion**
- die Berechnung ihres Gradienten bezüglich der Netzwerk-Parameter mittels **Backpropagation**
- die Wahl eines geeigneten **Optimierers** und seiner Hyperparameter, insbesondere **Lernrate** und **Lernratenplan**
  - *beliebte Tricks:* **Momentum** und **adaptive Lernraten**
  - **Batch-Normalisierung** vereinfacht die Suche nach geeigneten Parametern
- die Überwachung des Lernprozesses und den Einsatz geeigneter **Regularisierung** und **Augmentierung**

# Zum Nach- und Weiterlesen

- Ian Goodfellow, Yoshua Bengio, Aaron Courville:  
“Deep Learning.” MIT Press, 2016  
<https://www.deeplearningbook.org/>



- Christopher Bishop with Hugh Bishop:  
“Deep Learning. Foundations and Concepts.”  
Springer, 2024  
<https://www.bishopbook.com/>



# Kapitel 6b: Bildanalyse mit CNNs

Prof. Dr.-Ing. Thomas Schultz

URL: <http://cg.cs.uni-bonn.de/schultz/>

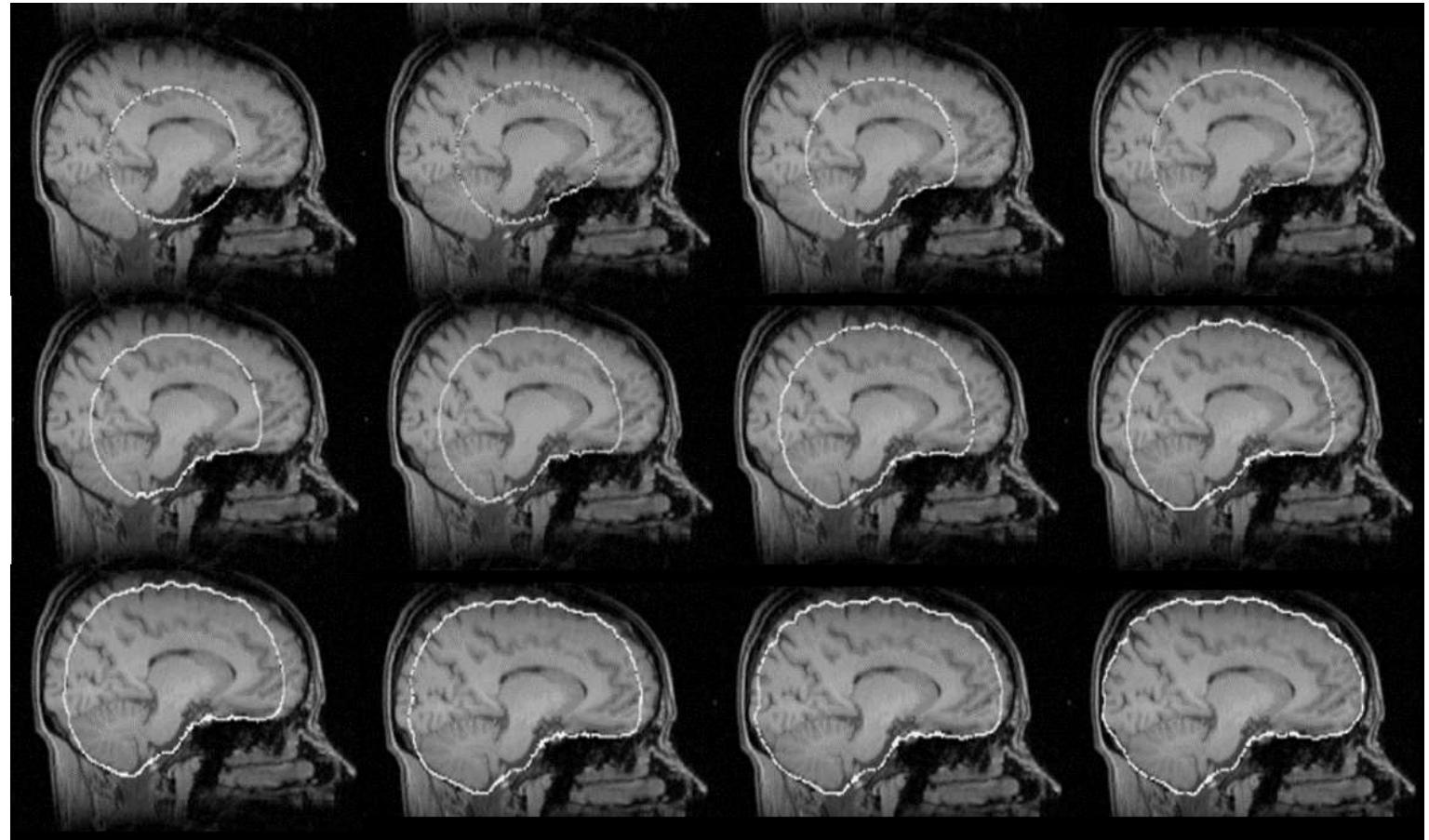
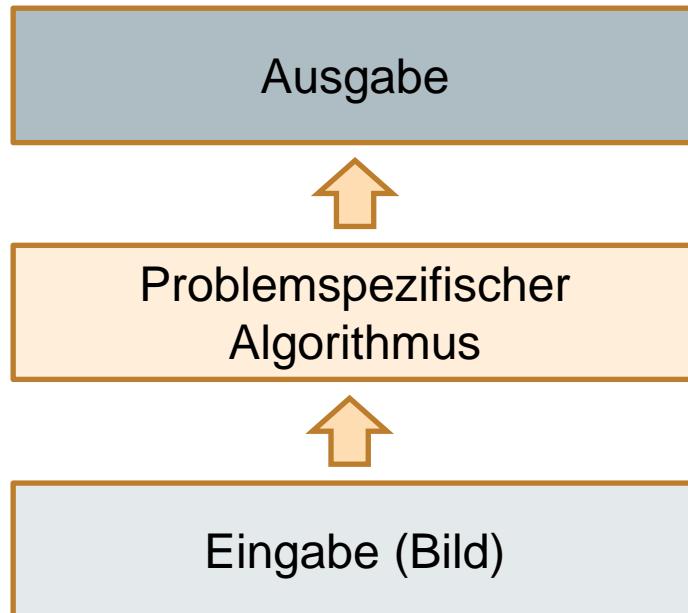
E-Mail: [schultz@cs.uni-bonn.de](mailto:schultz@cs.uni-bonn.de)

Büro: Friedrich-Hirzebruch-Allee 6, Raum 2.117

13./20./27. Januar 2025

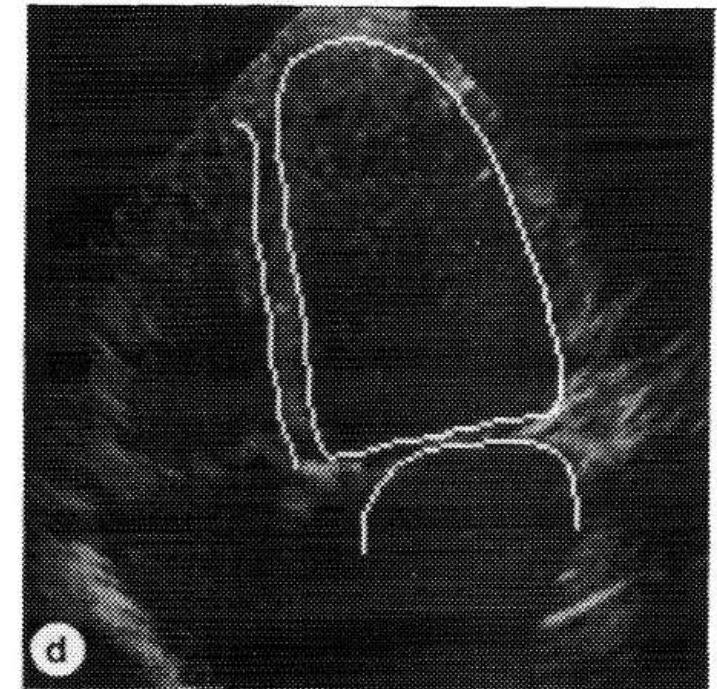
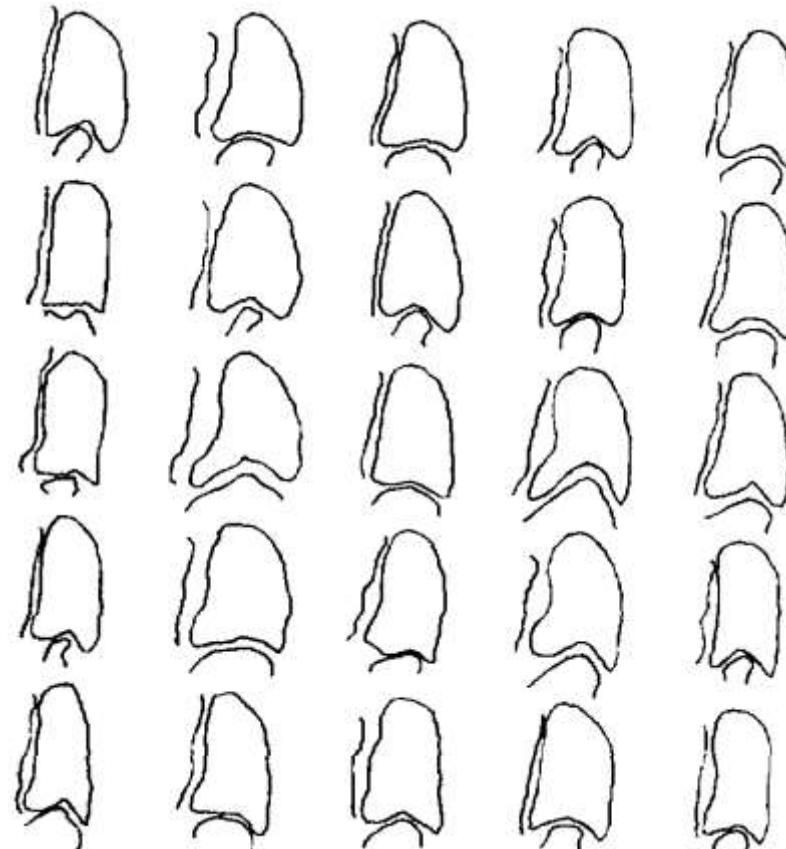
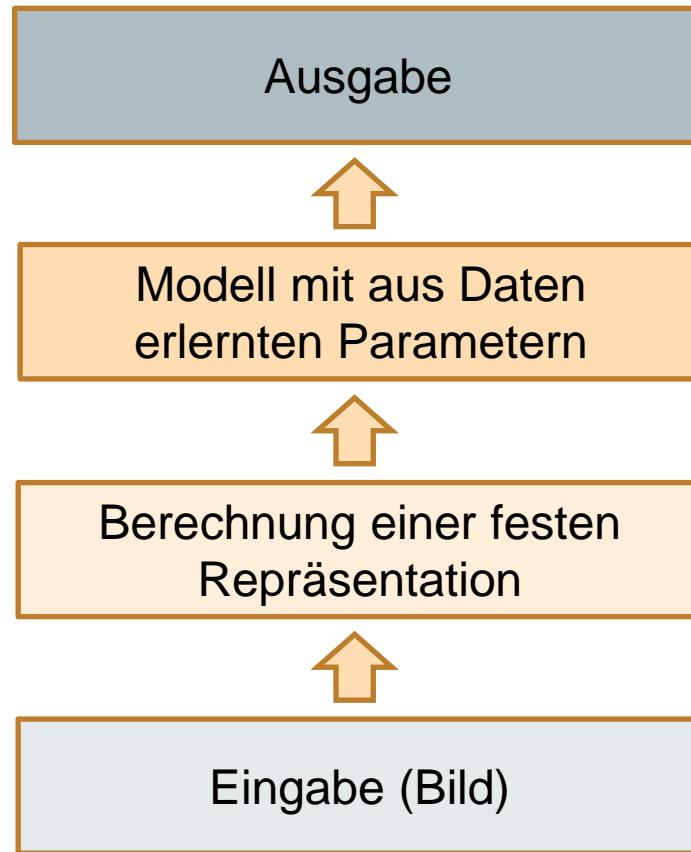
## **6b.1 Einführung**

# Regelbasierte Bildanalyse



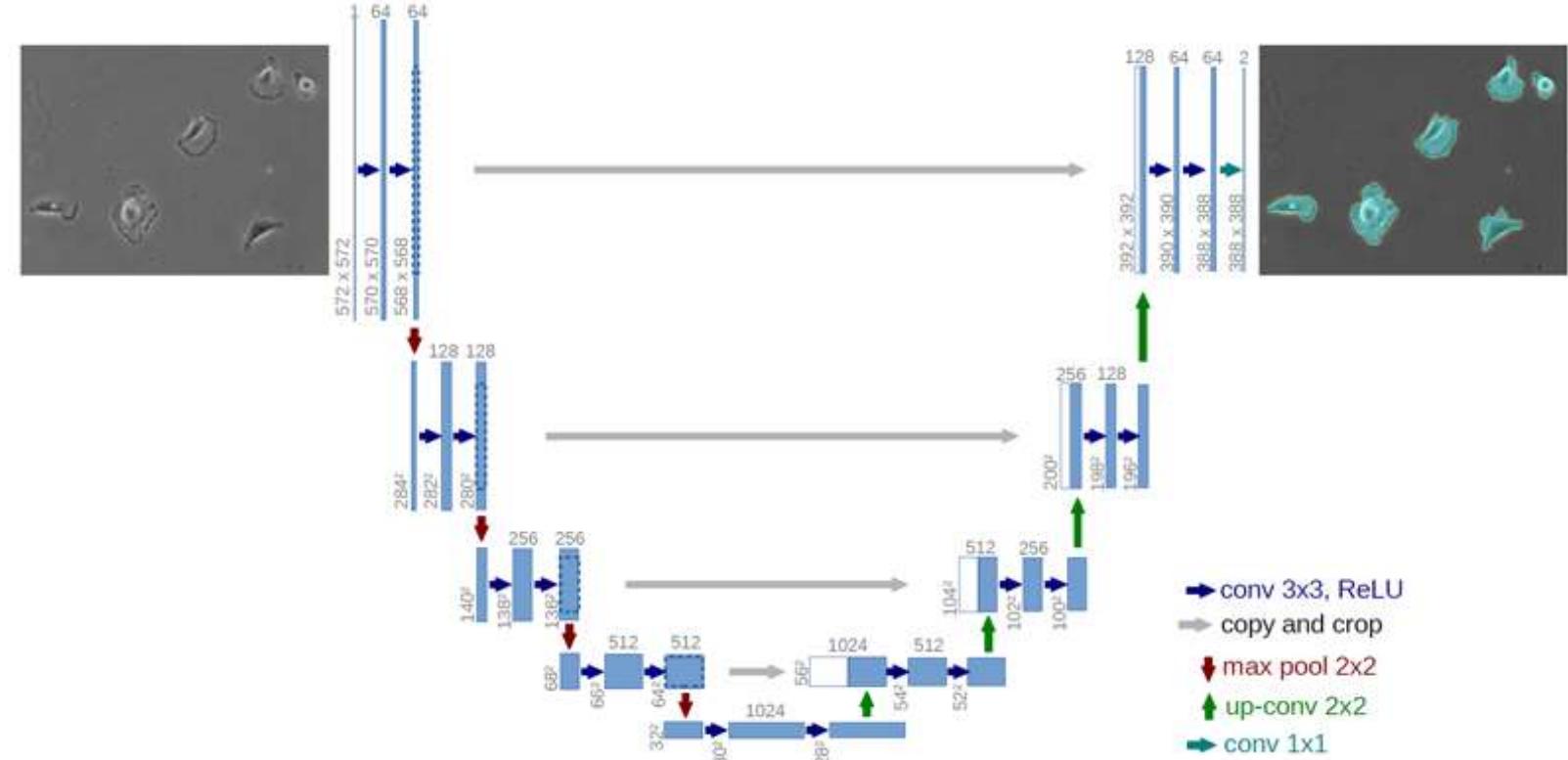
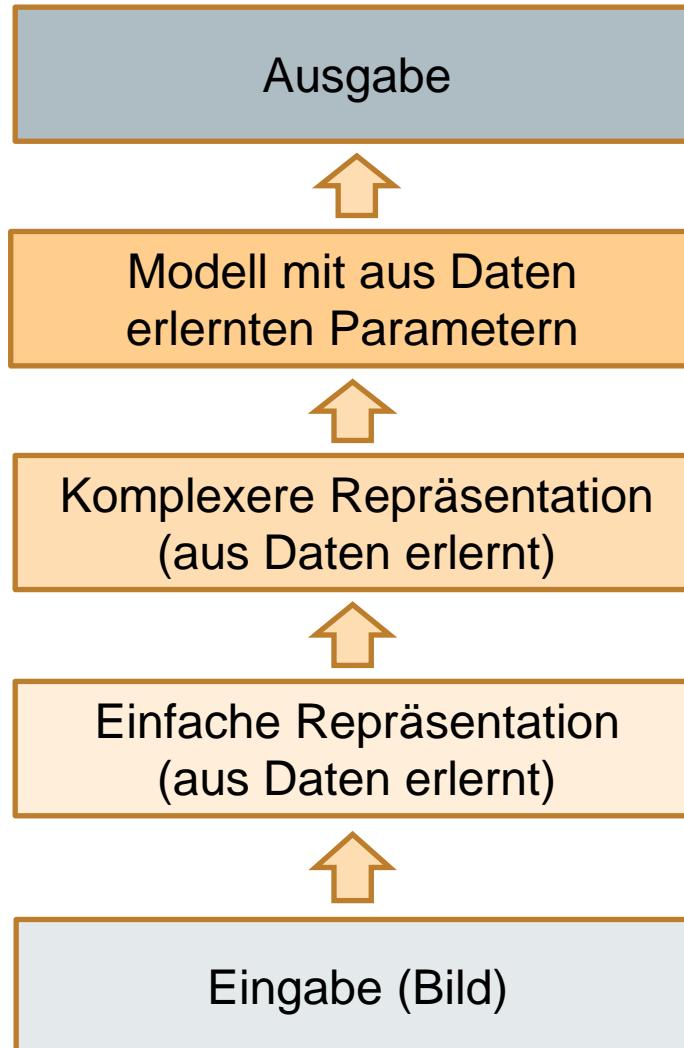
Bildquelle: S. Smith: *Fast Robust Automated Brain Extraction*. Human Brain Mapping 17:143-155, 2002

# Lernbasierte Bildanalyse



Bildquelle: Cootes et al., *Active Shape Models – Their Training and Application*. Computer Vision and Image Understanding, 1995

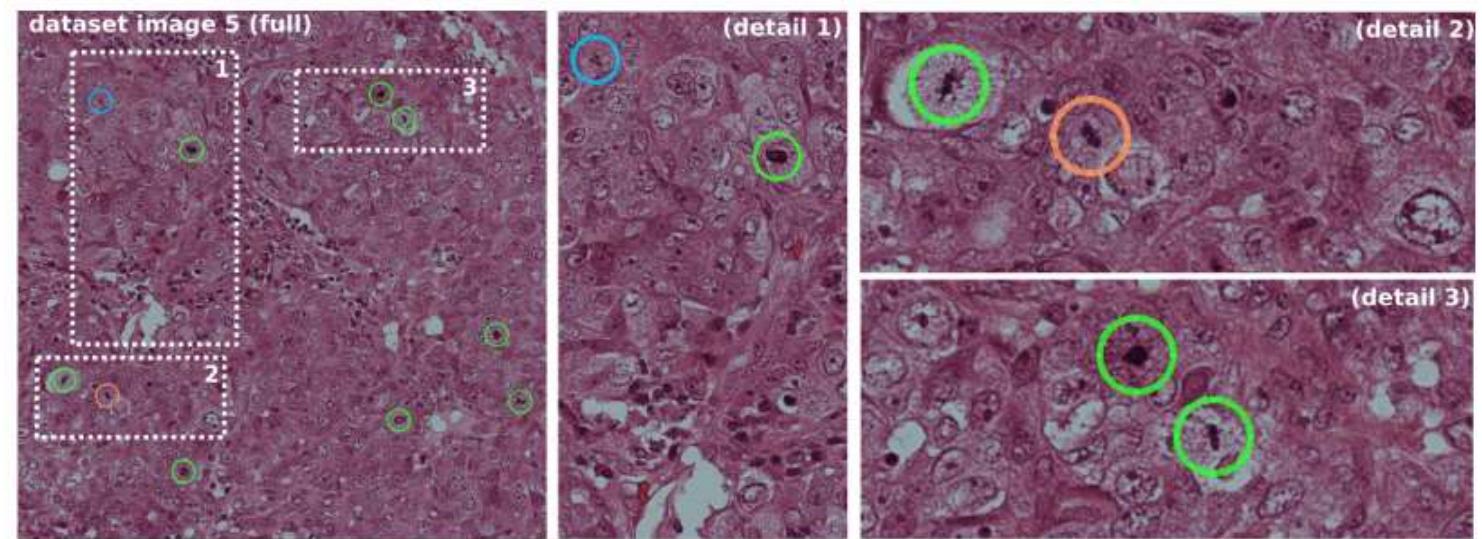
# Bildanalyse mit Tiefem Lernen



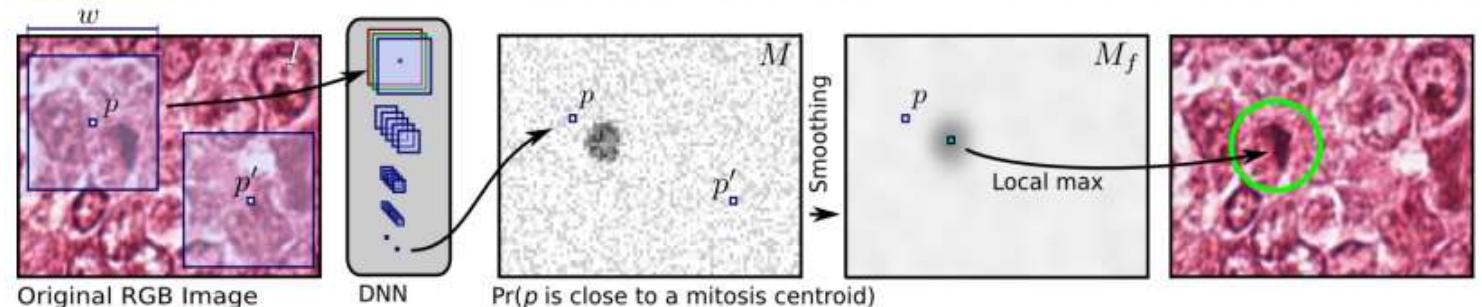
Bildquelle: O. Ronneberger, P. Fischer, T. Brox: *U-net: Convolutional networks for biomedical image segmentation*. MICCAI 2015.

# Beispiel: Erkennung von Mitosen

**Meilenstein 2012a:** Erstmals schlägt ein tiefes neuronales Netz in einem großen Wettbewerb zur *Objekterkennung* (Mitosen in histologischen Schnitten) mit Abstand alle traditionellen Verfahren

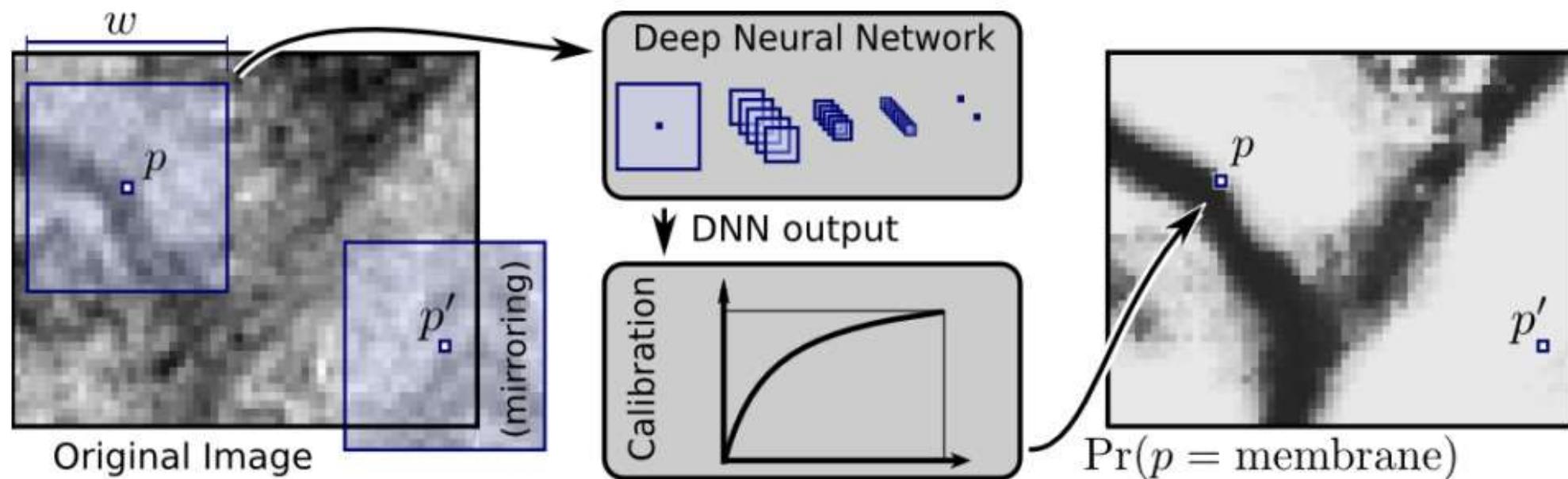


*Bildquelle:* Dan C. Cireşan et al., „Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks“ Medical Image Computing and Computer Assisted Intervention 2013, pp. 411-418



# Beispiel: Segmentierung von Zellmembranen

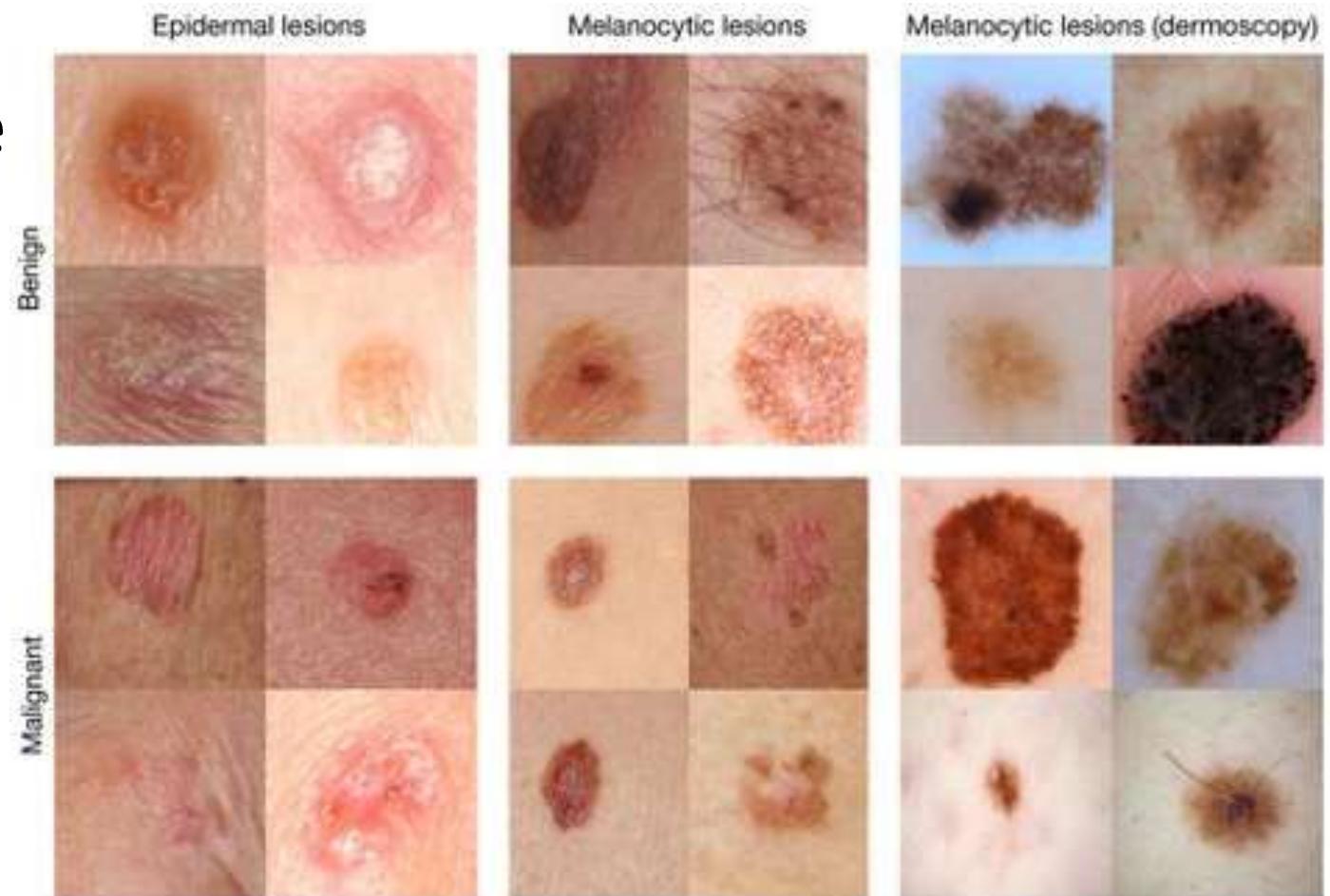
**Meilenstein 2012b:** Erstmals schlägt ein tiefes neuronales Netz in einem großen Wettbewerb zur *Bildsegmentierung* (Zellmembranen in elektronenmikroskopischen Bildern) alle traditionellen Verfahren



*Bildquelle:* Dan C. Cireşan et al., „Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images“ Proc. NeurIPS 2012, pp. 2852-2860

# Beispiel: Erkennung von Hautkrebs

[Esteva et al. 2017] berichten mittels  $\approx 130.000$  Fotos ein neuronales Netz trainiert zu haben, das zwei Typen von Hautkrebs so zuverlässig erkenne wie 21 Fachärzte für Dermatologie

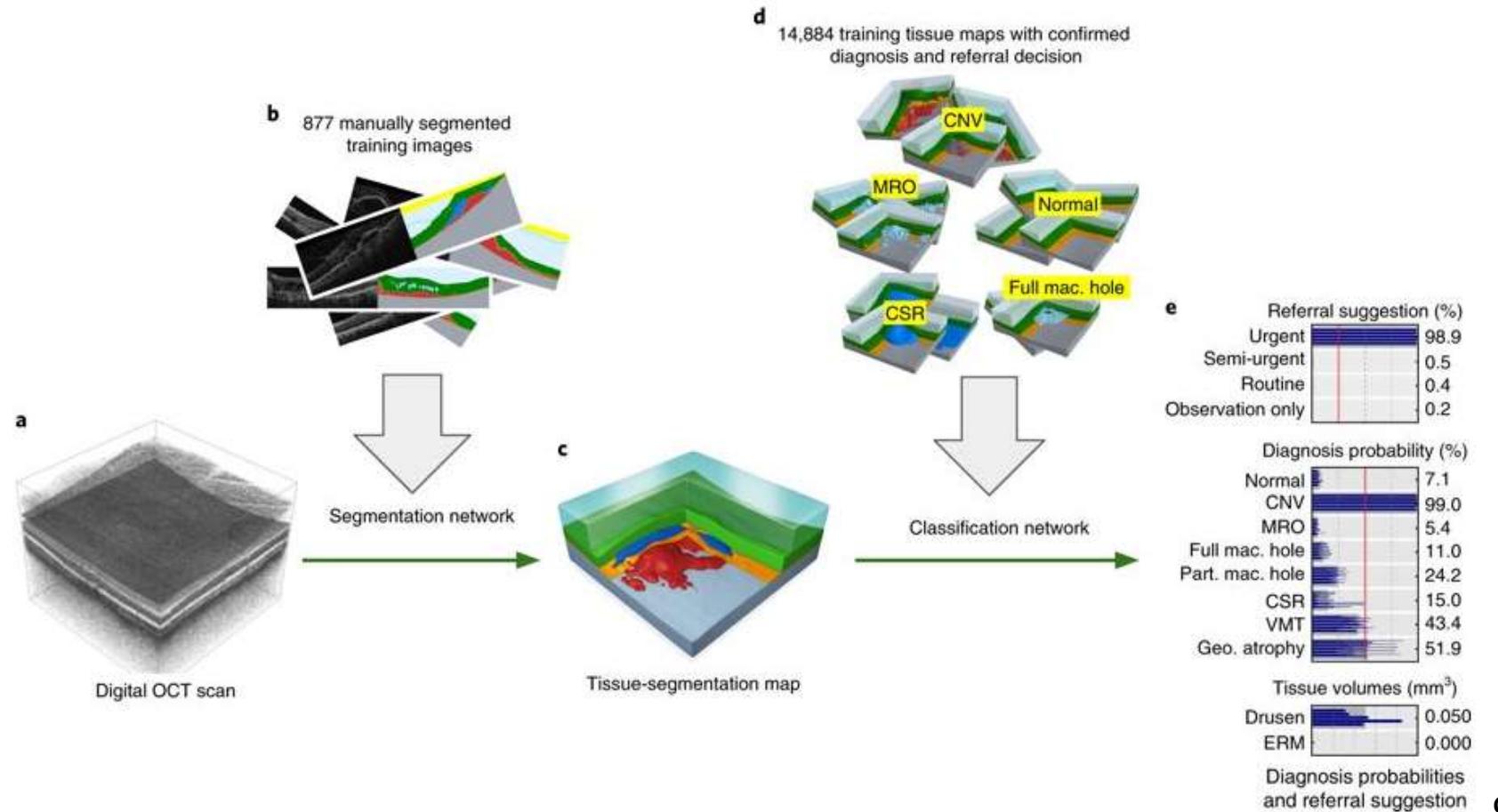


Bildquelle: Andre Esteva et al.,  
„Dermatologist-level classification of  
skin cancer with deep neural  
networks“ Nature 542:115-118, 2017

# Beispiel: Ersteinschätzung und Diagnose

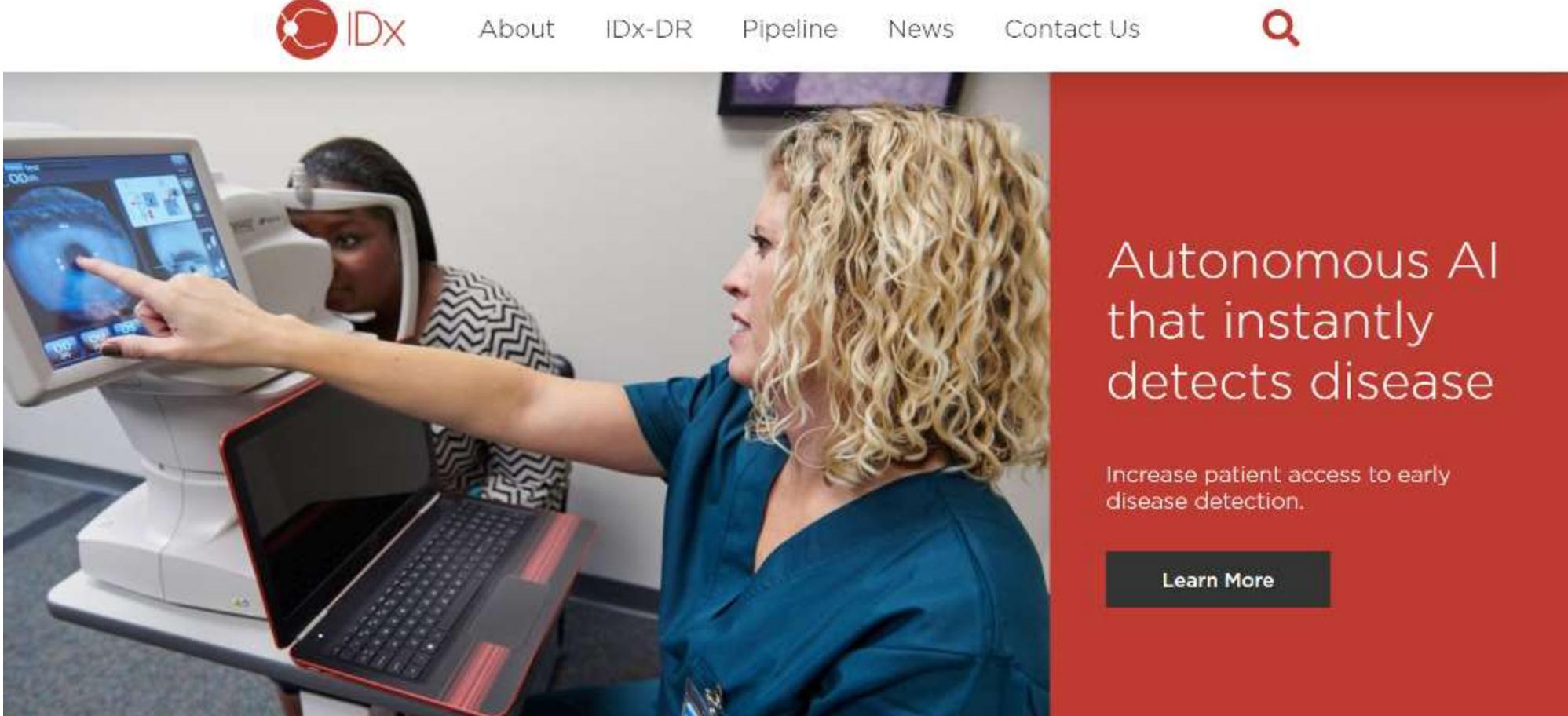
[De Fauw et al. 2018] berichten, dass ein neuronales Netz aus nur  $\approx 14.000$  OCT-Scans gelernt habe, die Dringlichkeit einer Behandlung so zuverlässig einzuschätzen wie klinische Experten

Bildquelle: Jeffrey De Fauw et al., „Clinically Applicable Deep Learning for Diagnosis and Referral in Retinal Disease“ Nature Medicine 24:1342-1350, 2018



# Beispiel: Praktischer Einsatz

**Meilenstein 2018:** US-amerikanische FDA lässt mit IDx-DR erstmals ein autonomes diagnostisches System zu



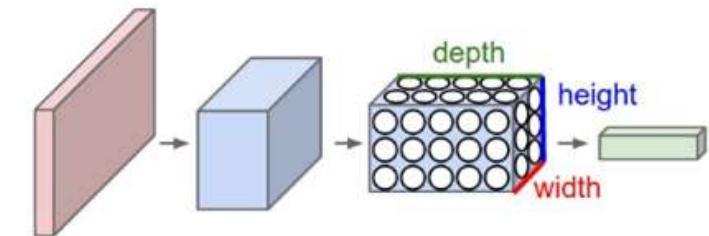
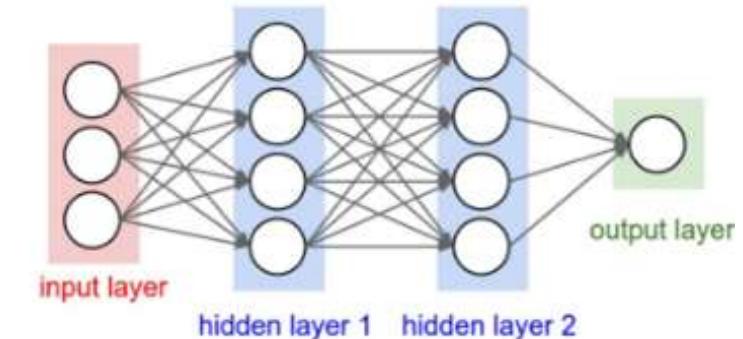
The image shows a screenshot of the IDx website. At the top, there is a navigation bar with the IDx logo, About, IDX-DR, Pipeline, News, Contact Us, and a search icon. Below the navigation bar is a photograph of a medical professional with curly blonde hair pointing at a touchscreen monitor of an IDx-DR machine. The machine has a built-in laptop. In the background, another person wearing a head-mounted display is also interacting with the machine. To the right of the photograph is a red sidebar with white text. The top part of the sidebar reads "Autonomous AI that instantly detects disease". Below that, it says "Increase patient access to early disease detection." At the bottom of the sidebar is a "Learn More" button.

<https://www.eyediagnosis.co/>

## **6b.2 CNNs – Faltende Netzwerke**

# Convolutional Neural Networks

- *Problem:* Analyse von Bildern realistischer Größe (nur) mit vollständig verbundenen Schichten ist nicht sinnvoll
  - Hohe Zahl von Gewichten erzeugt enormen Speicher- und Rechenaufwand und benötigt sehr große Trainingsdatensätze
- **Faltende neuronale Netze** (*engl.* convolutional neural networks, CNNs) verarbeiten Bilder u.a. durch Faltung mit erlernten Kernen
  - Verringert die Zahl der Modellparameter drastisch
  - Äquivariant bzgl. Verschiebungen: Verschiebung der Eingabe führt zu entsprechender Verschiebung der Ausgabe
    - Objekte werden unabhängig davon erkannt, wo im Bild sie sich befinden



# Klarstellung: Faltung vs. Kreuzkorrelation

Aus Kapitel 1: Faltung

$$f(i, j) = (h * g)(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k h(u, v) \cdot g(i - u, j - v)$$

und Kreuzkorrelation

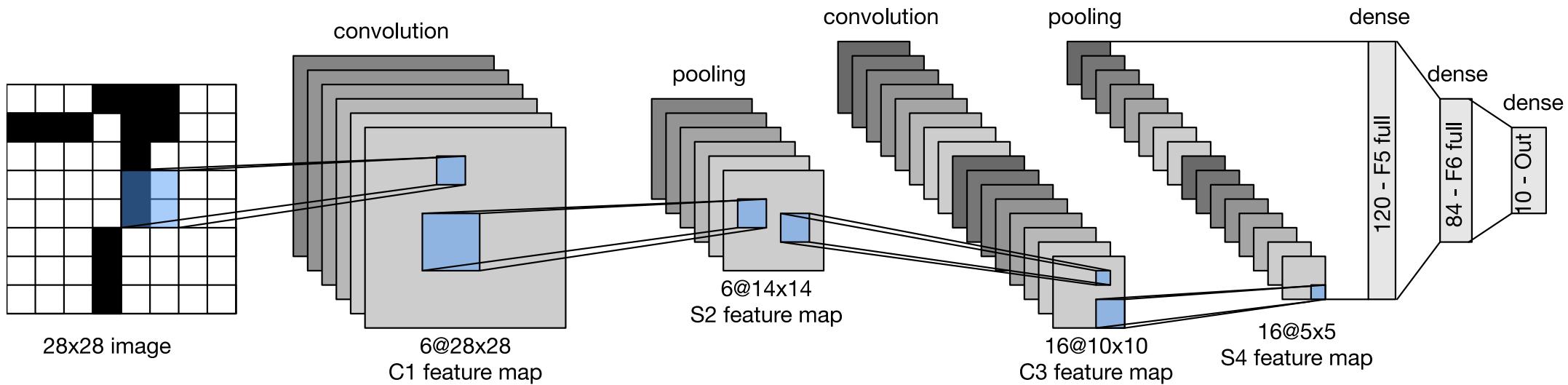
$$f(i, j) = (h \otimes g)(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k h(u, v) \cdot g(i + u, j + v)$$

unterscheiden sich durch die Vorzeichen

- Entspricht Spiegelung des Kerns in beiden Richtungen
- CNNs nutzen häufig Kreuzkorrelation, sprechen aber von „Faltung“
  - Streng genommen ist das Missbrauch der Terminologie
  - Solange die Kerne mit derselben Architektur gelernt werden, wird die Spiegelung „mitgelernt“, Unterschied daher praktisch nicht relevant

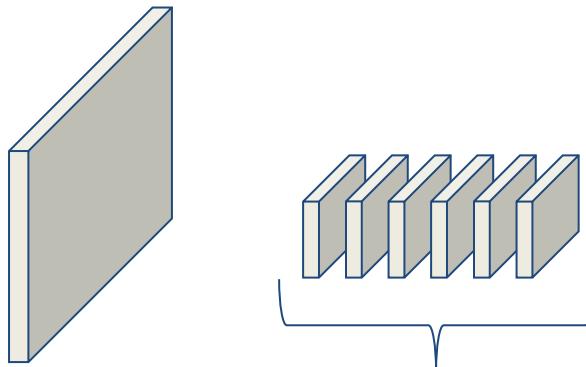
# Bausteine von CNNs

- LeNet, das erste CNN zur Unterscheidung handgeschriebener Ziffern [LeCun et al. 1998], enthält bereits wesentliche Bausteine:
  - Faltungsschichten
  - Pooling zum Downsampling
  - Vollständig verbundene Schichten am Ende des Netzwerks



# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

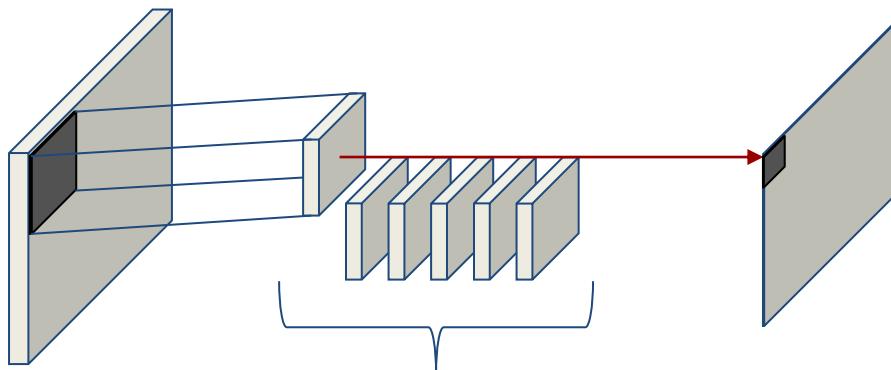


Eingabe  
32x32x3      Faltungskerne  
                Filtergröße: 5x5x3      Zahl der Filter: 6

Die Größe von Faltungskernen wird üblicherweise in 2D angegeben (hier: 5x5)  
Die dritte Dimension deckt implizit alle Kanäle der Eingabe ab.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

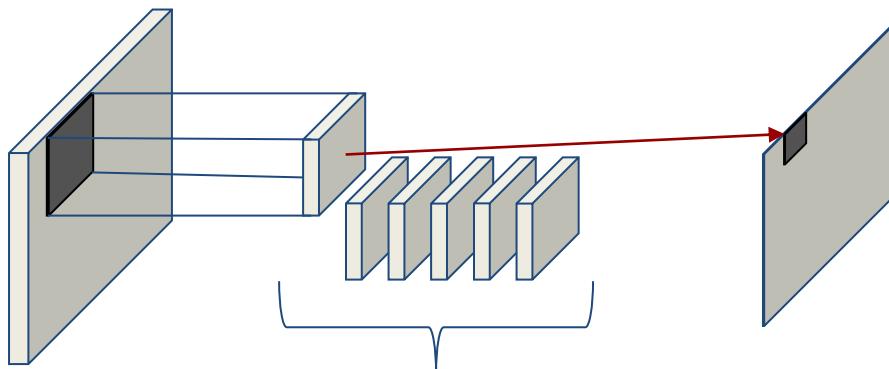


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

Die Faltung der Eingabe mit einem der Faltungskerne ergibt eine Aktivierungskarte. Ohne weitere Behandlung der Ränder ist diese kleiner als die Eingabe.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

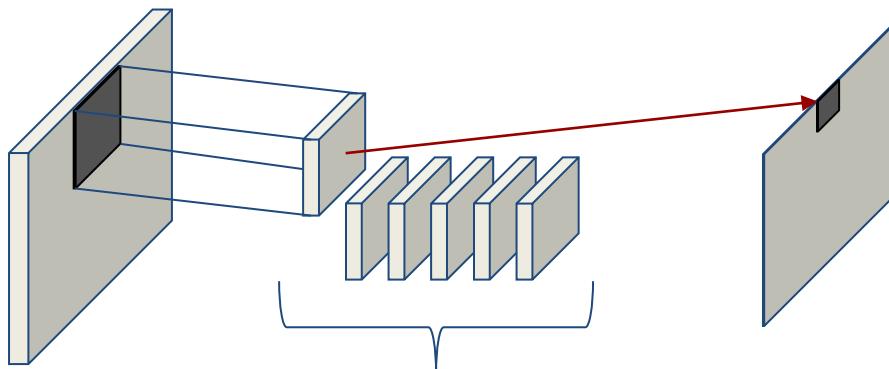


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

Die Faltung der Eingabe mit einem der Faltungskerne ergibt eine Aktivierungskarte. Ohne weitere Behandlung der Ränder ist diese kleiner als die Eingabe.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

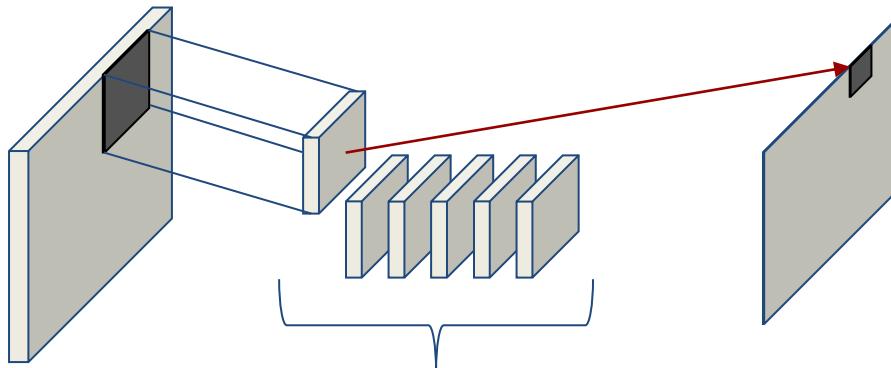


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

Die Faltung der Eingabe mit einem der Faltungskerne ergibt eine Aktivierungskarte. Ohne weitere Behandlung der Ränder ist diese kleiner als die Eingabe.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

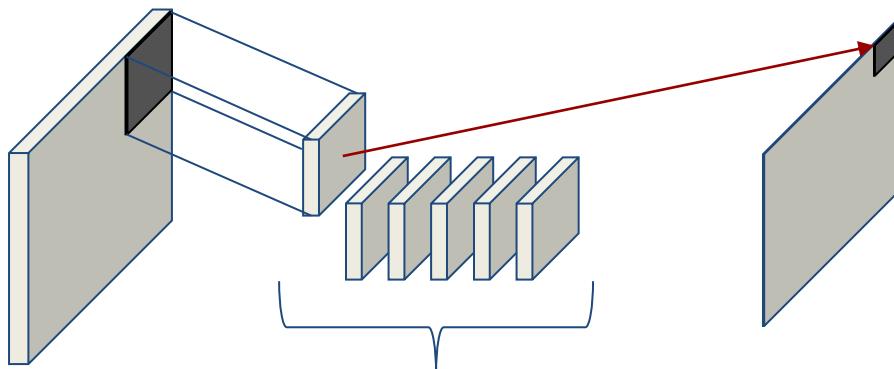


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3	28x28x1
	Zahl der Filter: 6	

Die Faltung der Eingabe mit einem der Faltungskerne ergibt eine Aktivierungskarte. Ohne weitere Behandlung der Ränder ist diese kleiner als die Eingabe.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

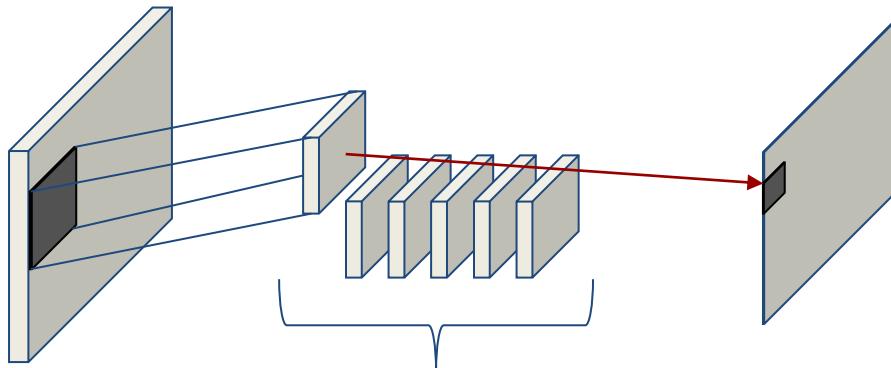


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3	28x28x1
	Zahl der Filter: 6	

Die Faltung der Eingabe mit einem der Faltungskerne ergibt eine Aktivierungskarte. Ohne weitere Behandlung der Ränder ist diese kleiner als die Eingabe.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

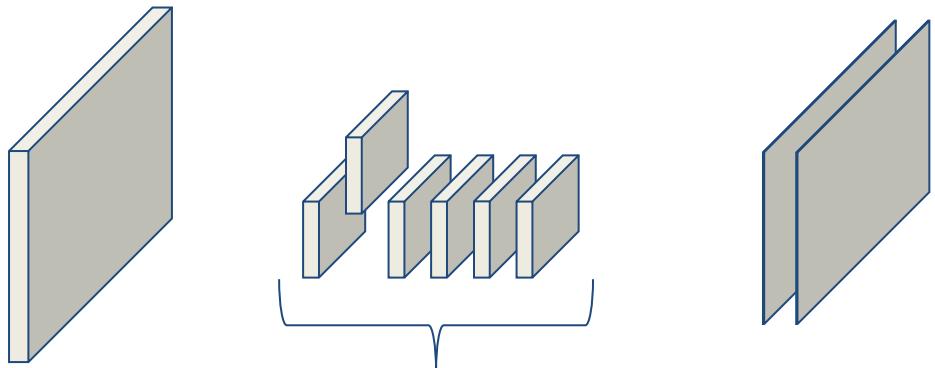


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

Die Faltung der Eingabe mit einem der Faltungskerne ergibt eine Aktivierungskarte. Ohne weitere Behandlung der Ränder ist diese kleiner als die Eingabe.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

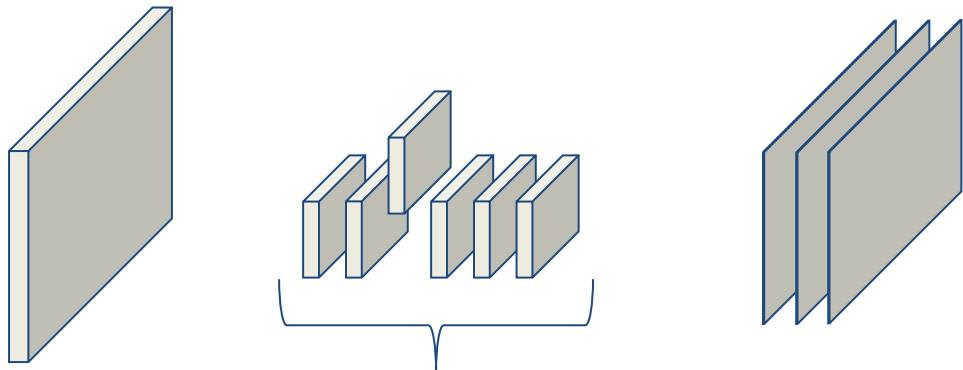


Eingabe	Faltungskerne	Aktivierungskarte
$32 \times 32 \times 3$	Filtergröße: $5 \times 5 \times 3$ Zahl der Filter: 6	$28 \times 28 \times 1$

Jeder Faltungskern erzeugt eine eigene Aktivierungskarte.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

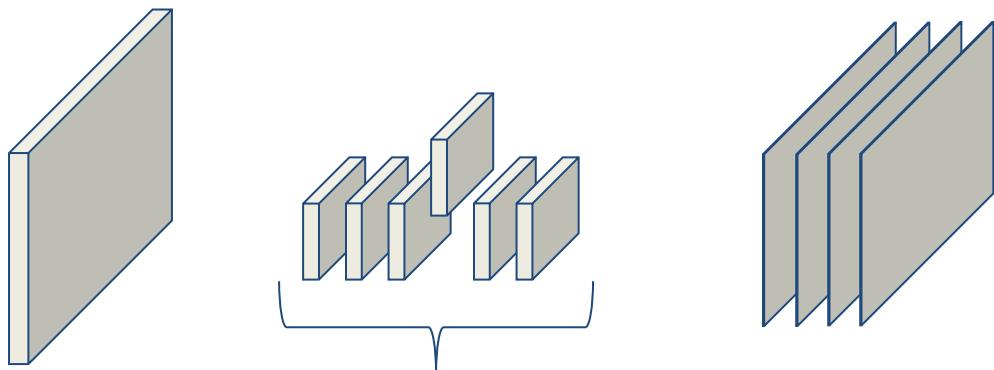


Jeder Faltungskern erzeugt eine eigene Aktivierungskarte.

Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

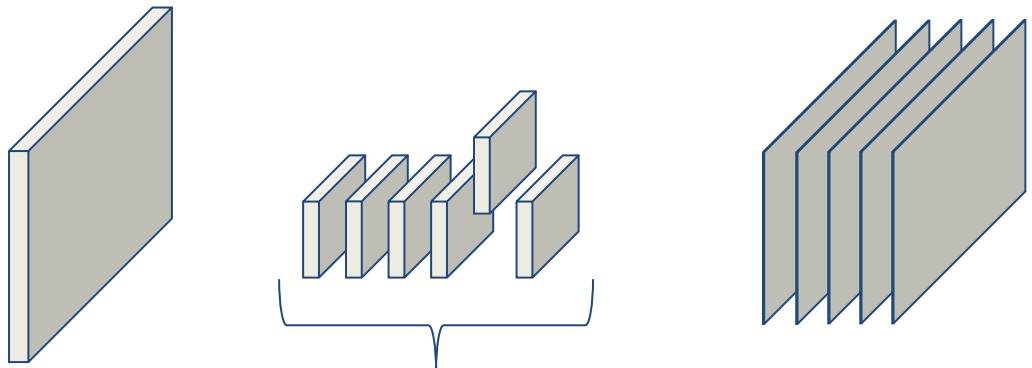


Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

Jeder Faltungskern erzeugt eine eigene Aktivierungskarte.

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

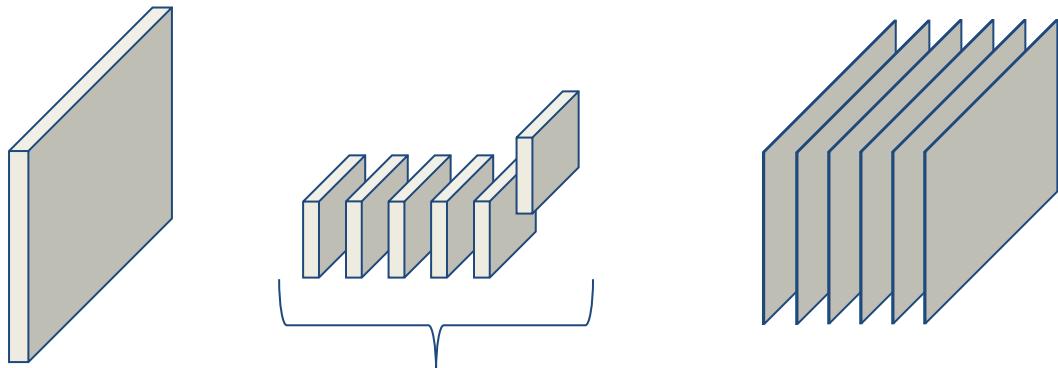


Jeder Faltungskern erzeugt eine eigene Aktivierungskarte.

Eingabe	Faltungskerne	Aktivierungskarte
32x32x3	Filtergröße: 5x5x3 Zahl der Filter: 6	28x28x1

# Faltungsschichten

- Eine Faltungsschicht ist gegeben durch
  - Zahl der Faltungskerne
  - Größe der Faltungskerne
  - Padding
  - Schrittweite

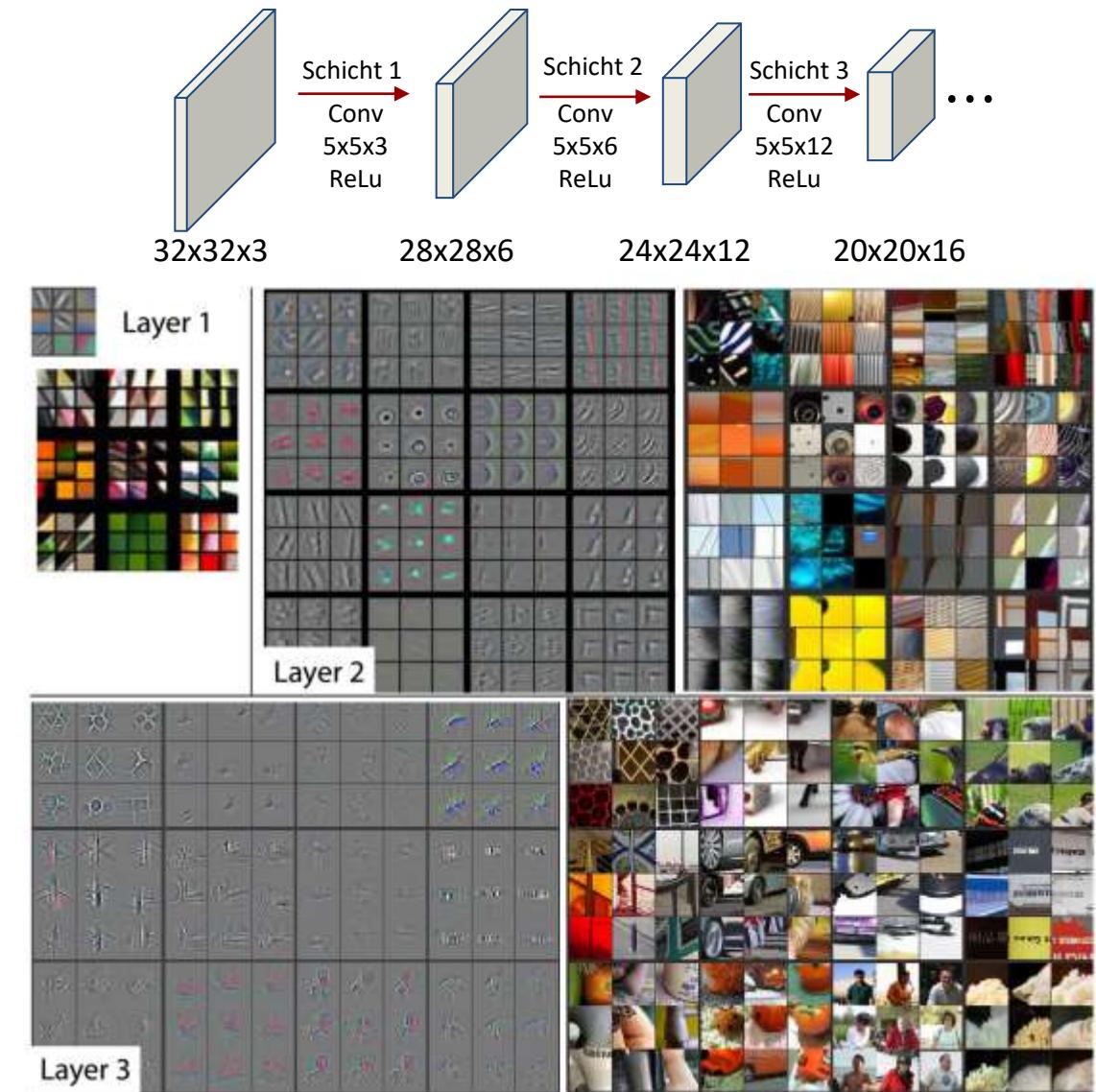


Eingabe	Faltungskerne	Aktivierungskarte
$32 \times 32 \times 3$	Filtergröße: $5 \times 5 \times 3$ Zahl der Filter: 6	$28 \times 28 \times 1$

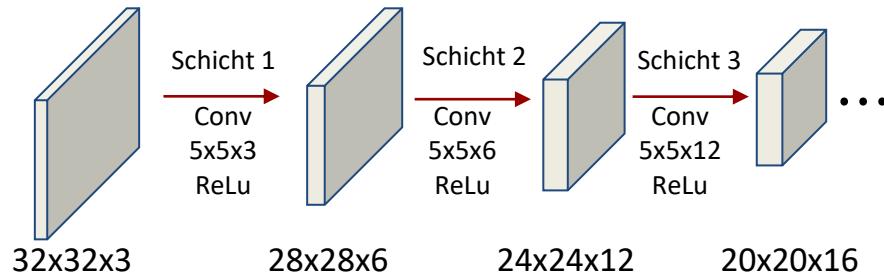
Das aus allen Aktivierungskarten bestehende 3D-Array („Tensor“) wird zur Eingabe der nächsten Schicht.

# Merkmalshierarchie

- Die Faltungskerne sind **Parameter** des Netzwerks
  - Zufällige Initialisierung
  - Optimierung via Backpropagation
- Durch die **hierarchische Anordnung** erkennen tiefere Schichten komplexe Merkmale
  - Das rezeptive Feld tieferer Neurone wird immer größer



# Das rezeptive Feld



Das **rezeptive Feld** eines Neurons ist der Ausschnitt seiner Eingabe, der einen Einfluss auf seine Ausgabe hat.

1<sup>te</sup> Faltungsschicht

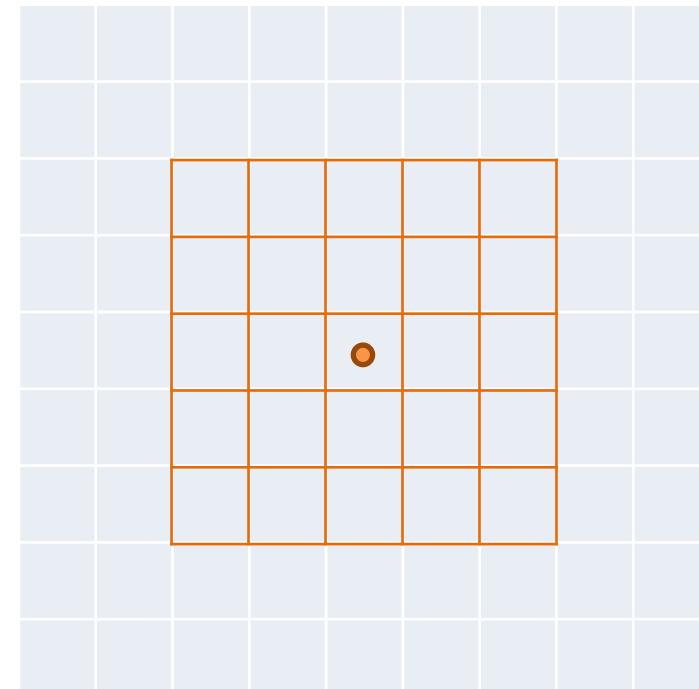
2<sup>te</sup> Faltungsschicht

Das rezeptive Feld der ersten Schicht ist 5x5

Quiz: Wie ist es mit Schicht 2 und Schicht 3?

Antwort: 9x9 bzw. 13x13

- Können wir es schneller vergrößern?
- ...ohne dabei so viele Pixel am Rand zu verlieren?



# Faltung: Verlust des Bildrandes

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

\*

1	-1	0
-1	0	0
1	0	1

3x3

Quiz: Welche Dimensionen hat die Ausgabe dieser Faltung?

Antwort: 5x5

# Padding: Faltung mit Auffüllen der Ränder

0	0	0	0	0	0	0	0	0	0
0	1	0	2	1	1	2	2	0	0
0	1	0	1	0	1	0	1	0	0
0	2	1	2	1	1	1	1	0	0
0	-1	0	1	2	1	2	2	0	0
0	1	2	0	-1	0	1	-1	0	0
0	6	1	7	-1	0	1	0	0	0
0	1	1	1	2	1	0	2	0	0
0	0	0	0	0	0	0	0	0	0

\*

1	-1	0
-1	0	0
1	0	1

3x3

*Quiz:* Welche Dimensionen hat die Ausgabe dieser Faltung?

*Antwort:* 5x5

*Quiz:* Wie ist die Dimensionen, wenn wir den Rand einen Pixel breit mit Nullen auffüllen (padding=1)?

*Antwort:* 7x7

# Faltung mit größeren Schrittweiten

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

**Stride=2**

**Padding=0**

$$\begin{matrix} * & \begin{matrix} 1 & -1 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} & = & \begin{matrix} & & \\ & & \\ & & \\ & & \\ & & \end{matrix} \end{matrix}$$

3x3

Faltungen mit größeren Schrittweiten (engl. stride) vergrößern das rezeptive Feld in den folgenden Schichten

# Faltung mit größeren Schrittweiten

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

**Stride=2**

**Padding=0**

$$\begin{matrix} * & \begin{matrix} 1 & -1 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} & = & \begin{matrix} 4 & & \\ & & \\ & & \end{matrix} \end{matrix}$$

3x3

Faltungen mit größeren Schrittweiten (engl. stride) vergrößern das rezeptive Feld in den folgenden Schichten

# Faltung mit größeren Schrittweiten

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

**Stride=2**

**Padding=0**

$$\begin{matrix} * & \begin{matrix} 1 & -1 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} & = & \begin{matrix} 4 & 3 & \\ & & \end{matrix} \end{matrix}$$

3x3

Faltungen mit größeren Schrittweiten (engl. stride) vergrößern das rezeptive Feld in den folgenden Schichten

# Faltung mit größeren Schrittweiten

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

**Stride=2**

**Padding=0**

$$\begin{matrix} * & \begin{matrix} 1 & -1 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} & = & \begin{matrix} 4 & 3 & 0 \\ & & \\ & & \end{matrix} \end{matrix}$$

3x3

Faltungen mit größeren Schrittweiten (engl. stride) vergrößern das rezeptive Feld in den folgenden Schichten

# Faltung mit größeren Schrittweiten

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

**Stride=2**

**Padding=0**

$$\begin{matrix} * & \begin{matrix} 1 & -1 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} & = & \begin{matrix} 4 & 3 & 0 \\ 3 & & \\ & & \end{matrix} \end{matrix}$$

3x3

Faltungen mit größeren Schrittweiten (engl. stride) vergrößern das rezeptive Feld in den folgenden Schichten

# Faltung mit größeren Schrittweiten

1	0	2	1	1	2	2
1	0	1	0	1	0	1
2	1	2	1	1	1	1
-1	0	1	2	1	2	2
1	2	0	-1	0	1	-1
6	1	7	-1	0	1	0
1	1	1	2	1	0	2

7x7

**Stride=2**

**Padding=0**

$$\begin{matrix} * & \begin{matrix} 1 & -1 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} & = & \begin{matrix} 4 & 3 & 0 \\ 3 & 0 & -2 \\ -5 & -4 & 2 \end{matrix} \end{matrix}$$

3x3

Faltungen mit größeren Schrittweiten (engl. stride) vergrößern das rezeptive Feld in den folgenden Schichten

# Pooling: Aggregation über Nachbarschaften

- **Pooling-Schichten** aggregieren Aktivierungen über eine räumliche Nachbarschaft. Sie sind definiert über
  1. Größe (z.B. 2x2)
  2. Art des Poolings (z.B. Maximum, Mittelwert)
  3. Schrittweite (stride)
- Pooling ist eine beliebte Alternative zum **Downsampling** innerhalb von CNNs
  - Am üblichsten ist Max-Pooling
  - Hierbei benötigt die Backpropagation die Positionen der Maxima („switches“)

1	0	2	1	1	2
1	0	1	0	1	0
2	1	2	1	1	1
-1	0	1	2	1	2
1	2	0	-1	0	1
6	1	7	-1	0	1

Max pool  
2x2  
Stride=2

1	2	2
2	2	2
6	7	1

# Zusammenfassung

- **Convolutional Neural Networks (CNNs)** nutzen Faltungsschichten, um auf Gittern definierte hochdimensionale Eingabedaten (insb. Bilder) effizienter zu verarbeiten
  - Reduzierte Zahl der **Parameter**, Äquivarianz bzgl. **Verschiebungen**
  - **Hierarchische Anordnung** der Faltungsschichten repräsentiert zu Beginn einfache, lokale Merkmale, erkennt tiefer im Netz mit größerem rezeptiven Feld komplexe Konzepte
- Hinreichend große **rezeptive Felder** mit praktikabler Zahl von Parametern werden ermöglicht durch
  - Faltungen mit größerer Schrittweite (*engl.* strided convolution)
  - Pooling-Schichten

## **6b.3 Bildklassifikation mit CNNs**

# Überblick

- Grundlagen der CNN-basierten Bildklassifikation sind seit 1989 (erste Version des LeNet) bekannt
- Fest etabliert hat sich diese Idee 2012 und wurde seither rapide weiterentwickelt
  - Wesentliche Faktoren für den Erfolg waren:
    - Ideen zum erfolgreichen Training sehr **tiefer Netze**
    - Verfügbarkeit hinreichender und kostengünstiger **Rechenkapazität** durch GPUs (hochgradig parallele Koprozessoren, urspr. zur Bildsynthese)
    - Verfügbarkeit sehr umfangreicher **Trainingsdatensätze**

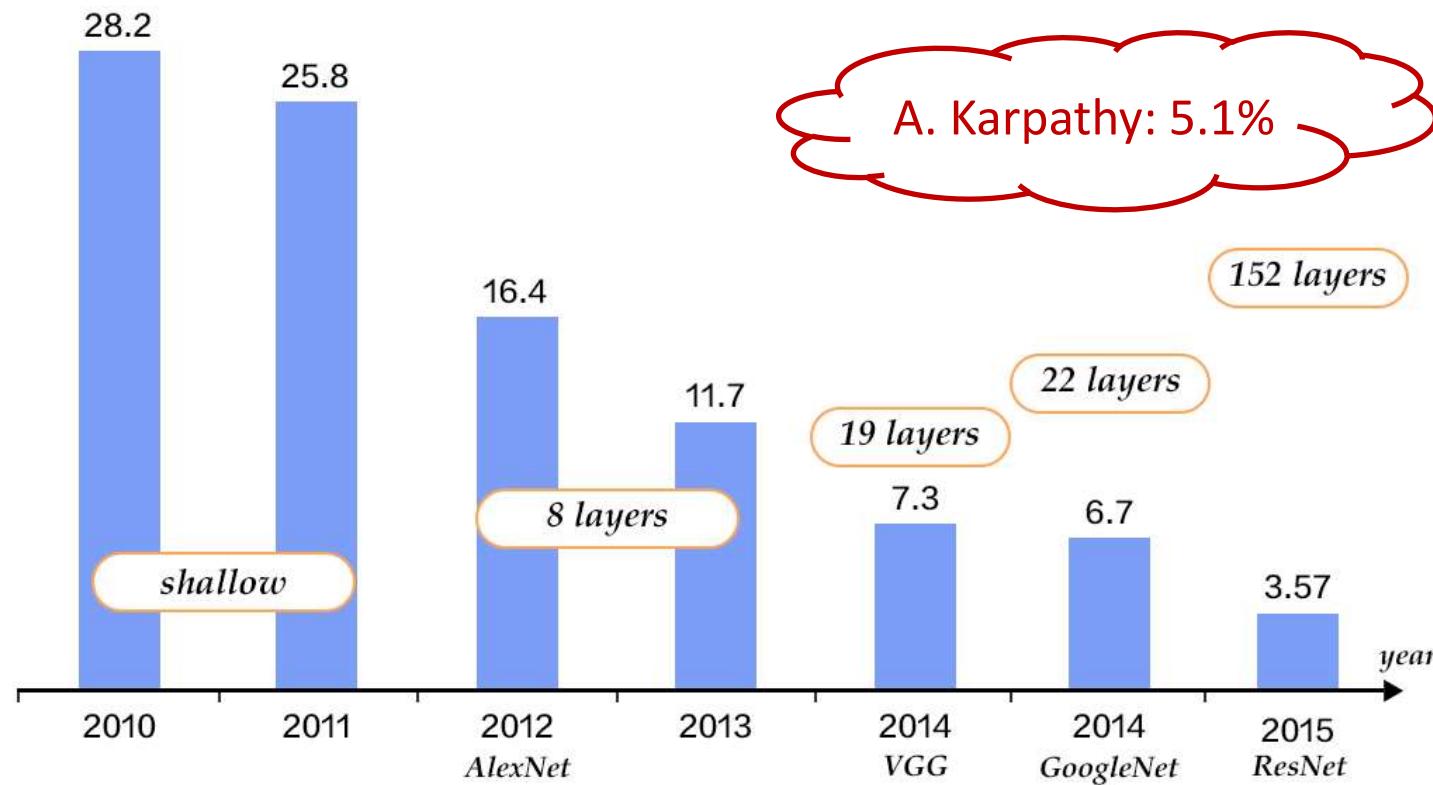
# ImageNet-Wettbewerb

- **ImageNet** ist eine Datenbank von mehr als 14 Mio Bildern, die jeweils einer von 20.000 Kategorien zugeordnet wurden
- Mit einer Teilmenge von ca. 1,5 Mio Bildern aus 1000 Klassen wurde 2010-2017 die **ImageNet Large Scale Visual Recognition Challenge (ILSVRC)** ausgerichtet
  - Unterschiede zum Teil sehr fein, allein 120 Klassen sind verschiedene Hunderassen
  - Top-5-Fehler wertet es als Erfolg, wenn die korrekte Klasse unter den 5 als am wahrscheinlichsten vorhergesagten ist



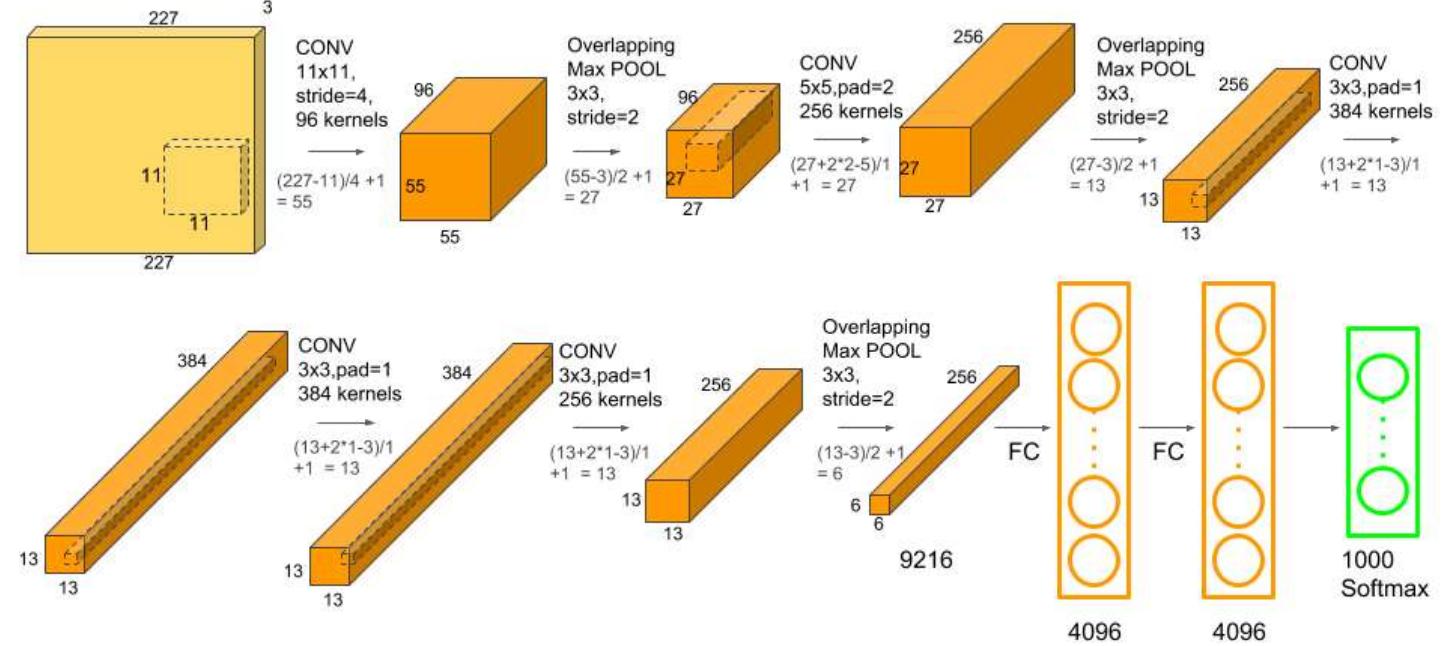
# Thema dieses Kapitels

Wir betrachten im Folgenden die CNN-Architekturen, die zu wesentlichen Fortschritten auf der ILSVRC geführt haben



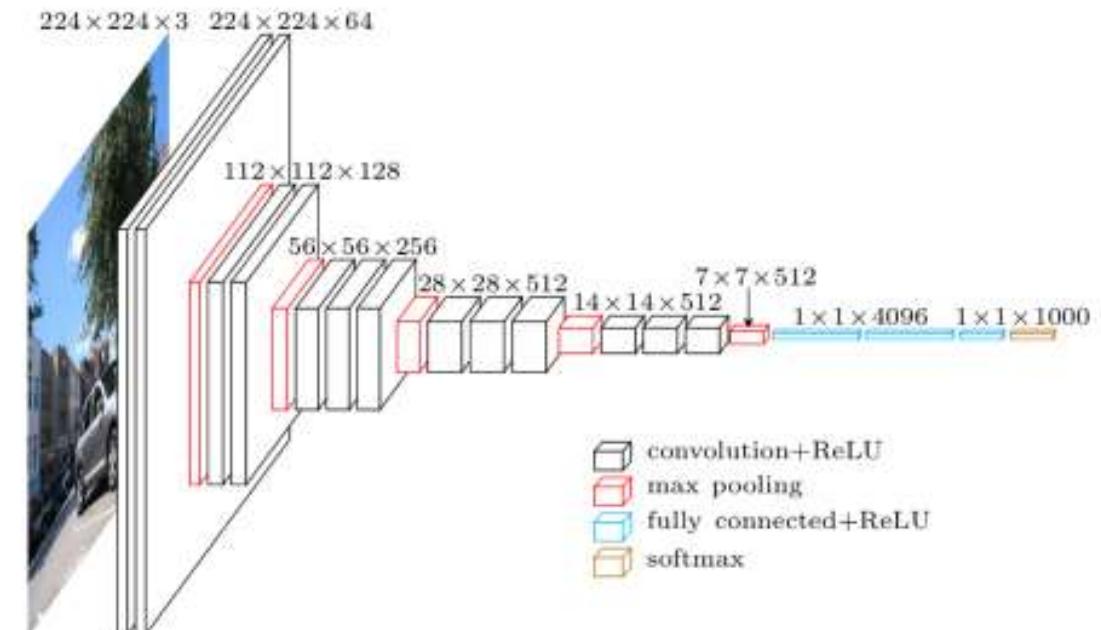
# AlexNet

- **AlexNet** [Krizhevsky et al.] gewann 2012 mit 16,4% Top-5-Fehler als erste CNN-Architektur die ILSVRC
  - Training auf zwei GPUs dauerte 5-6 Tage
- Grundidee und Bausteine ähnlich wie im LeNet. Neu waren:
  - Mehr Schichten
  - Viel mehr Parameter
  - Direkt hintereinander ausgeführte Faltungen
  - ReLU-Aktivierungen
  - Augmentierung
  - Dropout



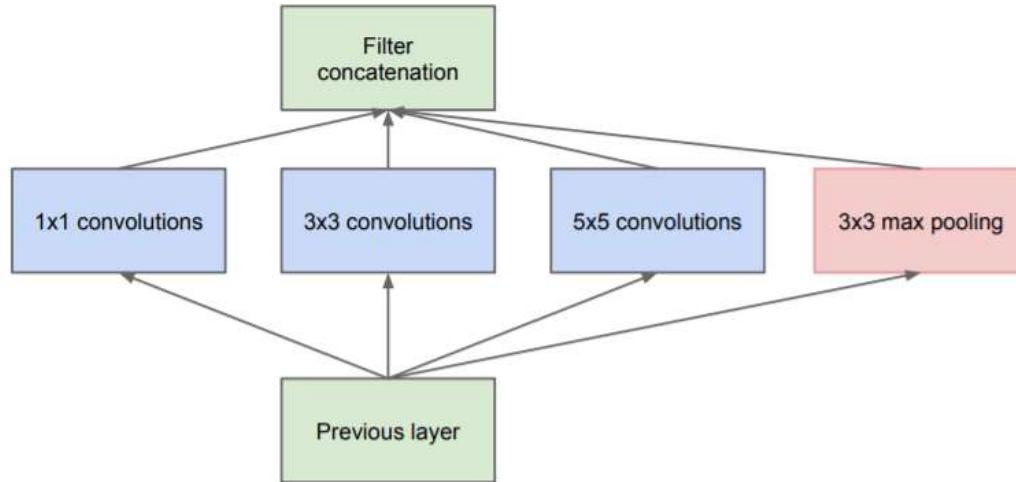
# VGG

- **VGG** [Simonyan/Zisserman] gewann bei der ILSVRC 2014 mit 7,3% Top-5-Fehler den zweiten Platz
  - Training auf 4 GPUs dauerte 2-3 Wochen
- **Hauptidee:** Tiefer ist besser!
  - Mehr, dafür kleinere Kerne (3x3)
  - Einheitlichere Architektur
  - Empfehlung nach Experimenten mit verschiedenen Tiefen:  
16-19 Schichten
    - 138/144 Mio Parameter statt 60 Mio im AlexNet

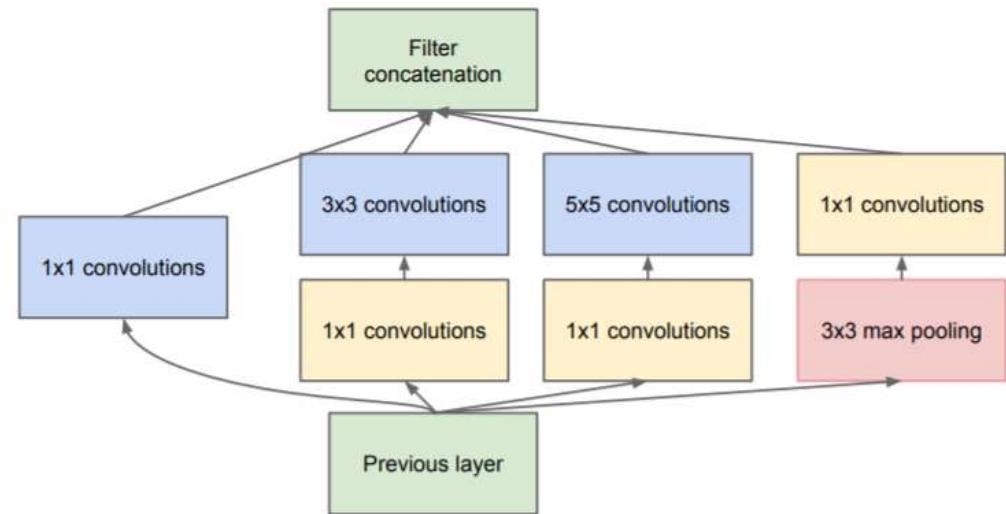


# Inception-Modul

- Das Inception-Modul ermöglicht die Konstruktion tiefer Netze mit relativ wenig Parametern
  - Erste Idee:** Kombination von Faltungskernen verschiedener Größe (Mehr-Skalen-Repräsentation) sowie max-Pooling mit Schrittweite 1
  - Zweite Idee:** Dimensionsreduzierung mit  $1 \times 1$ -Faltungen reduziert die Zahl der Parameter in den Faltungskernen



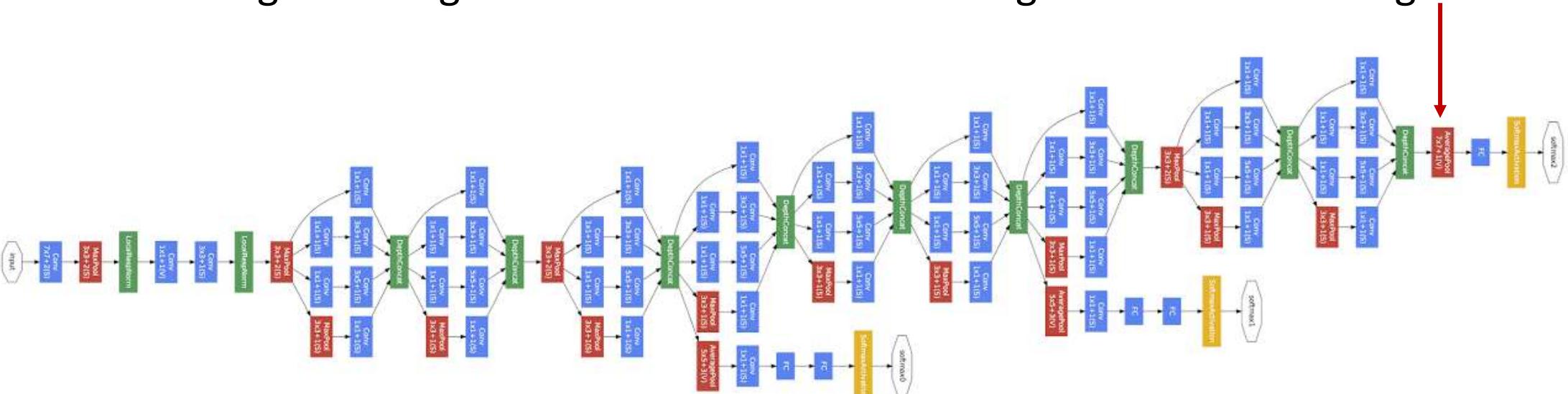
(a) Inception module, naïve version



(b) Inception module with dimension reductions

# GoogLeNet

- **GoogLeNet** [Szegedy et al.] gewann mit 6,7% Top-5-Fehler die ILSVRC 2014
  - Enthält 9 Inception-Module, insgesamt 22 Schichten
  - Hat dennoch viel weniger Parameter als AlexNet (5 Mio statt 60 Mio)
    - Wichtiger Beitrag hierzu: 7x7 Mittelwert-Pooling nach letzter Faltungsschicht



Faltung / Pooling / Softmax / Sonstige

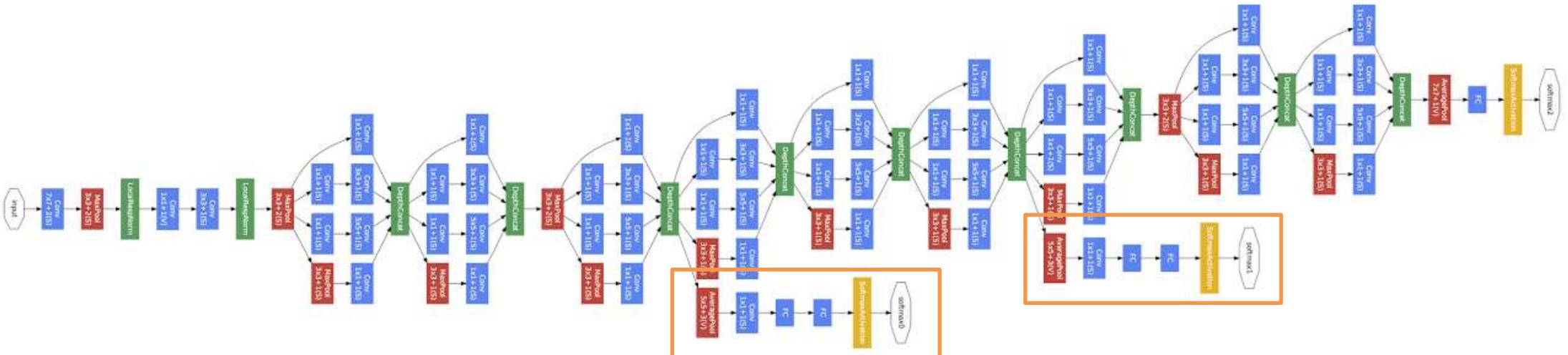
# GoogLeNet: Der Teufel im Detail (Teil 1)

- Den knappen Vorsprung hatte GoogLeNet u.a. einer sorgfältigen **Optimierung der Hyperparameter** zu verdanken
  - Betrifft u.a. genaue Zahl der Aktivierungskarten jeder Schicht

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

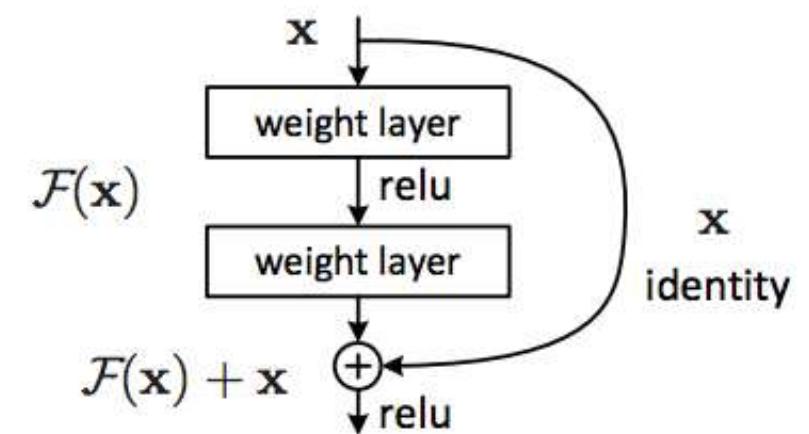
# GoogLeNet: Der Teufel im Detail (Teil 2)

- Der genaue **Trainingsprozess** ist wichtig um gute Ergebnisse zu erhalten, wird in der Publikation aber nicht beschrieben
  - Zitat: "Es ist schwer eine klare Richtschnur anzugeben, wie man diese Netzwerke am effektivsten trainiert."
  - Keine Angaben zu Hardware oder Dauer des Trainings
  - Um frühe Schichten zu trainieren kamen **zusätzliche Klassifikatoren** zum Einsatz, die nach dem Training wieder entfernt wurden



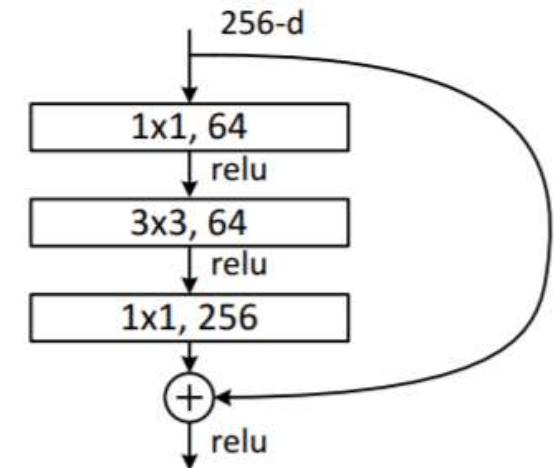
# ResNet: Grundidee

- [He et al. 2015] beschreiben ein **Degradierungs-Problem**:
  - Hinzufügen weiterer Schichten erhöht ab einem bestimmten Punkt den Trainingsfehler
  - Einzige Erklärung: Probleme beim Training solcher Architekturen
- **Grundidee**: Schichten sollten statt einer Funktion  $f(\mathbf{x})$  ein additives Residual  $f_{\text{res}}(\mathbf{x})$  ausgeben, so dass  $f(\mathbf{x}) = f_{\text{res}}(\mathbf{x}) + \mathbf{x}$ 
  - *Begründung*: Identität  $f(\mathbf{x}) = \mathbf{x}$  ist eine nützlichere „Grundeinstellung“ als die Nullfunktion  $f(\mathbf{x}) = 0$
  - *Implementierung*: Shortcut-Verbindung, erfordert keine weiteren Parameter



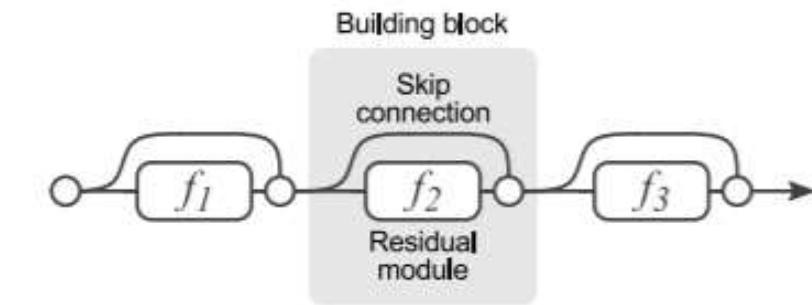
# ResNet: Flaschenhals-Architektur

- Eine **Flaschenhals-Architektur** mit  $1 \times 1$ -Faltungen ermöglicht die Konstruktion tieferer Netze
  - Ähnliche Idee wie bei Inception
- **ResNet-152** [He et al.] gewann mit 3,6% Top-5-Fehler die ILSVRC 2015
  - Besteht aus vielen dieser Blöcke
  - Ergibt insgesamt 152 Schichten
- Training: 2-3 Wochen auf 8 GPUs
  - Einzelner Vorwärts-Durchlauf schneller als bei VGG
    - 11,3 Mrd FLOPs vs. 15,3/19,6 Mrd bei VGG-16/19
    - Viele der Schichten im ResNet-152 sind günstige  $1 \times 1$ -Faltungen

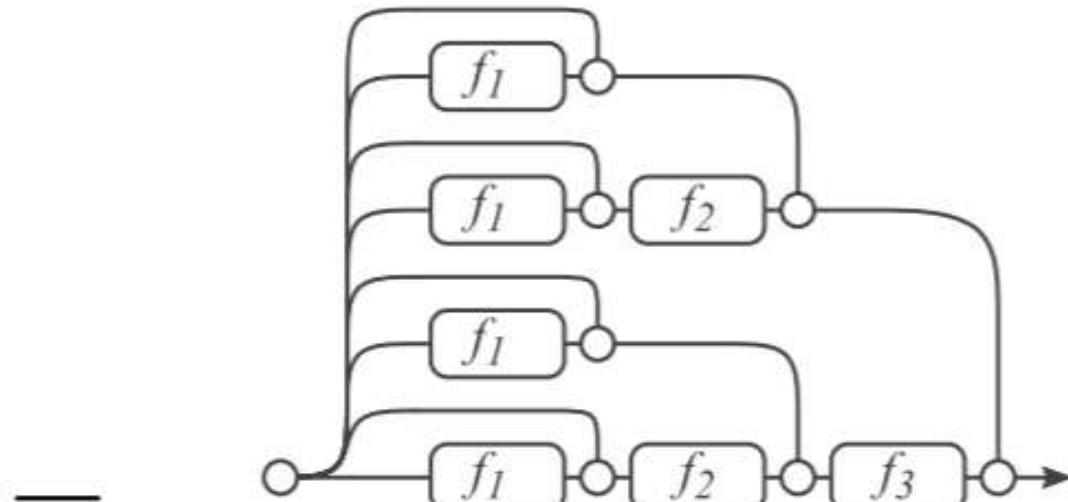


# ResNet: Sehr tief oder ein Ensemble?

- [Veit et al. 2016] schlagen vor, dass man sich ResNets besser als **Ensembles flacherer Netze** (mit 10-34 Schichten) vorstellen sollte denn als sehr tiefe Netze



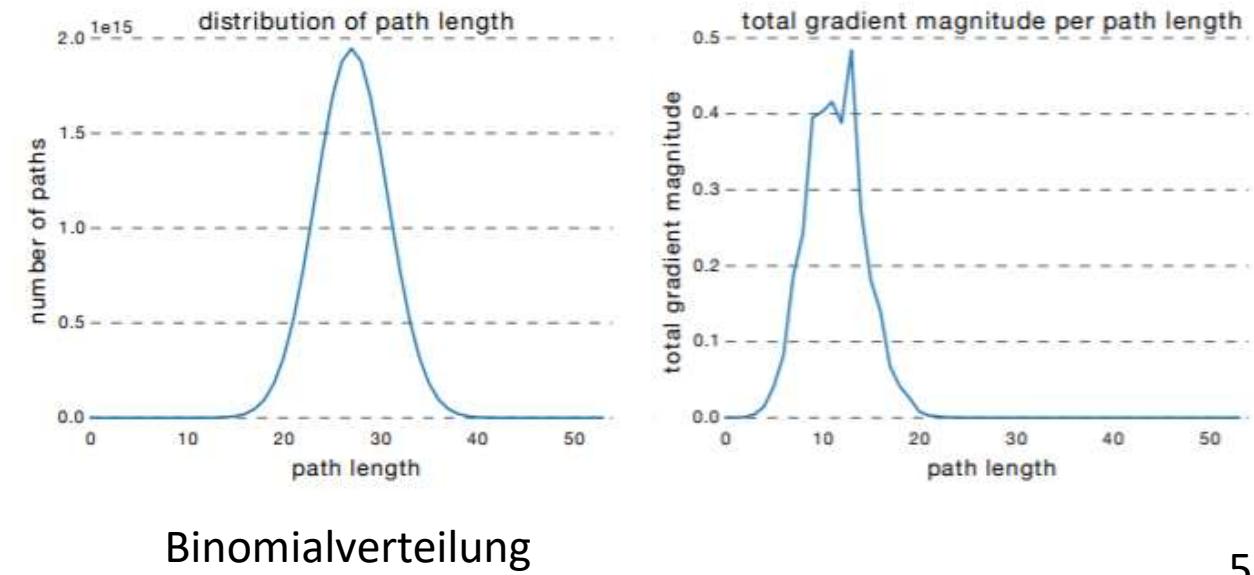
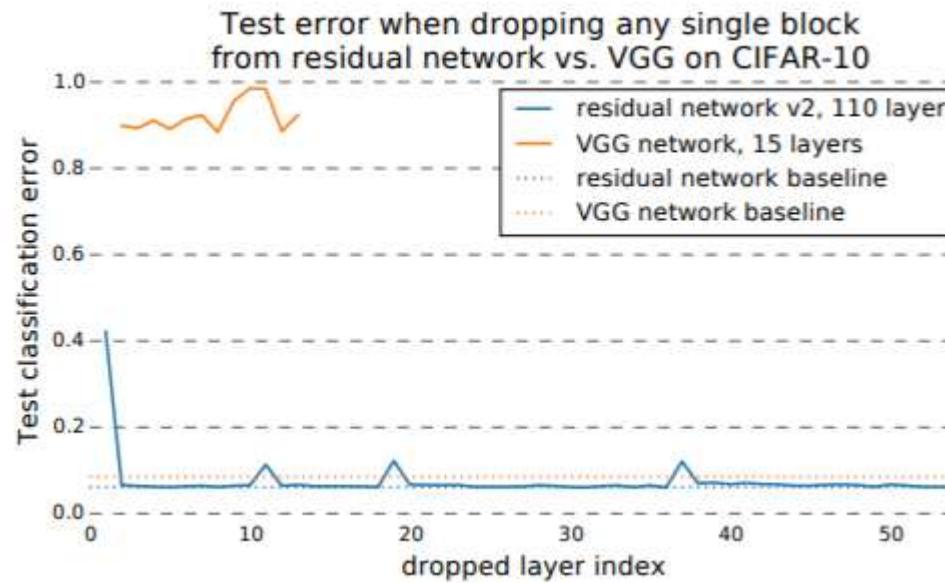
(a) Conventional 3-block residual network



(b) Unraveled view of (a)

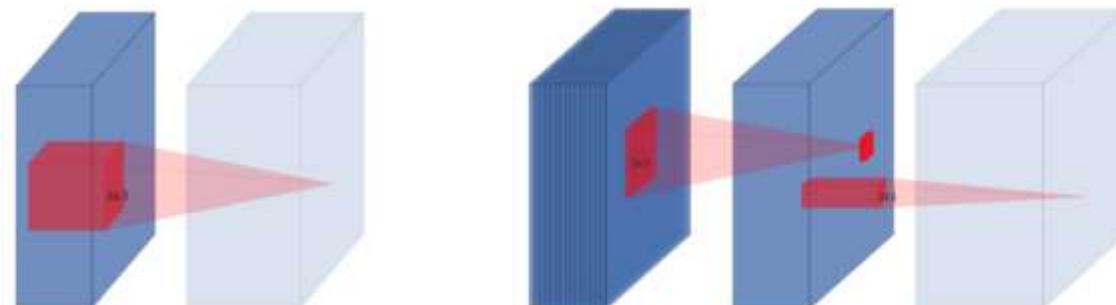
# ResNet: Gründe für die Interpretation als Ensemble

- **Läsionsstudie:** Entfernen einzelner ResNet-Module hat kaum einen Effekt
  - Ausnahme: Downsampling
- Untersuchung von **Pfadlängen:**
  - Die meisten Pfade sind etwa halb so lang wie die nominelle Tiefe
  - Gradienten entlang kurzer Pfade sind stärker

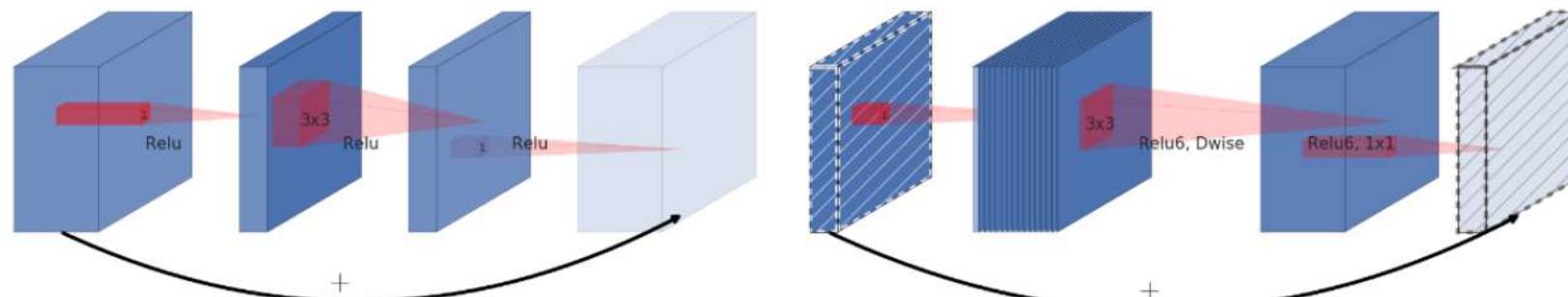


# EfficientNet: Grundarchitektur

- **EfficientNet** [Tan et al. 2019] optimiert die Effizienz
  - Maximierung der Klassifikationsrate *bei festem Rechenbudget*
- Grundarchitektur nutzt Ideen des **MobileNet** [Sandler et al. 2018]
  - **In Tiefenrichtung separierte Faltungen** („depthwise convolution“) sparen Parameter und Rechenoperationen

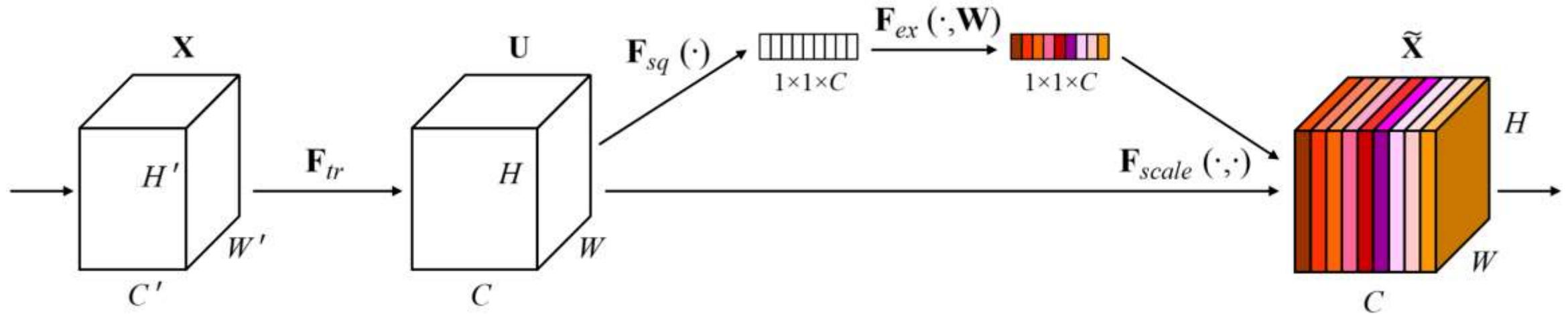


- **Invertierte Residual-Blöcke** kommen mit weniger Additionen aus



# EfficientNet: Squeeze and Excitation

- EfficientNet nutzt außerdem den **squeeze-and-excitation**-Mechanismus [Hu et al. 2018], um wichtige Aktivierungen zu erkennen und höher zu gewichten



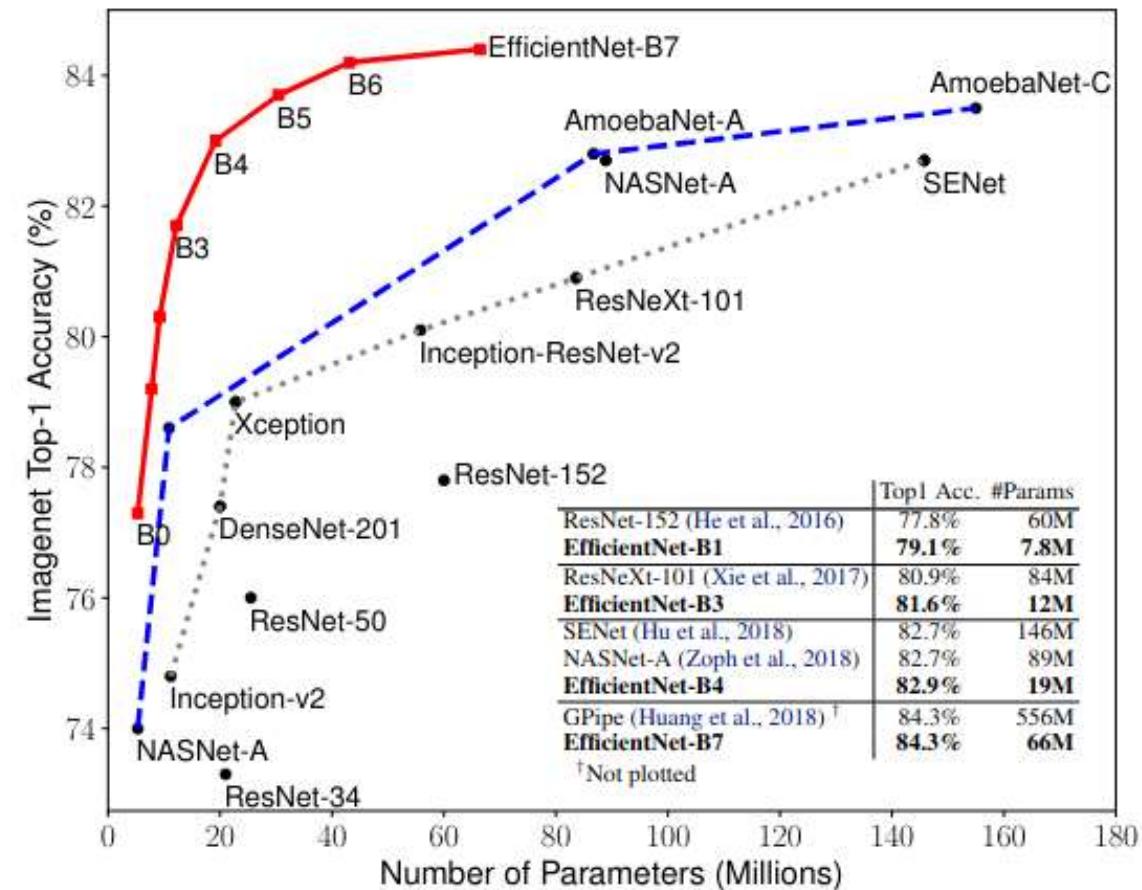
- „**Squeeze**“ durch globales Mittelwert-Pooling über Höhe und Breite
- „**Excitation**“ durch  $\sigma(\mathbf{W}_2 \text{ReLU}(\mathbf{W}_1 \mathbf{z}))$ 
  - $\mathbf{W}_1$  reduziert  $C$  Kanäle um den Faktor  $r$ ,  $\mathbf{W}_2$  stellt Dimension  $C$  wieder her

# EfficientNet: Gemeinsame Skalierung

- Die wesentliche neue Idee von EfficientNet ist es, die Tiefe  $d$  und Breite  $w$  der Grundarchitektur sowie die Auflösung  $r$  des Eingabebilds mit einem **gemeinsamen Exponenten  $\phi$**  zu skalieren:

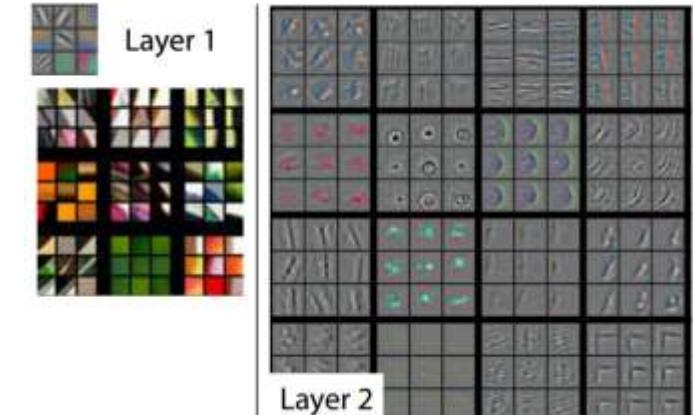
$$d = \alpha^\phi, w = \beta^\phi, r = \gamma^\phi$$

–  $\alpha \geq 1, \beta \geq 1, \gamma \geq 1$  so gewählt, dass  $\alpha \times \beta^2 \times \gamma^2 \approx 2$



# Transferlernen: Grundidee

- Zur Klassifikation medizinischer Bilder nutzt man meist etablierte CNN-Architekturen
  - Diese wurden aufwändig optimiert und sind schwer zu schlagen
  - Wenn relativ wenige Bilder zum Training zur Verfügung stehen, nutzt man häufig auf ImageNet vortrainierte Gewichte
    - *Idee:* Viele Merkmale, insbesondere in den ersten Schichten, sollten hinreichend generell sein um einen Transfer auf andere Aufgaben zu ermöglichen
    - Bibliotheken (z.B. torchvision) stellen fertig implementierte und vortrainierte „Modell-Zoos“ zur Verfügung



# Transferlernen: Umsetzung

Zwei übliche Strategien für Transferlernen sind

## 1. Verwendung des CNNs zur **Merkalsextraktion**

- Entfernen des finalen Klassifikators
  - i.d.R. vollständig verbundene Schichten am Ende
  - Bei sehr unterschiedlichen Aufgaben z.T. auch spätere Faltungsschichten
- Ersetzen durch beliebiges überwachtes Lernverfahren

## 2. Verfeinerung (*engl. fine tuning*) des CNN

- Wenn die Klassifikation wieder durch ein neuronales Netz erfolgt ist es möglich auch die Faltungsschichten weiter zu trainieren
- Viele Varianten sind denkbar, z.B.
  - Einfrieren des Merkalsextraktors, bis der Klassifikator trainiert wurde
  - Einfrieren früherer Schichten, die vermutlich generelle Merkmale enthalten
  - Optimierung früherer Schichten mit geringeren Lernraten

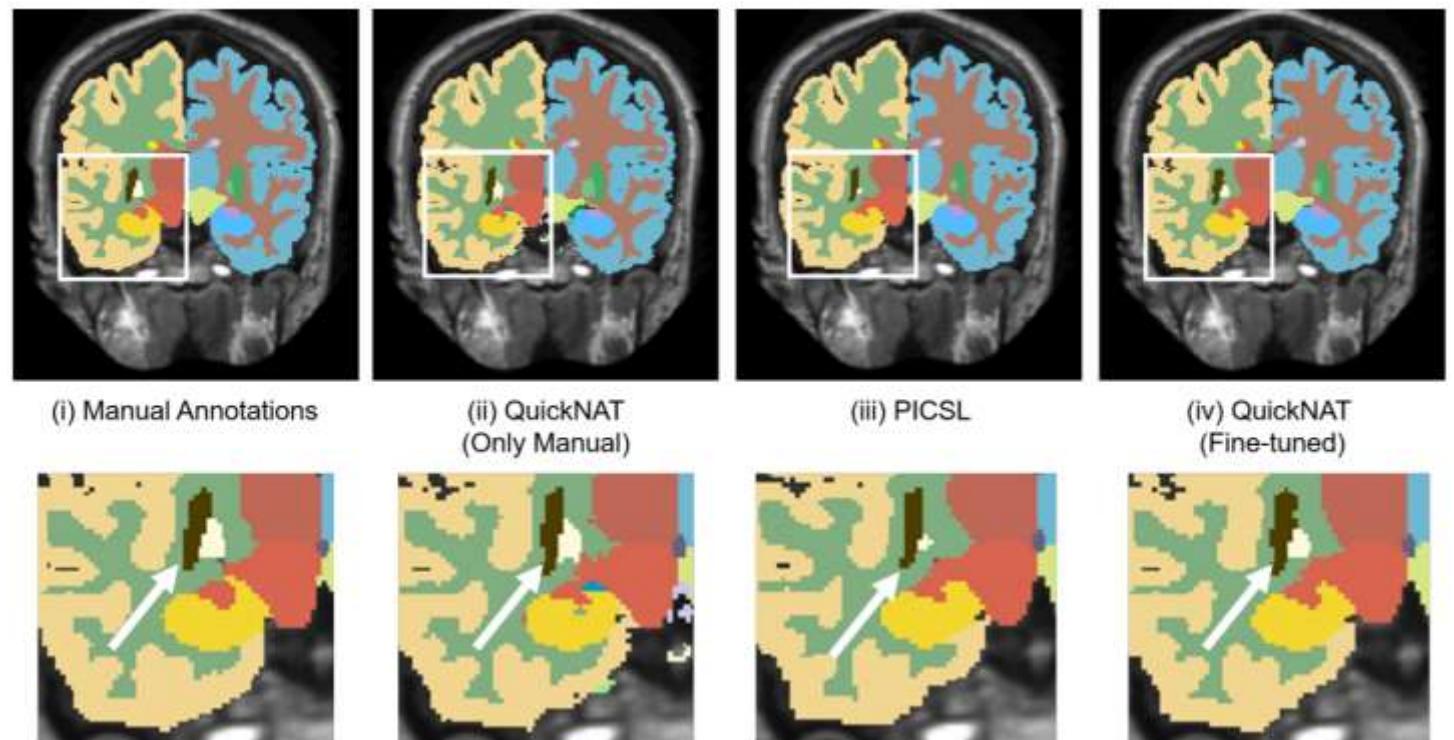
# Zusammenfassung

- **CNNs zur Bildklassifikation** wurden mit der Zeit immer tiefer:
  - **AlexNet** führte zur Etablierung von CNNs zur Bildklassifikation
  - **VGG-16/19** arbeiten einheitlich mit 3x3-Faltungen
  - Die Inception-Module in **GoogLeNet** sparen massiv Parameter ein
  - Shortcut-Verbindungen im **ResNet** erlauben viel tiefere Netzwerke
    - Alternative Interpretation als Ensemble flacherer Netze
  - **EfficientNet** ist zusätzlich im Hinblick auf Recheneffizienz optimiert
- In der Praxis nutzt man meist eine Variante dieser etablierten Architekturen, häufig in Verbindung mit **Transferlernen**

## **6b.4 Bildsegmentierung mit CNNs**

# Motivation: Segmentierung von Hirnstrukturen

- *Beispiel:* Die Segmentierung anatomischer Strukturen in Hirn-MRT-Scans ist ein intensiv beforschtes Problem
  - Klassische Verfahren haben einen hohen **Rechenaufwand**
    - PICSL ca. 30h pro Hirn, Freesurfer ca. 4h
  - **CNN-basierte Verfahren** benötigen weniger als eine Minute, bei höherer Genauigkeit



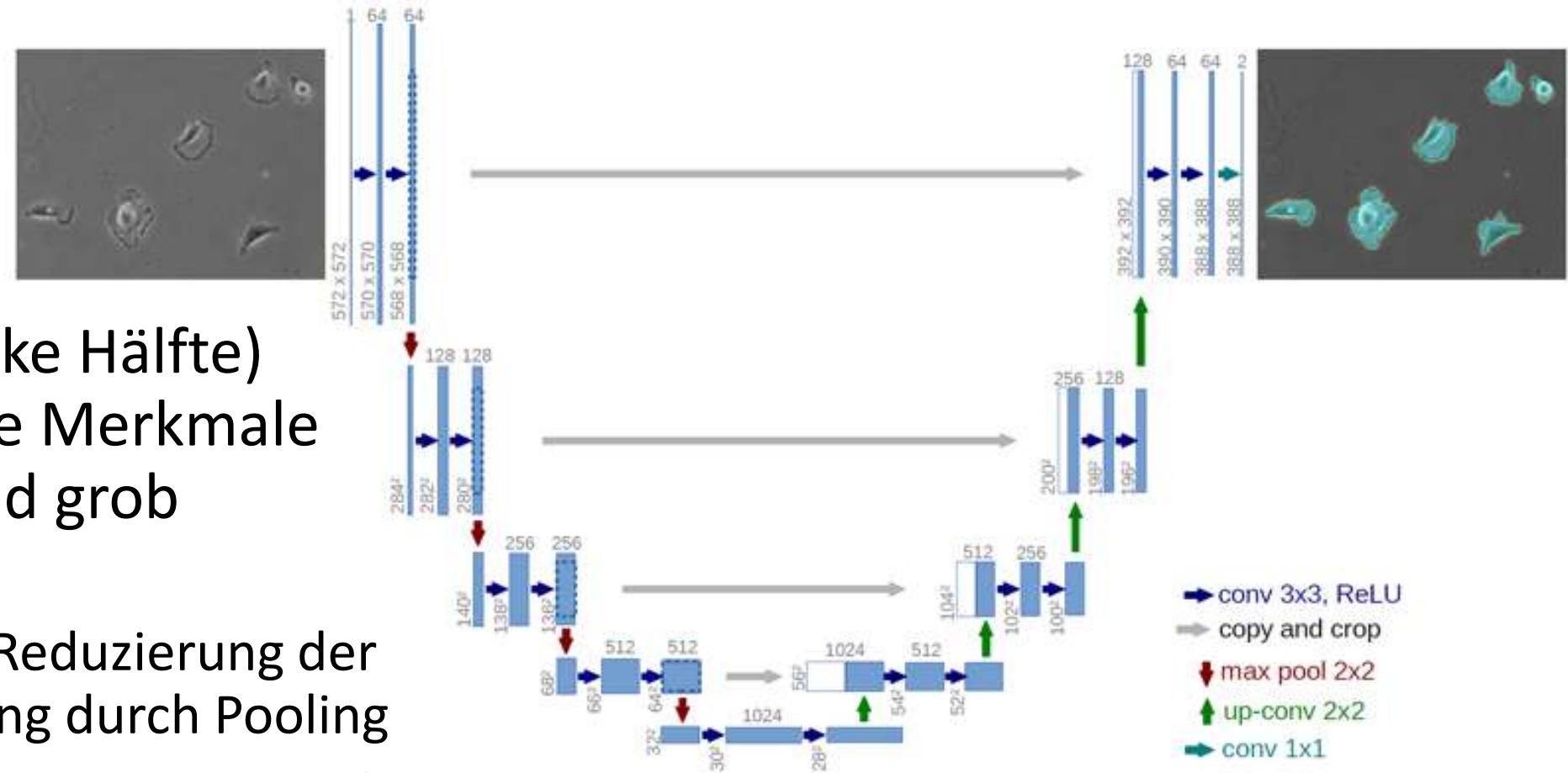
Bildquelle: Abhijit Guha Roy et al., „QuickNAT: A fully convolutional network for quick and accurate segmentation of neuroanatomy“ Neurolimage 2019

# Bildsegmentierung mit CNNs

- **Bildsegmentierung** erfordert Klassifikation jeden Pixels/Voxels
- Erste Ansätze nutzten ein **gleitendes Fenster** zur Klassifikation des zentralen Pixels
  - Rechenaufwändig, hohe Redundanz zwischen den Durchläufen
- **Vollständig faltungsbasierte** (*engl. fully convolutional*) **Ansätze** klassifizieren einen größeren Bildausschnitt in einem einzigen Durchlauf
  - Die in der medizinischen Bildanalyse mit Abstand beliebteste Architektur ist das **U-Net** [Ronneberger et al. 2015]
    - Bis Januar 2025: ≈99.500 Zitate

# Grundstruktur des U-Net

- **Kodierer** (linke Hälfte)  
soll relevante Merkmale erkennen und grob lokalisieren
  - Drastische Reduzierung der Bildauflösung durch Pooling
- **Dekodierer** (rechte Hälfte)  
soll die Klassen erkennen und pixelgenau abgrenzen
  - Transponierte Faltung stellt die Bildauflösung wieder her



# Transponierte Faltung

- Die **transponierte Faltung** (engl. auch „up convolution“) führt ein Upsampling mit lernbaren Gewichten durch
  - Pixel des Eingabe-Bilds skalieren den Filter
  - Ergebnisse werden mit einer vorgegebenen Schrittweite akkumuliert
  - Beispiel:* 2x2-Filter mit Schrittweite 1
    - Hinweis:* Der U-Net-Dekoder nutzt Schrittweite 2

$$\begin{array}{|c|c|} \hline 5 & -2 \\ \hline 3 & 1 \\ \hline \end{array} *^T \begin{array}{|c|c|} \hline 1 & 0 \\ \hline -1 & 2 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline \end{array}$$

Eingabe                      Up-conv-Filter                      Ausgabe

# Transponierte Faltung

- Die **transponierte Faltung** (engl. auch „up convolution“) führt ein Upsampling mit lernbaren Gewichten durch
  - Pixel des Eingabe-Bilds skalieren den Filter
  - Ergebnisse werden mit einer vorgegebenen Schrittweite akkumuliert
  - Beispiel:* 2x2-Filter mit Schrittweite 1
    - Hinweis:* Der U-Net-Dekoder nutzt Schrittweite 2

$$\begin{array}{|c|c|} \hline 5 & -2 \\ \hline 3 & 1 \\ \hline \end{array} *^T \begin{array}{|c|c|} \hline 1 & 0 \\ \hline -1 & 2 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 5 & 0 & \\ \hline -5 & 10 & \\ \hline & & \\ \hline \end{array}$$

Eingabe                      Up-conv-Filter                      Ausgabe

# Transponierte Faltung

- Die **transponierte Faltung** (engl. auch „up convolution“) führt ein Upsampling mit lernbaren Gewichten durch
  - Pixel des Eingabe-Bilds skalieren den Filter
  - Ergebnisse werden mit einer vorgegebenen Schrittweite akkumuliert
  - Beispiel:* 2x2-Filter mit Schrittweite 1
    - Hinweis:* Der U-Net-Dekoder nutzt Schrittweite 2

$$\begin{array}{|c|c|} \hline 5 & -2 \\ \hline 3 & 1 \\ \hline \end{array} *^T \begin{array}{|c|c|} \hline 1 & 0 \\ \hline -1 & 2 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 5 & -2 & 0 \\ \hline -5 & 12 & -4 \\ \hline \end{array}$$

Eingabe                      Up-conv-Filter                      Ausgabe

# Transponierte Faltung

- Die **transponierte Faltung** (engl. auch „up convolution“) führt ein Upsampling mit lernbaren Gewichten durch
  - Pixel des Eingabe-Bilds skalieren den Filter
  - Ergebnisse werden mit einer vorgegebenen Schrittweite akkumuliert
  - Beispiel:* 2x2-Filter mit Schrittweite 1
    - Hinweis:* Der U-Net-Dekoder nutzt Schrittweite 2

$$\begin{array}{|c|c|} \hline 5 & -2 \\ \hline 3 & 1 \\ \hline \end{array} *^T \begin{array}{|c|c|} \hline 1 & 0 \\ \hline -1 & 2 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 5 & -2 & 0 \\ \hline -2 & 12 & -4 \\ \hline -3 & 6 & \\ \hline \end{array}$$

Eingabe                      Up-conv-Filter                      Ausgabe

# Transponierte Faltung

- Die **transponierte Faltung** (engl. auch „up convolution“) führt ein Upsampling mit lernbaren Gewichten durch
  - Pixel des Eingabe-Bilds skalieren den Filter
  - Ergebnisse werden mit einer vorgegebenen Schrittweite akkumuliert
  - Beispiel:* 2x2-Filter mit Schrittweite 1
    - Hinweis:* Der U-Net-Dekoder nutzt Schrittweite 2

5	-2
3	1

Eingabe

1	0
-1	2

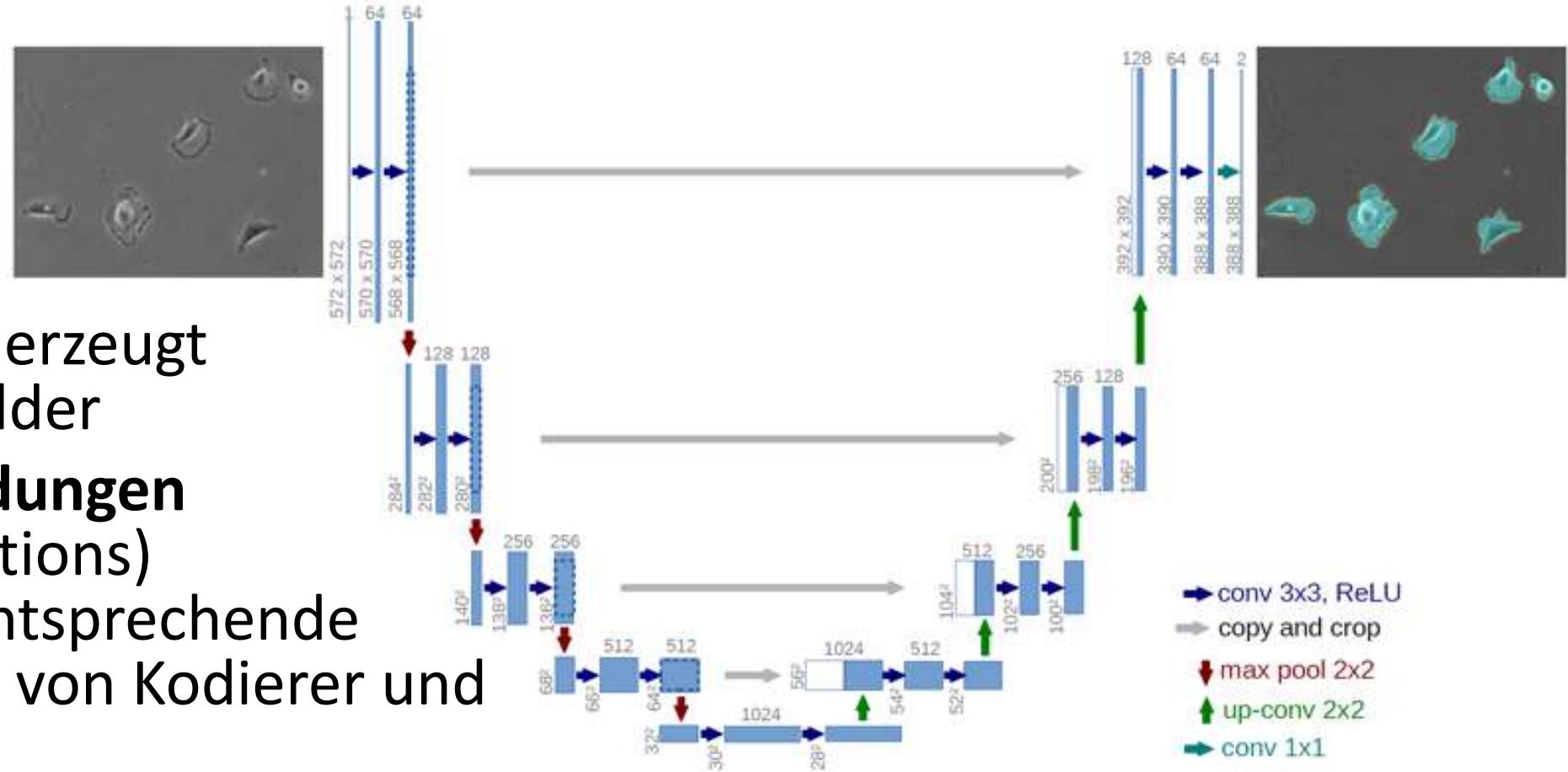
Up-conv-Filter

5	-2	0
-2	13	-4
-3	5	2

Ausgabe

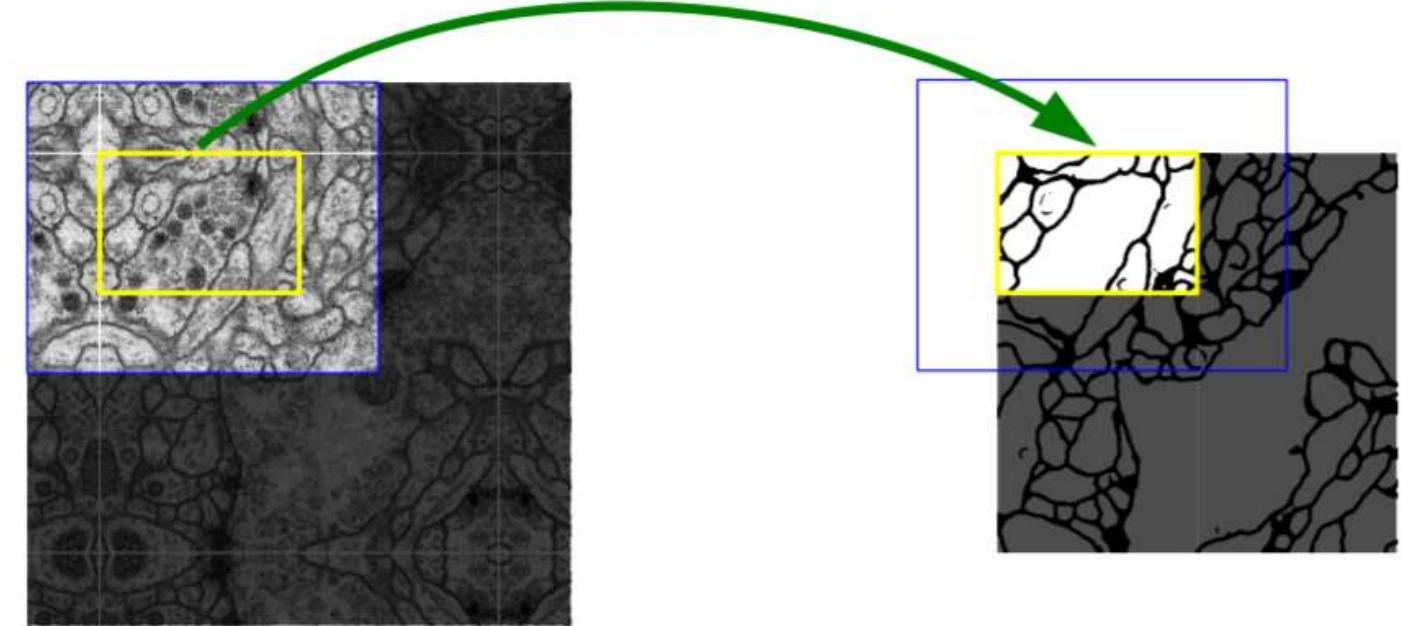
# Skip-Verbindungen

- **Upsampling** erzeugt unscharfe Bilder
- **Skip-Verbindungen** (skip connections) verbinden entsprechende Auflösungen von Kodierer und Dekodierer
  - Helfen bei der Dekodierung scharfer Segmentierungsmasken
  - Schneiden den bei „übersprungenen“ Faltungen verlorenen Rand ab



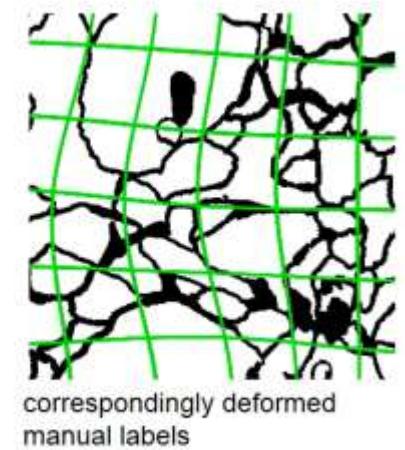
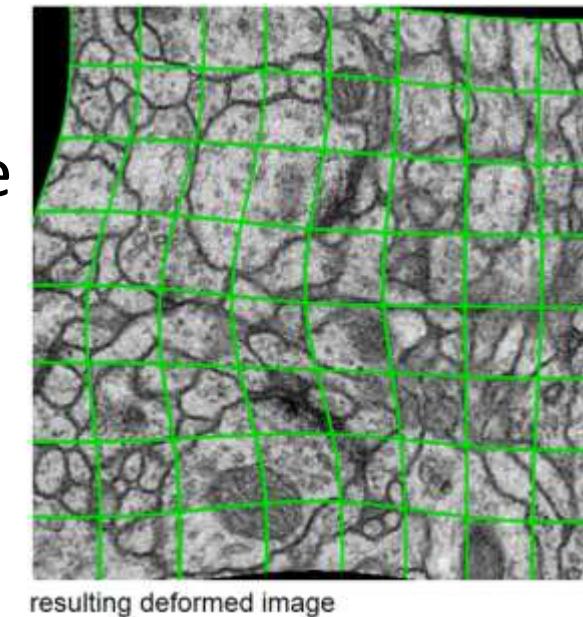
# Verarbeitung überlappender Bildausschnitte

- Das U-Net nutzt nur den gültigen Teil der Faltungen
  - Kein Padding innerhalb des Netzes
  - Padding des Eingabebilds durch Spiegelung
- Größere Bilder werden stückweise verarbeitet
  - Ausschnitte überlappen so, dass die Ausgaben nahtlos aneinanderpassen



# Augmentierung mit elastischen Deformationen

- Manuelle Annotationen von **Trainingsdaten** für Segmentierungsaufgaben sind besonders arbeitsintensiv
- [Ronneberger et al. 2015] nutzen zufällige **elastische Deformationen** als mächtige Augmentierung
  - Sinnvoll für die Segmentierung von Mikroskopiebildern von Zellen
- Augmentierungen müssen
  - sowohl auf die Bilder als auch auf die Segmentierungskarten angewandt werden
  - zur konkreten Anwendung passen

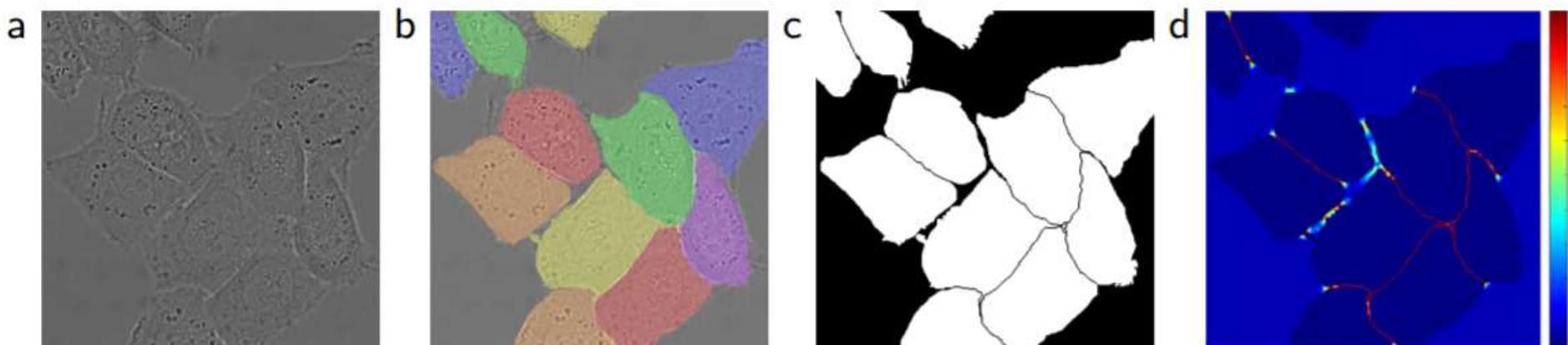


# Gewichtete Verlustfunktion

- **Mittelung der Kreuzentropie** über Pixel ist eine naheliegende Verlustfunktion für Segmentierungsaufgaben
  - Aber: Vernachlässigt kleine Klassen und feine Strukturen
- Zur Zellsegmentierung nutzen [Ronneberger et al. 2015] daher **Pixelgewichte**

$$w(x) = w_c(x) + w_0 \exp\left(-\frac{(d_1(x) + d_2(x))^2}{2\sigma^2}\right)$$

- $w_c$  berücksichtigt Häufigkeit der Klassen,  $w_0$  Grenzen zwischen Zellen
- $d_1/d_2$  sind Abstand zur nächsten und zweitnächsten Zelle

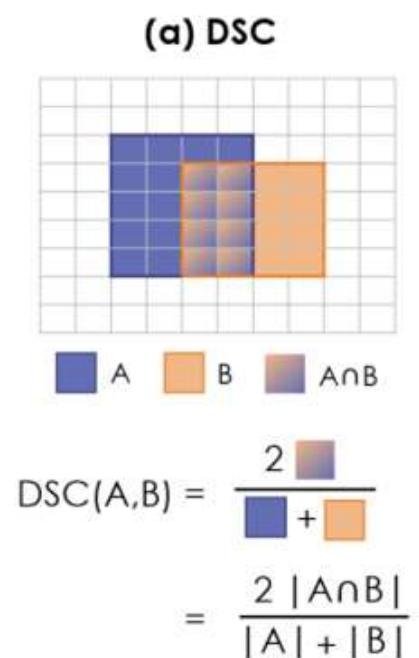


# Alternative: Dice als Verlustfunktion

- *Erinnerung:* In Kapitel 4 haben wir den Dice-Score zur Quantifizierung des Segmentierungsüberlapps eingeführt
- [Milletari et al. 2016] leiten davon eine differenzierbare Verlustfunktion ab, mit der Netzwerke zur Segmentierung trainiert werden können:

$$DL(a, b) = 1 - \frac{2 \sum_i a_i b_i + \epsilon}{\sum_i a_i^2 + \sum_i b_i^2 + \epsilon}$$

- Summen laufen über Pixel  $i$ , kleines  $\epsilon > 0$  zur Regularisierung
- Bestraft Fehlen von Vordergrund-Pixeln auch dann, wenn der Hintergrund dominiert



# Zusammenfassung

- **Vollständig faltungsbasierte Ansätze** ermöglichen eine effiziente Bildsegmentierung
- Das **U-Net** ist ein besonders beliebtes Beispiel
  - Basiert auf einer **Kodierer-Dekodierer-Grundstruktur**
  - **Transponierte Faltungen** lernen optimales Upsampling
  - **Skip-Verbindungen** unterstützen Dekodierung scharfer Karten
- **Erfolgreiches Training** wird ermöglicht durch
  - Geeignete **Augmentierungen**, z.B. elastische Deformationen
  - **Wichtung** der pixelweisen Kreuzentropie oder **Dice-Verlustfunktion**

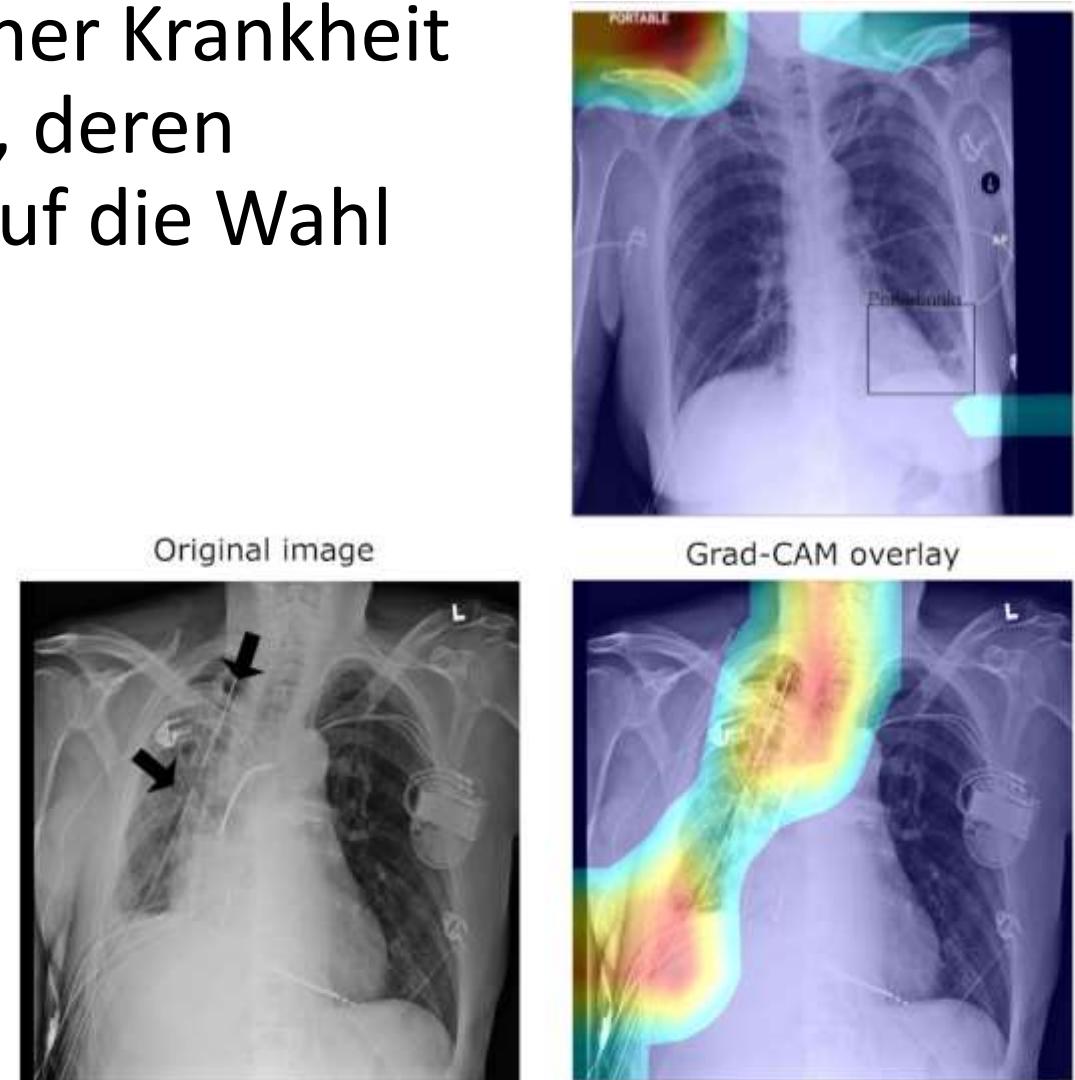
## **6b.5 Einschränkungen Neuronaler Netze**

# Der „Kluger Hans“-Effekt

- Statt **unmittelbaren Anzeichen** einer Krankheit lernen neuronale Netze bisweilen, deren Behandlung oder Auswirkungen auf die Wahl der Bildgebung zu erkennen

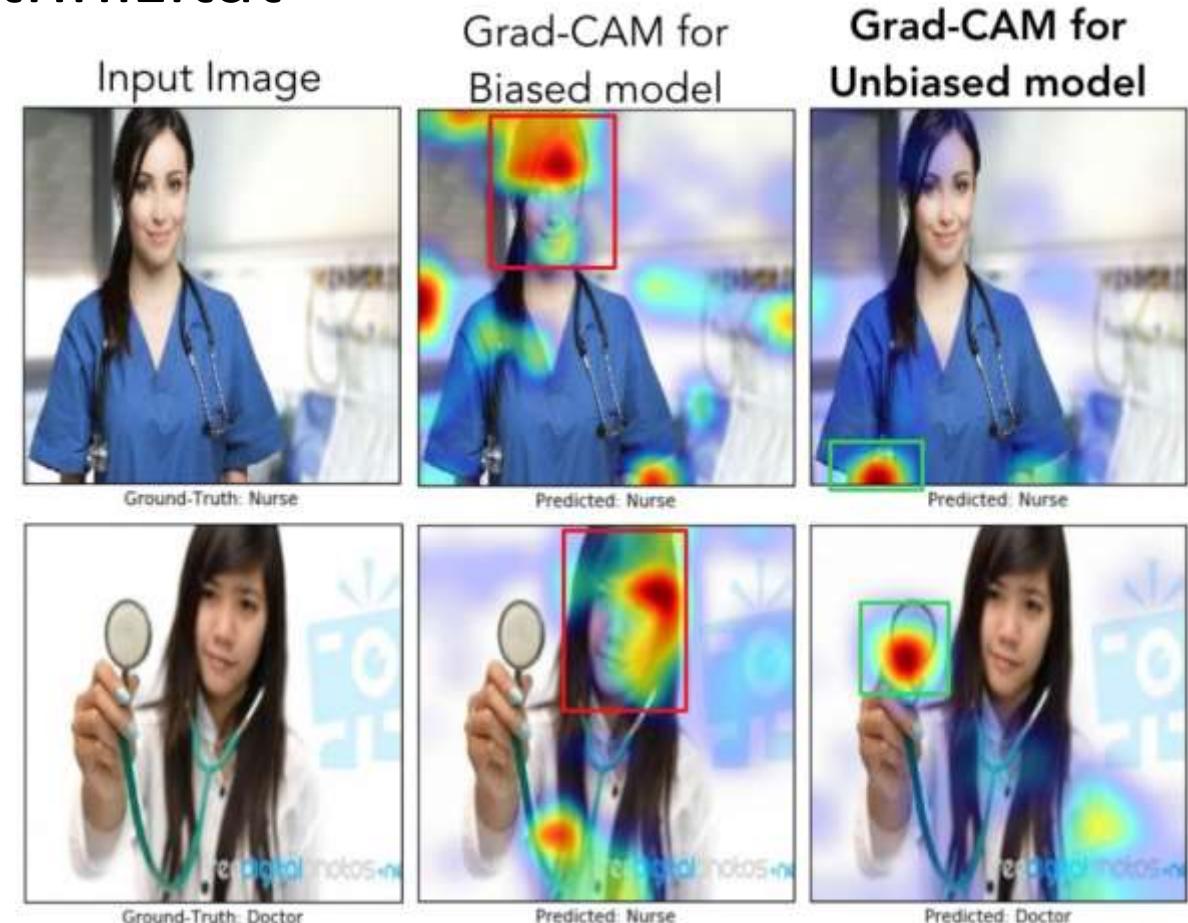
- *Beispiele:*

- Schwerere Fälle werden häufiger mit portablen Röntgengeräten untersucht
- Drainageschlauch ist ein Indiz für einen behandelten Pneumothorax, aber keine sinnvolle Grundlage einer Diagnose



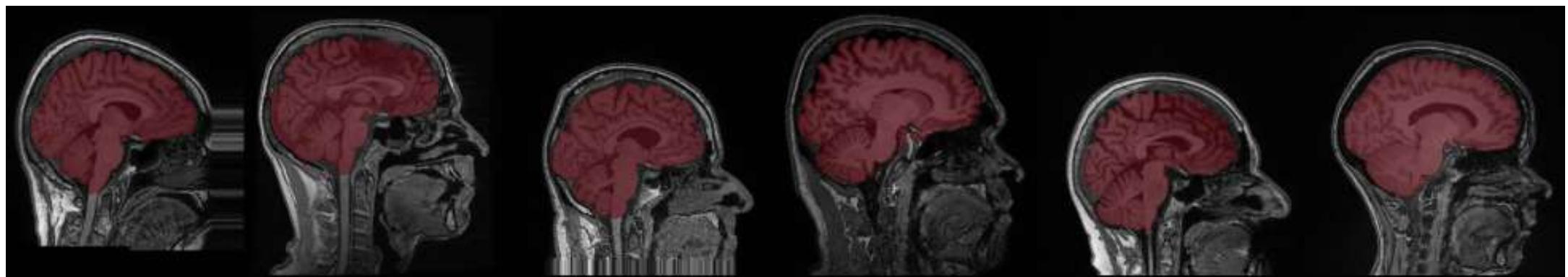
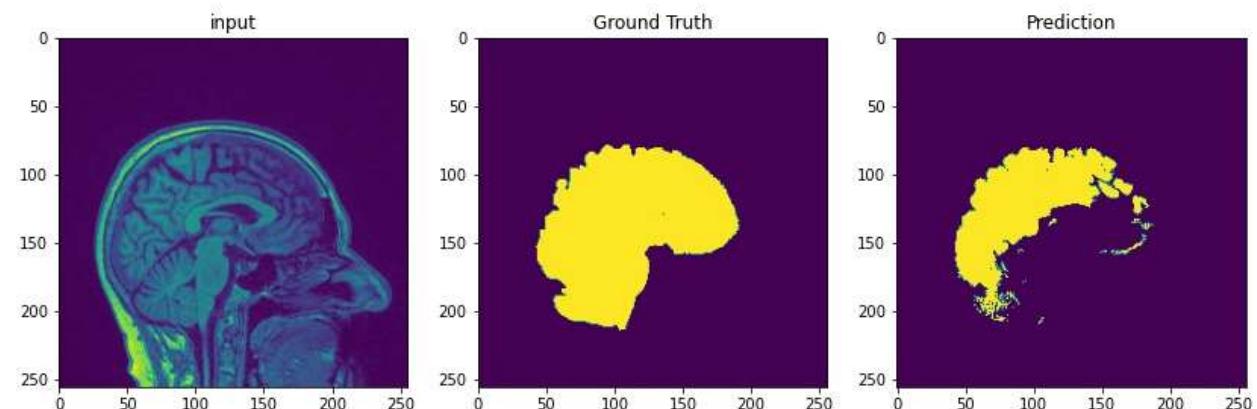
# Mangelnde Fairness

- Neuronale Netze **benachteiligen** häufig Personen aufgrund von Faktoren wie Geschlecht oder Ethnizität
  - Im Beispiel rechts wurde dieser Effekt durch sorgfältigere Auswahl der Trainingsdaten reduziert
- **Visualisierung** kann dabei helfen, solche Effekte aufzudecken
  - Visual Data Analysis, SoSe 2025



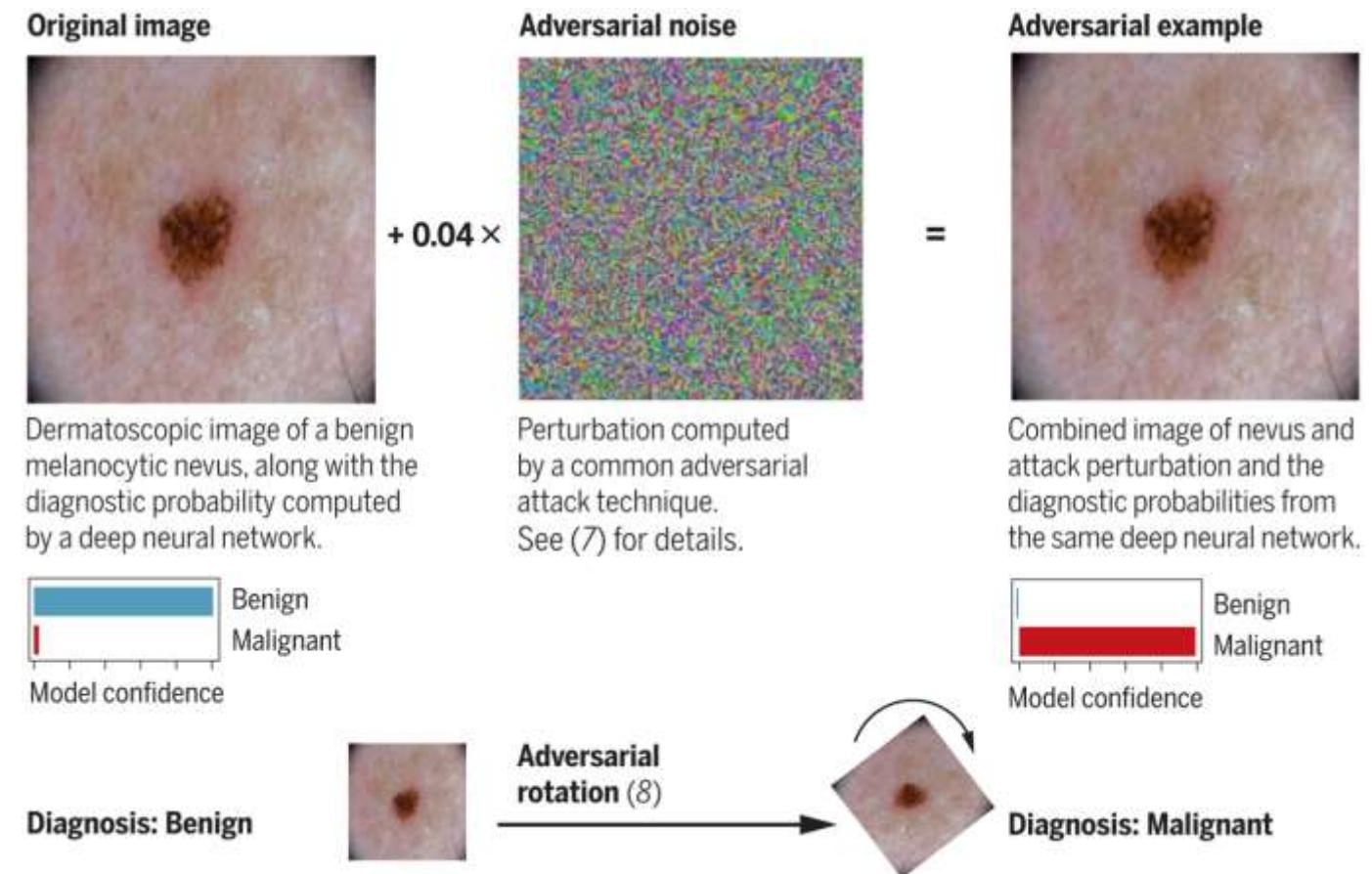
# Domain Shift

- Genauigkeit neuronaler Netze bricht häufig ein, wenn die Bildcharakteristika sich von den Trainingsdaten unterscheiden (“domain shift”)
  - *Beispiel:* Wechsel / Upgrade des Scanners
  - Zusätzliches Problem: „Stilles“ Versagen



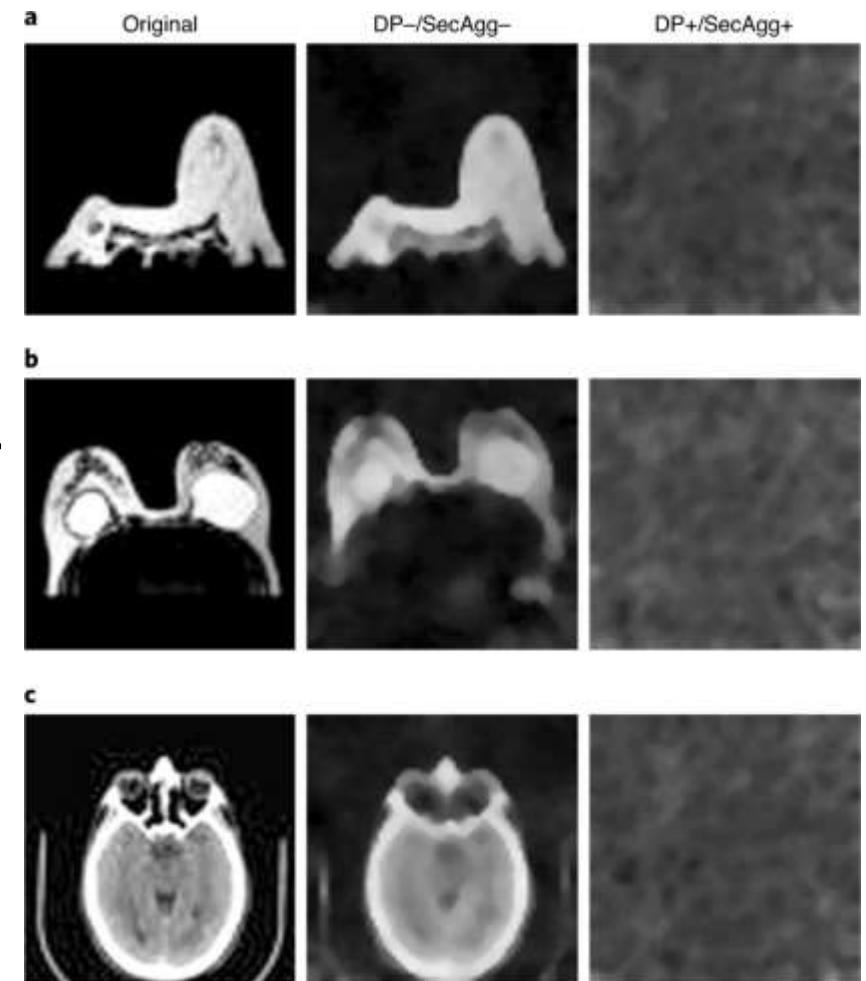
# Angreifbarkeit mit manipulierten Bildern

- Durch gezielte, für das menschliche Auge nicht erkennbare Manipulationen (*engl. adversarial attacks*) lassen sich die Entscheidungen neuronaler Netzwerke umkehren



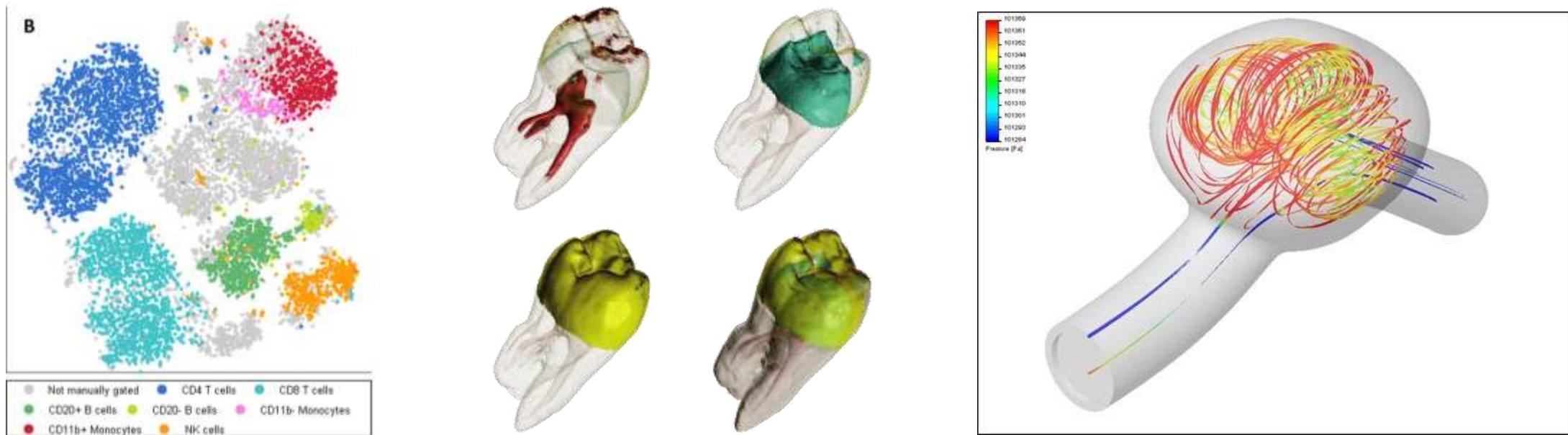
# Rekonstruierbarkeit vertraulicher Trainingsdaten

- Training neuronaler Netze zur medizinischen Bildanalyse erfordert große Mengen **besonders sensibler personenbezogener Daten**
- Ziel des **föderierten Lernens** ist es, die Bilder selbst vertraulich zu halten und nur (Updates der) Modellparameter auszutauschen
- Ohne weitere Sicherheitsvorkehrungen ist es jedoch möglich, erkennbare Bilder aus den Updates zu **rekonstruieren**



# Angebote im SoSe 2025

- Unsere **Lehr-/Lernangebote im SoSe 2025:**
  - BSc-Projektgruppen und BSc-Arbeiten zum Thema „Medizinische Bildanalyse“
  - Vorlesung „Visual Data Analysis“ (4+2 SWS)
  - MSc-Lab „Visualization and Medical Image Analysis“



# Zum Nach- und Weiterlesen

- Ian Goodfellow, Yoshua Bengio, Aaron Courville: “Deep Learning.” MIT Press, 2016 <https://www.deeplearningbook.org/>
- Christopher Bishop with Hugh Bishop: “Deep Learning. Foundations and Concepts.” Springer, 2024 <https://www.bishopbook.com/>
- Y. LeCun, L. Bottou, Y. Bengio, P. Haffner: Gradient-based learning applied to document recognition.  
Proc. of the IEEE 86(11):2278-2324, 1998
- A. Krizhevsky, I. Sutskever, G.E. Hinton: ImageNet Classification with Deep Convolutional Neural Networks. NIPS 2012
- K. Simonyan, A. Zisserman: Very Deep Convolutional Networks for Large-Scale Visual Recognition. ICLR 2015

## Zum Nach- und Weiterlesen

- C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich: Going Deeper with Convolutions. CVPR 2015
- K. He, X. Zhang, S. Ren, J. Sun: Deep Residual Learning for Image Recognition. CVPR 2016
- A. Veit, M. Wilber, S. Belongie: Residual Networks Behave Like Ensembles of Relatively Shallow Networks. NIPS 2016
- O. Ronneberger, P. Fischer, T. Brox: U-net: Convolutional networks for biomedical image segmentation. MICCAI 2015.