LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN



Vorlesung „Computergrafik 1”

Andreas Butz, Axel Hoppe

EINFÜHRUNG, ORGANISATORISCHES

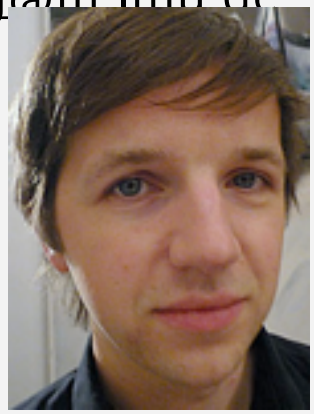
Wer? Wo?





- Prof. Dr.-Ing. Axel Hoppe, MD.H und LMU
  - axel.hoppe@ifi.lmu.de
- Prof. Dr.-Ing. Andreas Butz, LMU
  - butz@ifi.lmu.de

- Dominikus Baur, dominikus.baur@ifi.lmu.de
- Sebastian Boring, sebastian.boring@ifi.lmu.de
- Tilman Dingler, Til-D@gmx.de
- Dominic Bremer







- mimuc.de/cg1

ECTS? SWS?

- Pflichtmodul im Bachelor Medieninformatik, 4.FS
  - ECTS-Credits: 4 (Vorlesung) + 2 (Übung)
- Kann in die Diplomprüfung Medieninformatik eingebracht werden
  - 3 SWS Vorlesung (immer)
  - 2 SWS Übung (bei bestandenem Schein)

- Wieso die gleiche VL für Bachelor und Diplom??

- Bekomme ich hier irgendwas billiger?
  - CG1 = 0.9 * 3DCG + 0.6 * 2DG = 1.5 * 2SWS = 3SWS (also nein)

Vorlesungskonzept

- HISQIS: „In der Vorlesung Computergrafik 1 wird ein Überblick über 2D- und 3D-Computergrafik gegeben.“

- 3D-Grafik: DM+MT: VRML/X3D
- 2D-Grafik:
  - 2D-Vektorgrafik (DM+MT: SVG, Flash)
    - ...fällt bei der 3D Grafik quasi nebenbei ab!
  - 2D Bitmap Grafik (DM+MT: Fotos, Bildformate)
    - DM+MT: Bildverarbeitung, ehem. Vorlesung 2DG
- Überblick??
  - wichtigstes Handwerkszeug des Gebietes beherrschen

Stoffplan der Vorlesung

Termin	Nr.	Thema	Bem.
23.4.	1	Organisatorisches, Kontext	
30.4.	2	Historie	
7.5.	3	Peripherie	Hoppe
14.5.	4	Rendering Pipeline, Geometrie	
21.5.			Himmelfahrt
28.5.	5	Farbe, Kamera, Projektion	
4.6.	6	Rasterisierung, Shading, Rendering	
11.6.			Fronleichnam
18.6.	7	Sichtbarkeitsbestimmung, prozedural etc..	
25.6.	8	Pixelgrafik-Basics, Konvolution, Skalierung	
2.7.	9	Fouriertransformation, FFT	
9.7.	10	Bildverbesserung	Butz
16.7.	11	Segmentierung, morphologische Op.	
23.7.	12	Klassifikation, evtl. Klausurtermin	

Geschichte

- 2001 aufwändige und sehr überzeugende Gesichtsanimationen von virtuellen Charakteren im Film „Shrek“





Rotation: Herleitung der Berechnungs-

I. $x = r \cos \phi$

II. $y = r \sin \phi$

III. $x' = r \cos (\alpha + \phi) = r \cos \phi € \cos \alpha € - r \sin \phi € \sin \alpha$

IV. $y' = r \sin (\alpha + \phi) = r \cos \phi € \sin \alpha € - r \sin \phi € \cos \alpha$

- anschließend (I) in (IV) einsetzen:

  $x' = x \cos \alpha € - y \sin \alpha$

  $y' = x \sin \alpha € + y \cos \alpha$

Separabilität

$$F(u,v) = \frac{1}{N^2}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}(um+vn)\right] =$$

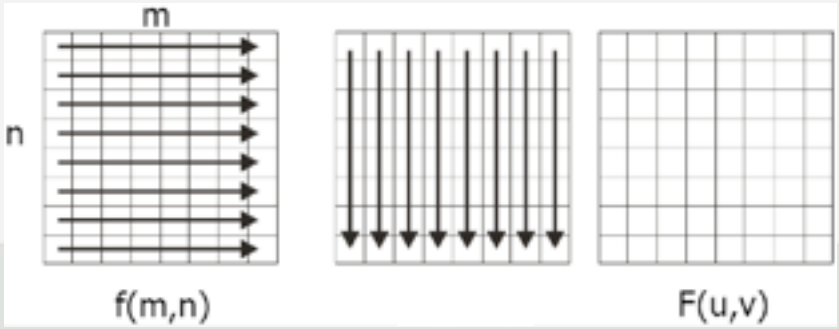
$$= \frac{1}{N^2}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}um\right]\exp\left[-i\frac{2\pi}{N}vn\right] =$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\left(\frac{1}{N}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}vn\right]\right)\exp\left[-i\frac{2\pi}{N}um\right] =$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\exp\left[-i\frac{2\pi}{N}um\right]\left(\frac{1}{N}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}vn\right]\right) =$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\exp\left[-i\frac{2\pi}{N}um\right]F_u(m)$$

Vorgehensweise: $F_u(m)$ für alle Spalten m berechnen und dann bei den Zeilen verwenden.



Stoffplan der Übung

1	Einführung in C/C++
2	Einführung ins Qt Framework
3	Projektionen
4	Primitive und Extrusionskörper
5	Interaktion
6	Texturen und Licht
7	HSL-Filter
8	Konvolution
9	Frequenzraumtransformationen
10	Hough-Transformation
11	Segmentierung
12	Region Growing

Übungen

- Ergänzung zur Vorlesung
- Besuch optional
- Aber: Erfolgreiche Bearbeitung gibt Bonuspunkte in der Klausur
- Abgaben der Übungsblätter in Gruppen von bis zu 5 Personen



Plagiate

- Plagiate?
- … besser nicht!

Service

- PDF der Vorlesungs-Folien nachts zuvor
- Podcast am Tag danach (Video der Folien mit Ton des Dozenten)
  - auch auf iTunes U
- Links in den Folien, wo passend
- Literaturhinweise in den Folien

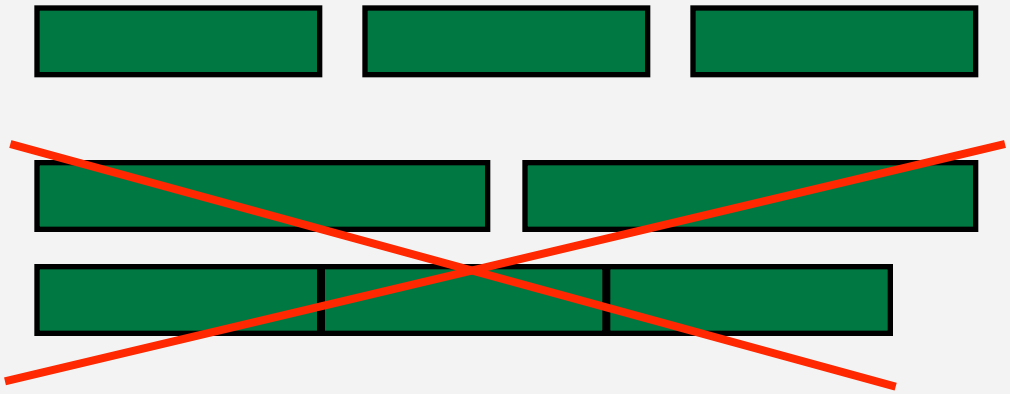
- PDF der Übungsblätter
- PDF der Übungsfolien
- weitere Materialien zur Übung

- Forum zur Vorlesung



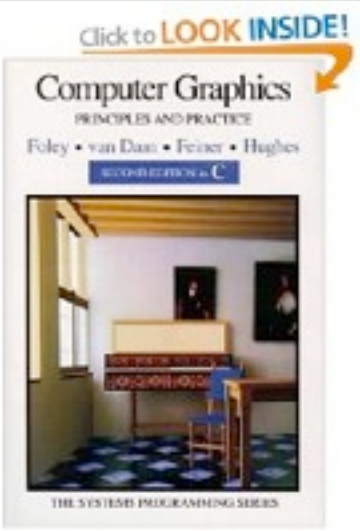
Timing der Vorlesung?

- Vorlesung 3-stündig
  - Beginn s.t. oder c.t. möglich -> abstimmen
  - Pausen?
    - Vorschlag: 3x45min. mit je 10/15 min. Pause
    - gute Alternativen??



Welche Bücher rauskramen?

- Bücher und Skripte zur linearen Algebra
- „Computer Graphics: Principles and Practice in C“ (2nd Edition) by James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes
- „Grundlagen der Bildverarbeitung“ von Klaus D. Tönnies
- Diese Bücher bilden die Grundlage der Vorlesung
  - weitere bei Spezialthemen angegeben
- Ziel: Vorlesung und Übung sollen auch ohne die





Klausur

- Klausur am Ende des Semesters (z.B. letzter VL-Termin??)
- geprüft wird, ob das Handwerkszeug sitzt
  - Verständnis, nicht auswendig lernen!
  - Daumenregel: Wer die Übungsblätter geschafft hat, braucht die Klausur nicht zu fürchten!
- Punkte aus den Übungsblättern zählen als Bonuspunkte in der Klausur (Größenordnung max. 10-20%)

- Wiederholungsklausur wie vorgeschrieben
  - vermutlich kurz vor Beginn des folgenden Semesters



WERBUNG

Diplomarbeiten

- Derzeit viele (16) freie Projekt- und Diplomthemen!
- mimuc.de -> Lehre -> Proj.+Dipl. Arb. -> freie Themen
  - http://www.medien.ifi.lmu.de/lehre/arbeiten/themen.xhtml

- am Lehrstuhl, integriert in Grundlagenforschung
- bei BMW, etwas angewandter, (bezahlt)

- viele der Themen sehr visuell + interaktiv



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik und -animation

Einführung und historischer Kontext



Inhalte der Vorlesung

1. Einführung
2. Historischer Kontext
3. Eingabegeräte und
4. Ausgabegeräte für die Computergrafik
5. 3D-Repräsentationen
6. Allgemeine Rastergrafik
7. Rastergrafik
8. Verfahren zur verbesserten Darstellung von Rastergrafiken
9. Farben, Farbmodelle
10. Koordinatensysteme
11. Transformationen
12. Synthetische Kamera
13. Sichtbarkeitsbestimmung
14. Projektionen
15. Shading, Texturierung, Ausleuchtung – Inszenierung
16. Texturen und Materialien
17. Modelle für Globale Beleuchtung
18. Animationstechniken
19. 3D-Editoren
20. Grafikprogrammierung



Was brauche ich für 3D-

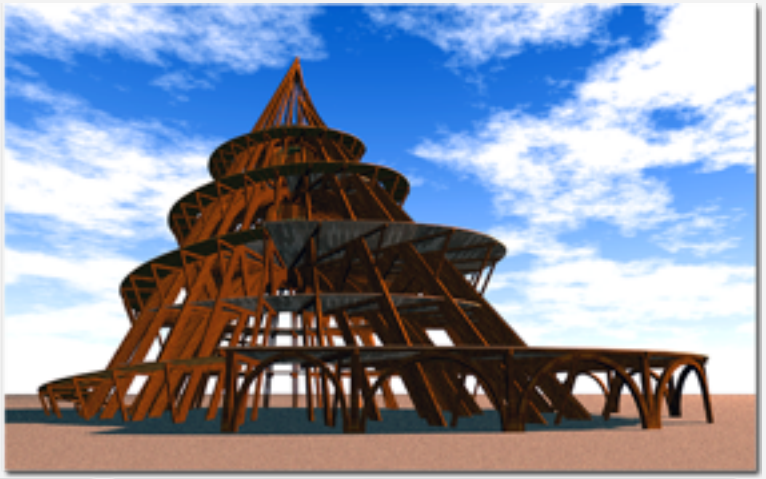
- Sinn für Gestaltung und Ästhetik
- Räumliches Vorstellungsvermögen
- Algorithmische Grundlagen
- (Programmierkenntnisse)
- geeignete Werkzeuge und Disziplin bei deren Nutzung
- Mathematische Grundkenntnisse

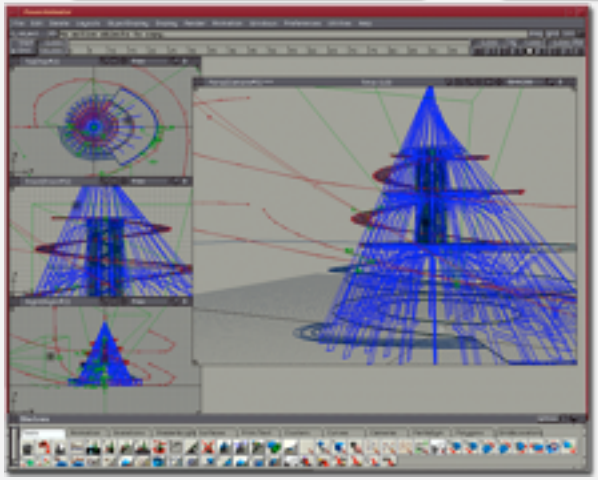
Literatur

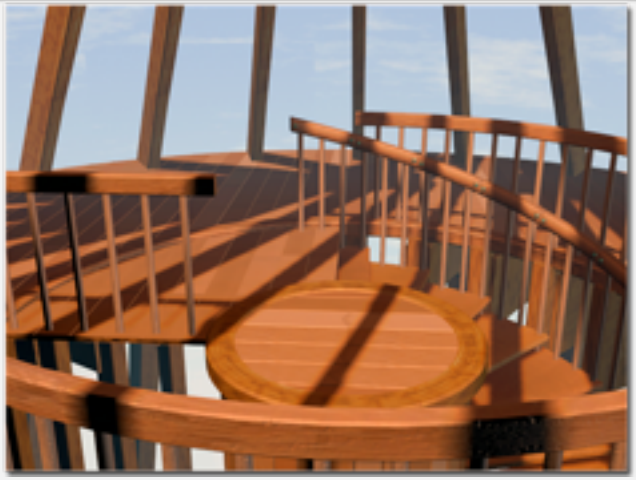
- Foley, van Dam, Feiner, Hughes: *Computer Graphics, Principles and Practice.* Second edition, Addison Wesley. ISBN 0–201–84840–6
- Alan Watt: *3D Computer Graphics.* Addison Wesley. ISBN 0–20–139855–9
- David F. Rogers, J. Alan Adams: *Mathematical Elements for Computer Graphics.* Second edition, McGraw Hill. ISBN 0–07–053530–2
- David F. Rogers: *Procedural Elements of Computer Graphics.* Second edition, William C Brown Publishers. ISBN 0–07–053548–5

Zur Person



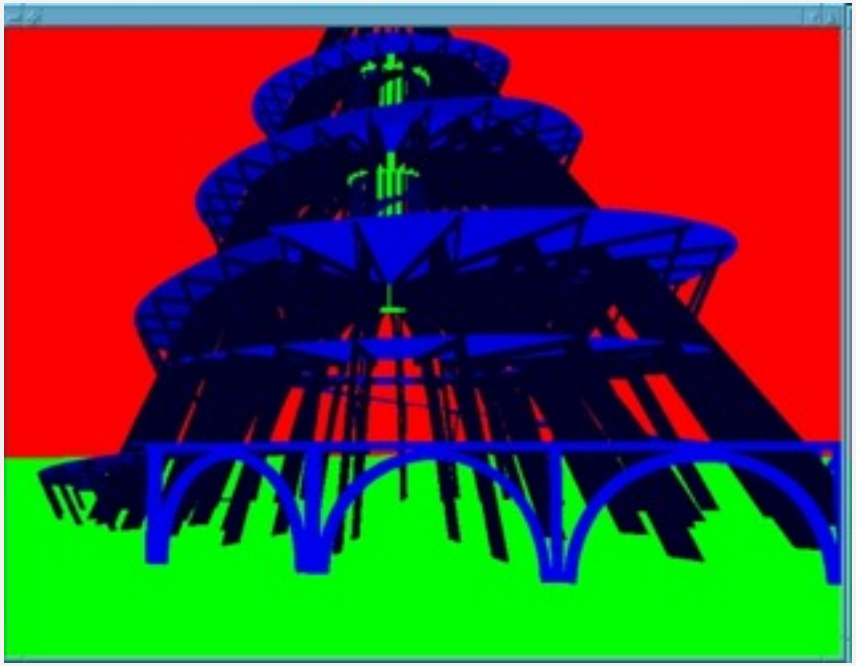




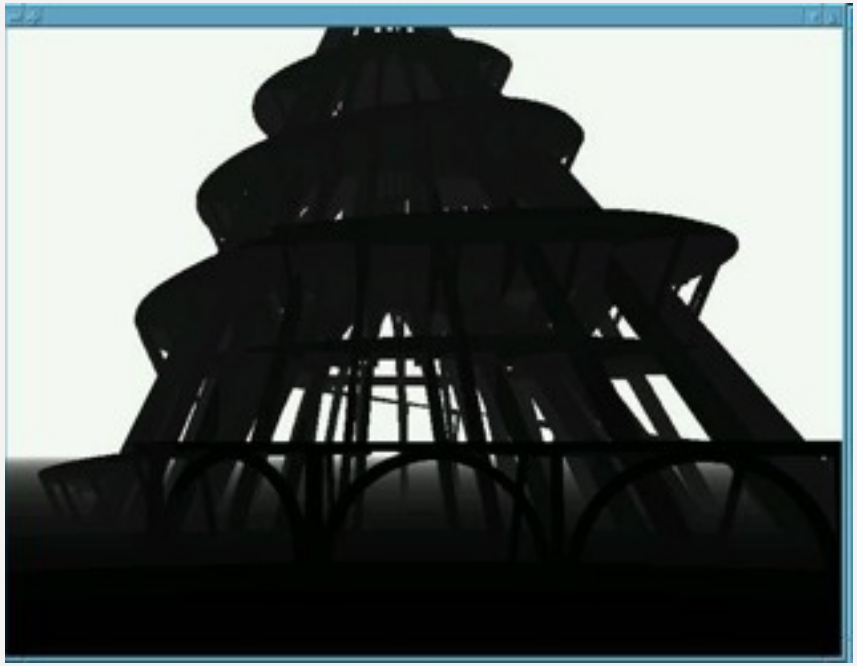


Zur Person

ID-Buffer



Z-Buffer



Zur Person

Physikalisch korrekte Lichtquelle, „Absaufen“ der Binnenstruktur



Tiefenabhängige Farbkorrektur, Akzentuierungen



Zur Person

Unbefriedigende Kontrastverhältnisse

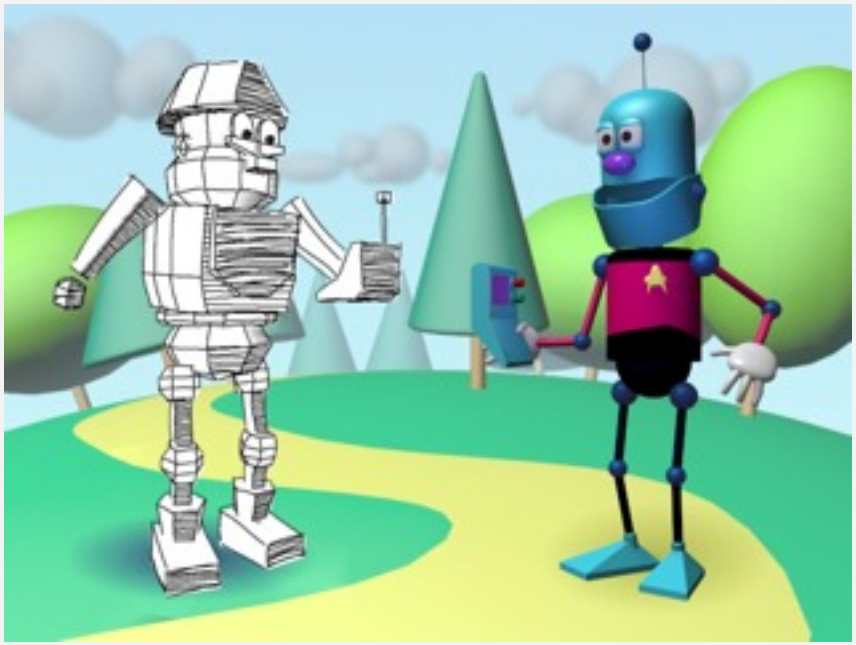


korrigierte Kontraste und Akzentuierungen

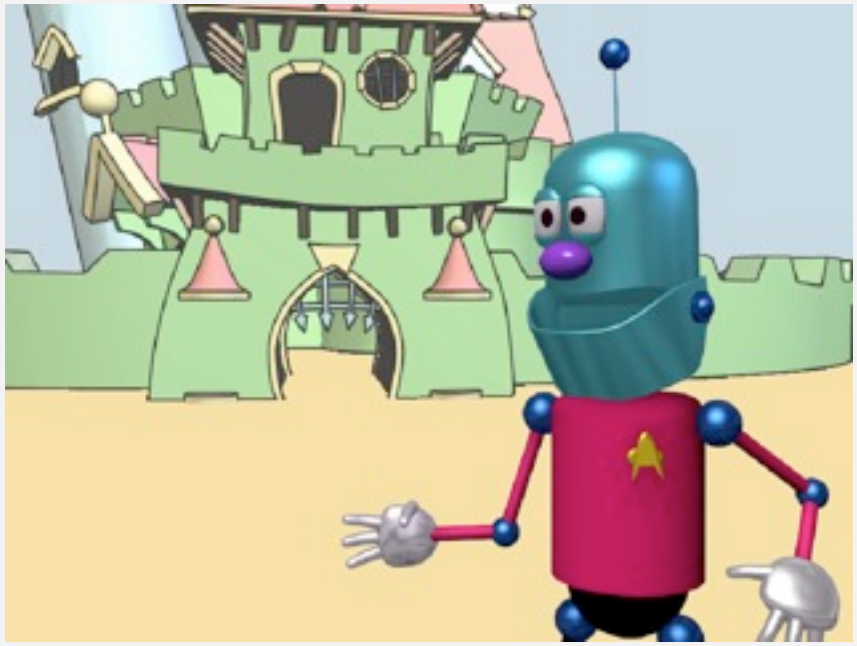


Zur Person

Mix mit Tuschezeichnungs-Stil



Mix mit Comic-Stil



Zur Person



Zur Person





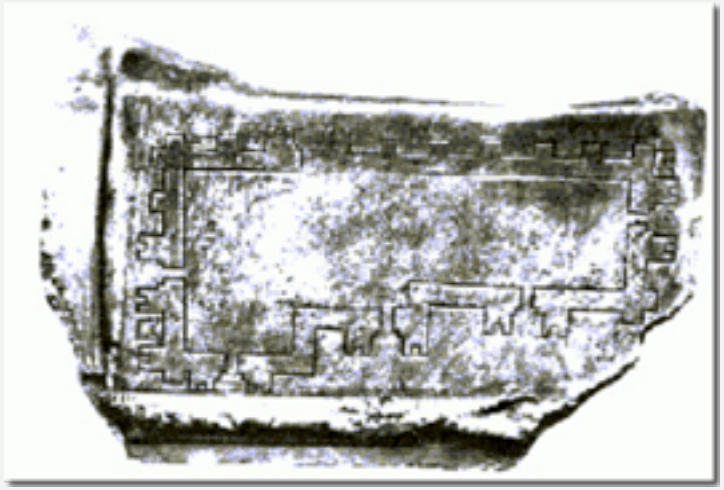
3D-Computergrafik

HISTORISCHER KONTEXT

Vorgeschichte

- Eiszeitliche Darstellung (vor ca. 15.000 Jahren, Höhle „Les Trois Frères“): bereits in Höhlenzeichnungen Wunsch und Bestreben, Bewegung darzustellen
- 2150 v. Chr.: Technische Zeichnung eines Gebäudes aus Mesopotamien





Vorgeschichte

- 1525 Albrecht Dürer: „Der Zeichner der Laute“
- 1572 Hans Vredeman de Vries: „Lazarus vor dem Palast des Reichen“





Vorgeschichte

1831 „Phantoskop“ (auch „Phänakistoskop“) von J. A. Plateau



1834 „Zoetrope“ von Horner



Vorgeschichte

- 1892 Anmeldung des Patents auf den Zeichenfilm durch Emile Reynaud
- er malte seine Szenen auf perforiertes Zelluloid;
- 1892 „Théâtre Optique“
- längster Film „Le Pauvre Pierrot“ von 1892 mit einer Vorführdauer von 15 Minuten
- 1908 „Le Cauchemar du Fantoche“ von Emile Cohl
  - 80 min
  - Trickfilmfigur „Fantoche“ als Held mehrerer Abenteuer





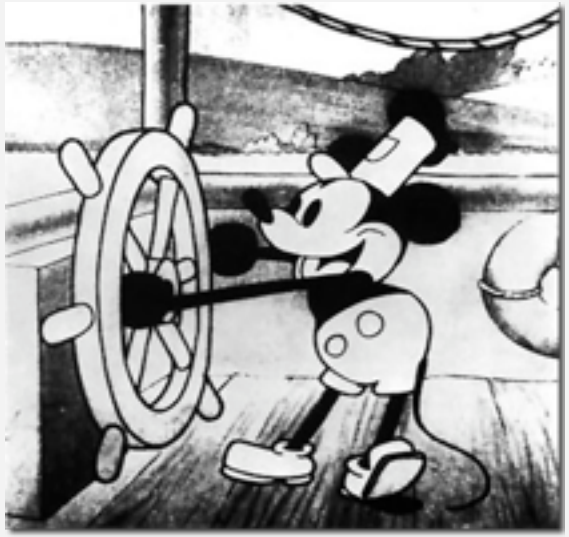
Vorgeschichte



- 1909 „Gertie the Trained Dinosaur“
  - von Winsor MacCay
  - ca. 10.000 handgezeichnete Bilder
- 1915 Erfindung der Cel-Animation

Vorgeschichte

- 1928 „Mickey Mouse“ von Walt Disney im ersten Kurzfilm „Steamboat Willie“
- 1837 „Snow White and the Seven Dwarfs“ – erster abendfüllender Zeichentrickfilm

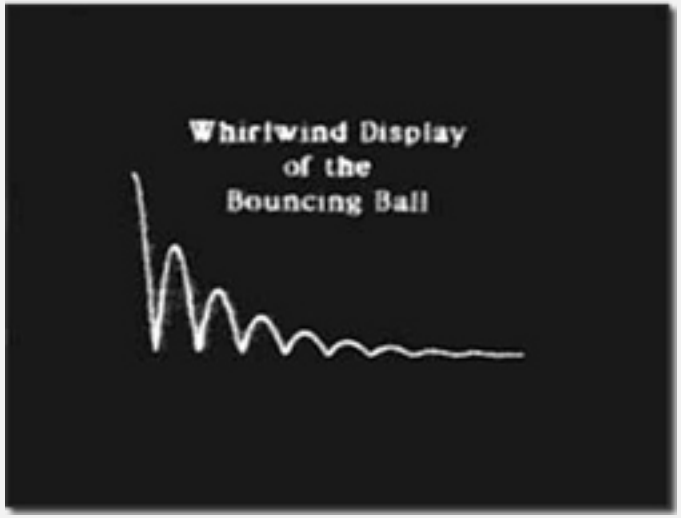




Geschichte

- 1949 erste Computergrafik auf Whirlwind-Computer des MIT
- 1951 MIT: Simulation eines springenden Balls am Oszilloskop





Geschichte

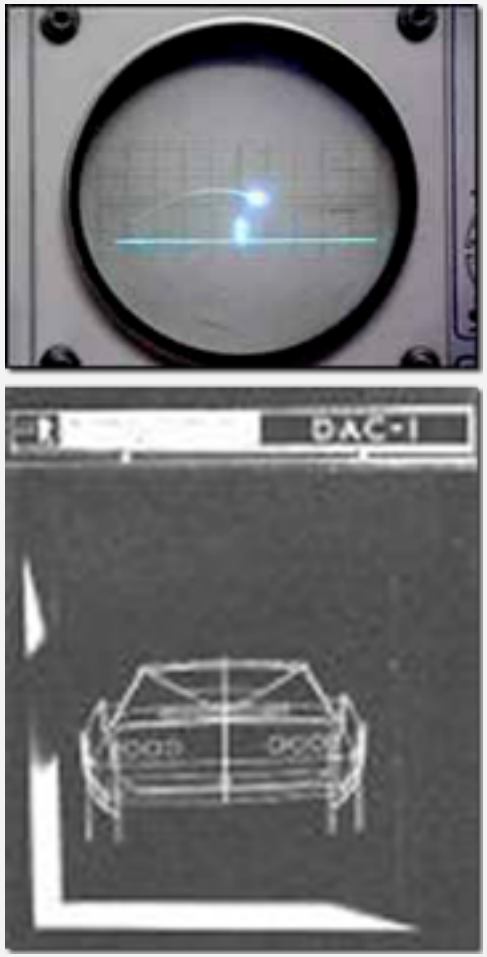
- 1952 Einsatz der Computergrafik zur Kennzeichnung von Flugobjekten auf Radarbildschirmen
- SAGE-Computer mit 82 Grafikkonsolen, erster Einsatz eines Lichtgriffels





Geschichte

- 1958: „Tennis for Two“ vom Physiker William Higinbotham am Brookhaven National Laboratory entwickelt und konstruiert
- 1959 DAC-1: Erstes computergestütztes Zeichnungssystem – gemeinsame Entwicklung von GM und IBM, wurde erst 1964 bei der JointComputer Conference in Detroit vorgestellt



Geschichte

- 1961 „Spacewar“:
  - erstes Videospiel
  - entwickelt von Steve Russel am MIT für den DEC PDP-1





Geschichte

- 1963: Ivan Sutherland
  Promotion über SKETCHPAD:
  „A Man-Machine Graphical Communication System“
  - Bildkomposition aus grafischen Standardelementen
  - Interaktion mit Tastatur und Lichtgriffel zur Arbeit mit Menüs
  - Verwaltung grafischer Daten



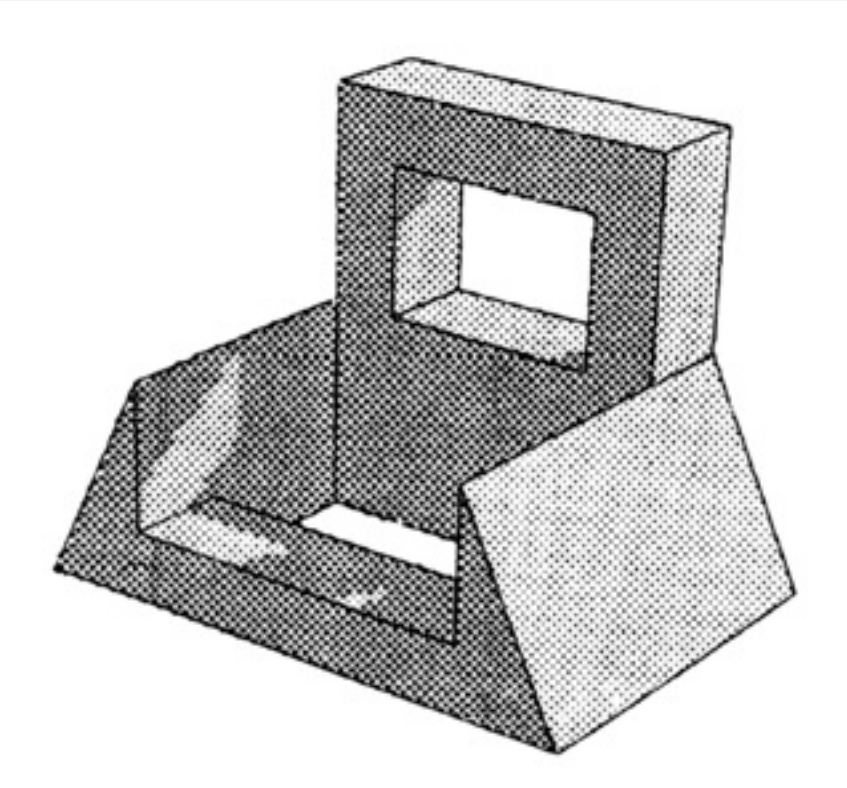


Geschichte



- Mitte der 60er Jahre: Beginn einer Vielzahl von Forschungsprojekten (am MIT, General Motors, Bell Telephone Lab., Lockheed Aircraft usw.)
- 1965: Erstes kommerzielles Vektor-Display von IBM (Preis ca. 100.000 $)

Geschichte



- 1967 Erster Bildspeicher-Display von Tektronix (bietet einem breiten Interessentenkreis Zugang zur Computergraphik, Preis ca. 10.000 $)
- 1968 erste mittels Photon Tracing (spezielles Raytracing-Verfahren) generierte Bilder

Geschichte



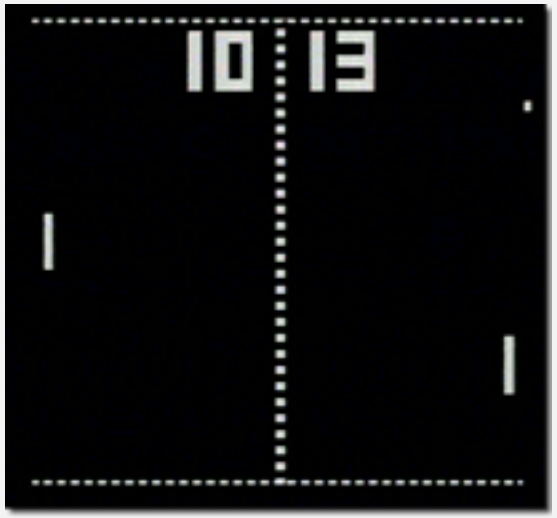
- 1969 Das erste CAD Drahtgitter-Zeichnungs-System wird von E&S veröffentlicht: LDS-1 (Line Drawing System)
- 1969 John Warnok entwickelt den Warnok Recursive Subdivision Algorithmus für Hidden Surface Romoval
- 1969 Alan Kay entwickelt die Metapher eines graphischen User Interface (GUI)

Geschichte

- Anfang der 70er Jahre: erste kommerzielle CAD/CAM-Systeme kommen auf den Markt
- 1971 Raster-Scan-Prinzip von M. Noll (Bell Lab.) vorgeschlagen

Geschichte

- 1972 „PONG“
  - entwickelt von Nolan Bushnell, dem späteren Gründer von Atari
  - Zahlreiche Varianten werden erstellt
  - Riesiger kommerzieller Erfolg für Atari

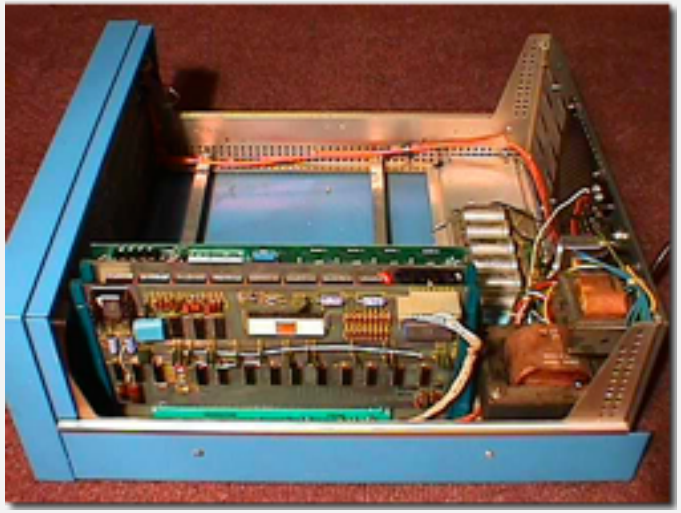




Geschichte

- 1972 „Altair 8800“
  - Erster Heimcomputer als Bausatz 2 MHz Intel 8080
  - 256 Bytes Standard-RAM
  - Benutzungsschnittstelle: Schalter und LEDs an Vorderseite
- Preis:
  - 395 $ (Bausatz),
  - 495 $ (montiert)





Geschichte



- 1973 Erste Konferenz der SIGGRAPH (Special Interest Group on Computer Graphics) der ACM (Association of Computing Machinery)
- beschäftigt sich ausschließlich mit Computergrafik
- ca. 1.200 Teilnehmer

Geschichte

- 1973 Erstes Objekt, das vollständig im Computer erzeugt wurde:
  - steht immer noch in Vegreville, Alberta, Kanada: „The Easter Egg Capitol of the World“
  - von Ronald Resch, Pionier im Bereich Computerkunst
- 1973 Frank Crow entwickelt Anti-Aliasing-Verfahren zur Kantenglättung

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik und -animation

historischer Kontext – Fortsetzung



Geschichte

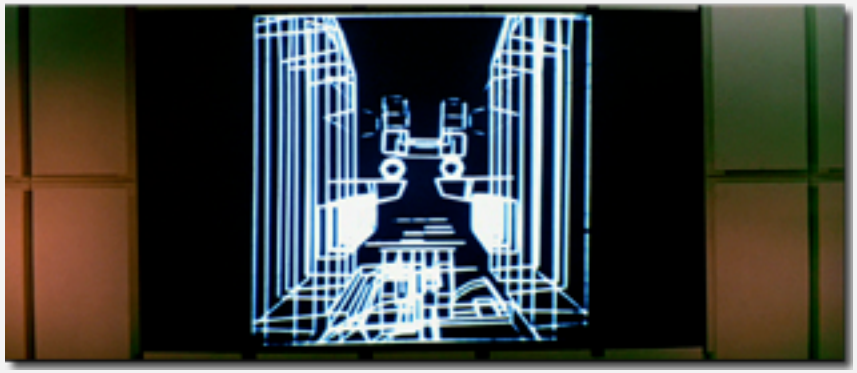
- 1974 „Hunger (La Faim)“:
  - von Peter Foldes, 11 min Dauer
  - Parabel über Kontrast zwischen Nahrungsüberfluss und Hunger von Millionen
  - erster, vollständig mit Hilfe des Computers animierter Film



Geschichte

- 1976 „Future World“: Erste Anwendung von Computergrafik in einem Kinofilm: Eine Drahtgitter-Hand von Edwin Catmull
- 1977 „Star Wars—A New Hope“: schematische Darstellung und Animation des Todessterns





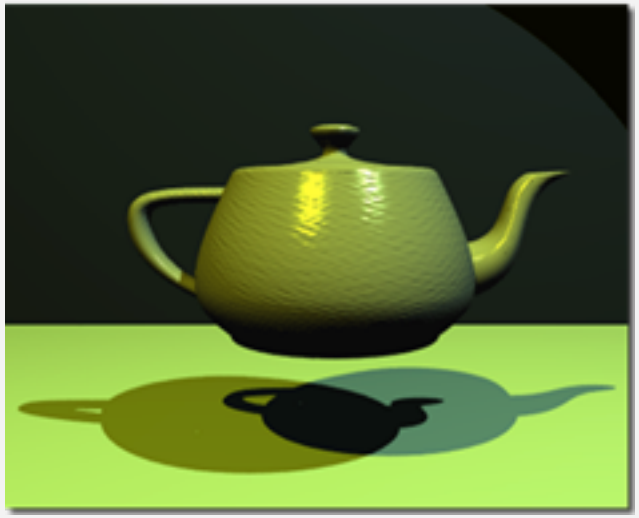
Geschichte

- Graphische Unterprogrammpakete (PLOT10, CAL-Comp)
- kommerzielle Raster-Displays
  - 512 × 512 Pixel,
  - 8 Bit pro Pixel,
  - Preis ca. 100.000 DM

Geschichte

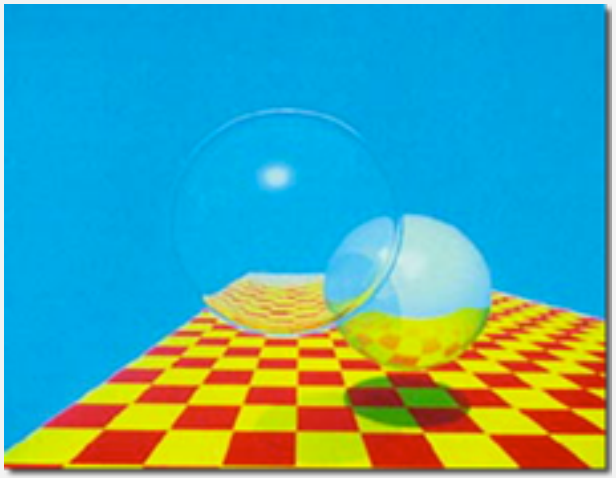
- Shading-Verfahren (Gouraud 1971, Phong 1975)
- Texturierung (Catmull 1974)
- Beleuchtungsverfahren (Phong 1975, Blinn 1977)
- 1975 legendärer Utah Teapot
- Schattenwurf (Crow 1977)

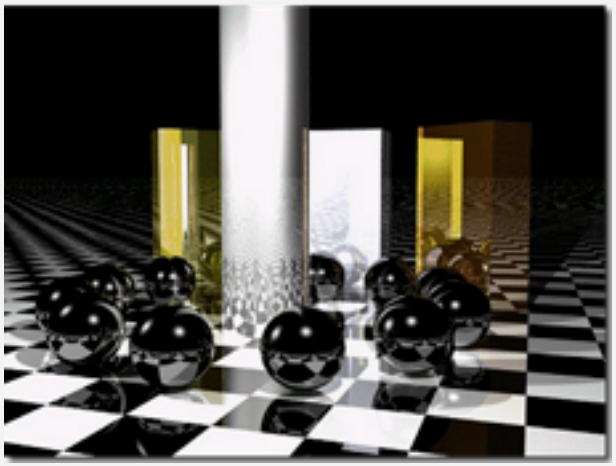




Geschichte

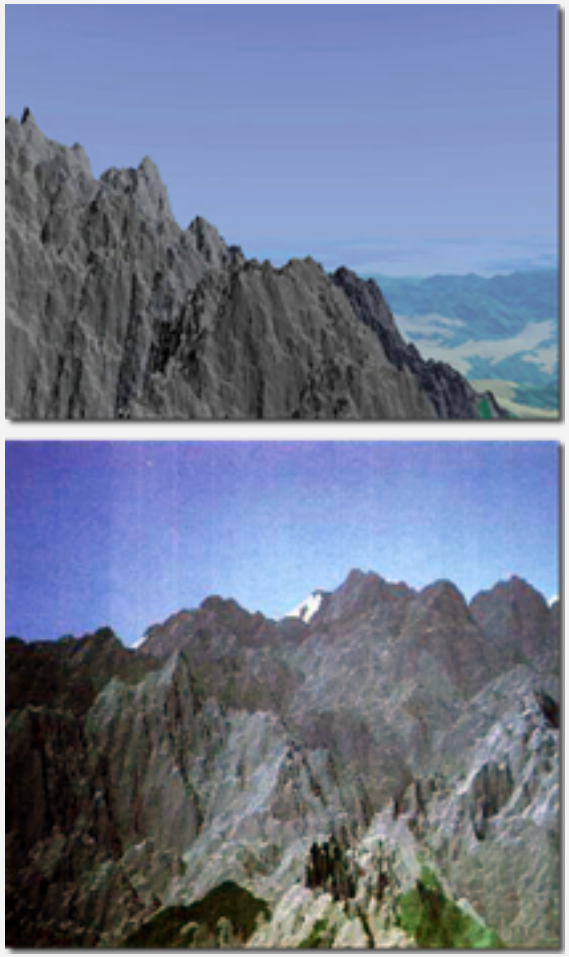
- 1977 Erste Vorschläge zur Standardisierung von Grafiksoftware – CORE
- 1979 Erstmalige Darstellung von Spiegelung, Reflexion und Transparenz mit dem Raytracing-Verfahren
- 1979 Gründung der Computergrafik-Abteilung von LucasFilm durch George Lucas





Geschichte

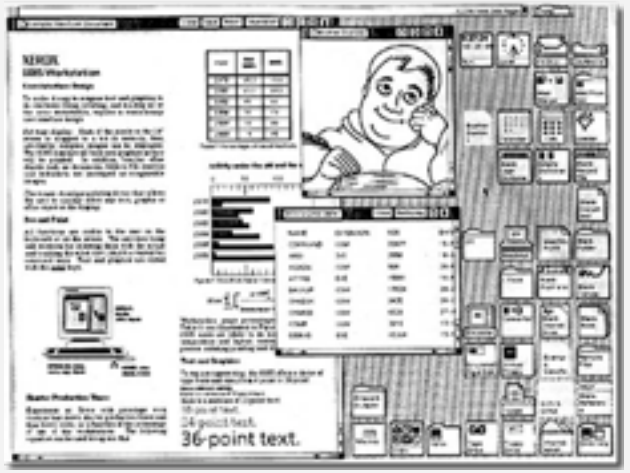
- 1980 „Vol Libre“
  - von L. Carpenter, Boeing
  - gezeigt auf der SIGGRAPH’80
  - Flug durch fraktale Landschaft
  - ausgeschlossen von der Jury der SIGGRAPH’80



Geschichte

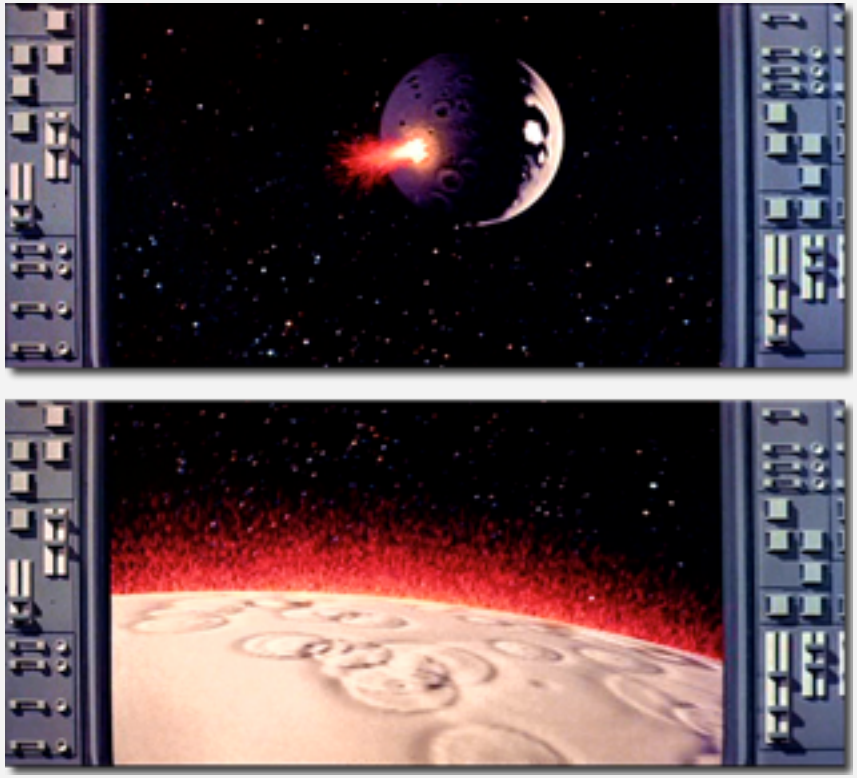
- 1981
  - Der „Xerox Star“
  - mit grafischer Benutzungsoberfläche, Fensterverwaltung und Desktop-Metapher





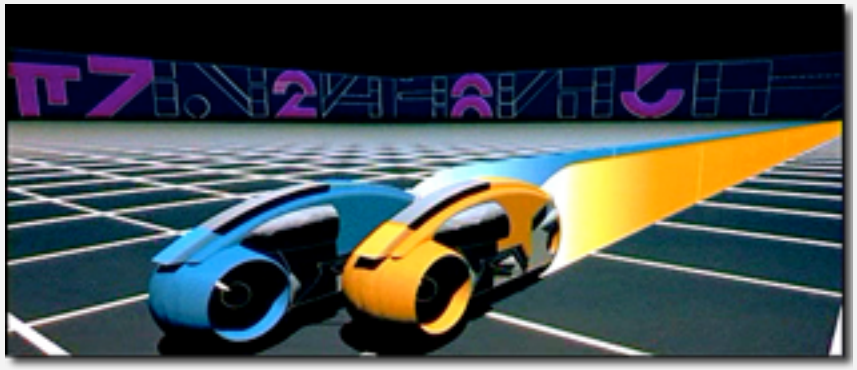
Geschichte

- 1982 „Star Trek II: Der Zorn des Khan (The Wrath of Khan)“
  - Erste vollständig computer-erzeugte Sequenz in einem Kinofilm
  - Landschaftsgenerierung
  - Partikelsystem für die Genesis-Sequenz



Geschichte

- 1982
  - „Tron“
  - ca. 30 min Computer-Animationen
- 1984
  - „The Last Starfighter“
  - Computergrafik von Digital Productions





Geschichte

- 1981 Erstes Rendering-System REYES („Rendering Everything You Ever Saw“ von Loren Carpenter für LucasFilm, wird später u. a. zu Renderman weiterentwickelt)
- 1982 Gründung Silicon Graphics Inc. (SGI), J. Clark, (Entwicklung von Hochleistungsrechnern für grafische Anwendungen)







Kleiner Exkurs

Grafik-Hardware in den 1980er Jahren

Exkurs: SGI

- SGI: Workstations mit Spezialisierung auf interaktive Computergrafik
- Trennung zwischen CPU und Grafik-Subsystem
- CPU:
  - RISC: „Rechnen mit **reduziertem** Befehlssatz“
  - CISC: „Rechnen mit **komplexem** Befehlssatz“

Exkurs: SGI

- Prozessor mit RISC-Architektur:
  - kleiner (reduzierter) Befehlssatz
  - die Befehle können weniger, sind aber dadurch schneller verarbeitet
- Prozessor mit CISC-Architektur:
  - großer Befehlssatz
  - Befehle können mehr (komplexer), werden dadurch aber langsamer verarbeitet

Exkurs: SGI

- 1982: Gründung von SGI
- 1984: IRIS 1000
  - grafisches X-Terminal
  - Motorola 68000-Mikroprozessor
  - IRIX-Betriebssystem
  - 8 MHz
  - $ 37.500,–

Exkurs SGI

- Iris 1000



Exkurs SGI

- Indigo
- gedacht als Desktop-Lösung



Exkurs SGI

- Onyx 2
- Komplexe Visualisierungen
- Kino – Beispiel „Independence Day“



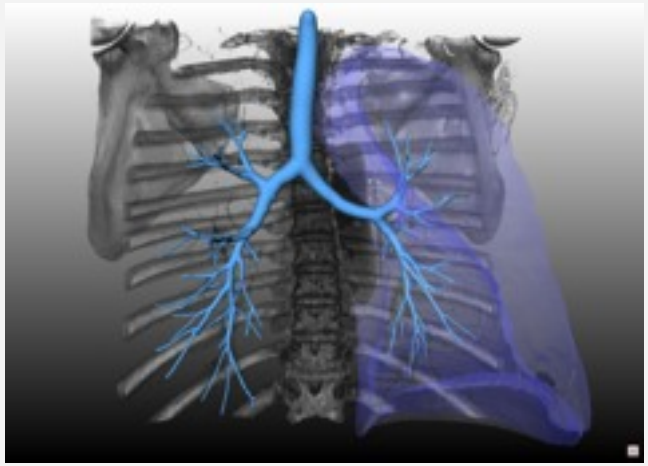
Exkurs: SGI

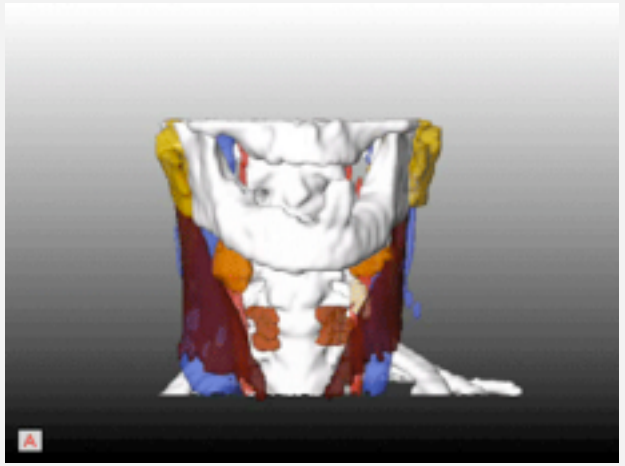




- Betriebssystem Irix
- Low-Level-API Irix GL, später Open GL
- High-Level-APIs wie Open Inventor

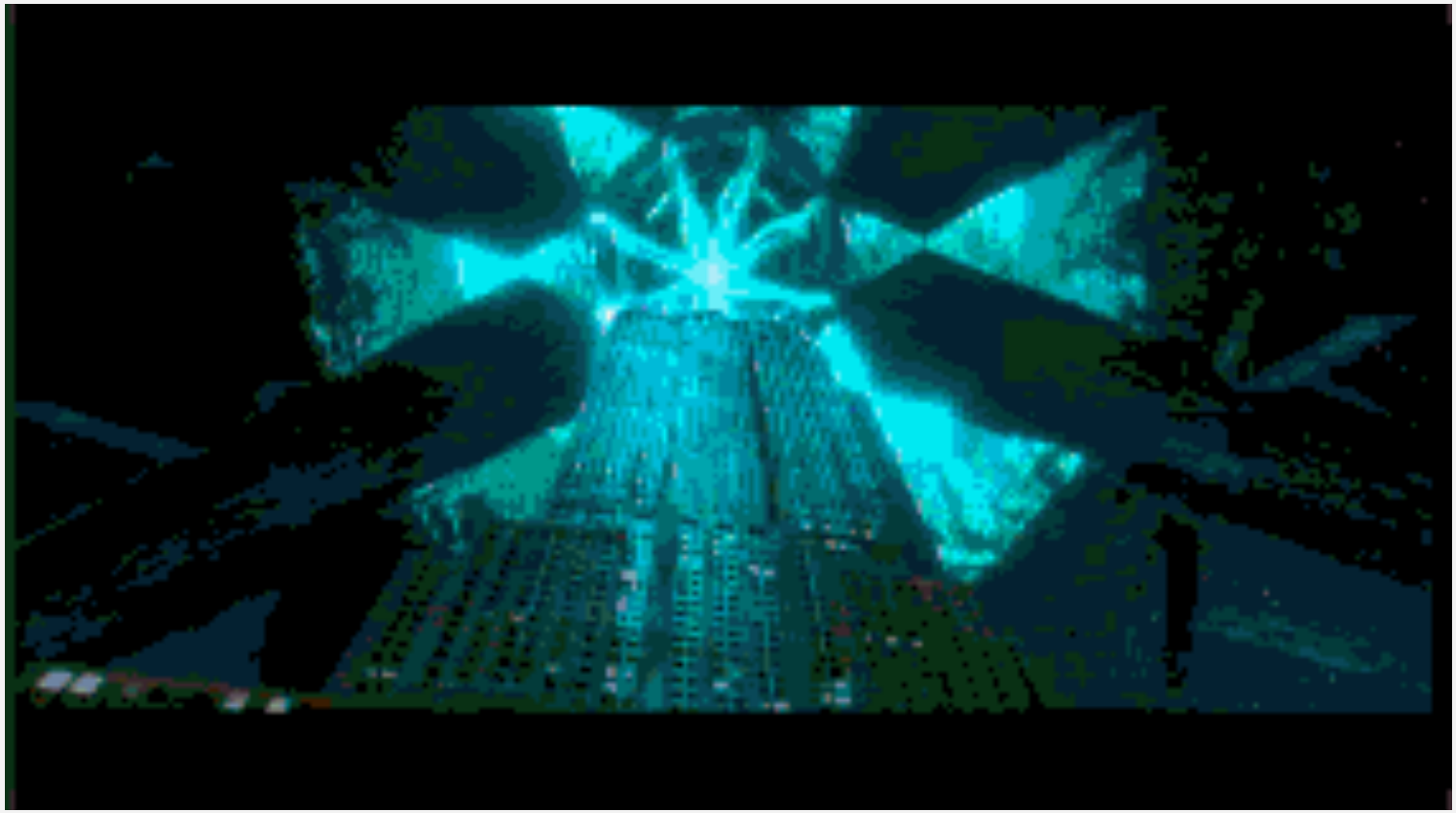
Exkurs: SGI





- komplexe interaktive Visualisierungen
- komplexe
  - 3D-Konstruktionen,
  - Animationen mit Previews in Echtzeit-3D und
  - Renderings

Exkurs: SGI



„Independence Day“

Exkurs: SGI



320 Visual Workstation

- Single Pentium III
- 550 MHz
- 128 MB
- 10.1 GB
- DM 6.850,–
- schwächer als Onyx2

Exkurs: SGI



Origin 3800 Hochleistungsserver

Exkurs: SGI

- 2003:
  - die Origin-Linie wird von der Altix-Linie abgelöst
  - Itanium-Prozessoren von Intel
  - Angepasstes Linux statt IRIX: Altix 4700

Exkurs: SGI



Altix 4700 im LRZ

Exkurs: SGI

- 2004:
  - Verkauf von Alias mit 3D-Software Maya
- 2005:
  - sinkende Umsätze und Gewinne
  - Aktie von der Börse genommen

Exkurs: SGI

- 2006:
  - Mai: Gläubigerschutz
  - Oktober: von Gläubigerschutz befreit: Konzentration auf Server, die unter Linux betrieben werden

Exkurs: Apollo, HP



- Apollo
- Gründung 1980
- einer der ersten Hersteller
- 1981: DN100
  - Aegis
  - 680x0

Exkurs: Apollo, HP

- 1984 DSP160
  - keine 680x0-Mikroprozessoren, sondern 32-Bit Bit-Slice CPU
- 1988 DSP10000
  - mit RISC-Architektur
  - teilweise Vorreiter für PA-RISC von HP

Exkurs: Apollo, HP

- 1980-1987: Sehr bedeutender Hersteller vernetzter Workstations
- Ende der 80er: Finanzielle Probleme
  - Apollo wird für 475 Millionen USD von Hewlett-Packard übernommen
  - Zusammenführung beider Techniken

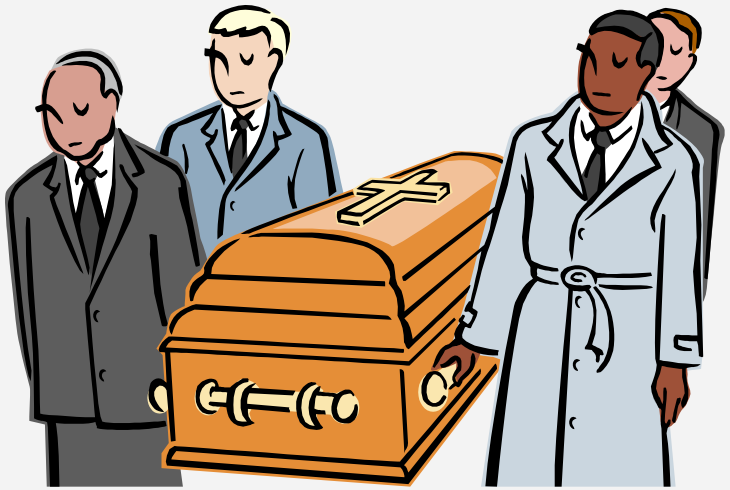
Exkurs: Apollo, HP

- Apollo
  HP 9000 Serie 700
- erfolgreichster Vertreter der Serie



Exkurs: Apollo, HP

- Preis-Leistungsverhältnis: PC der Workstation überlegen
- Ende der Workstations



Apple I

- Erstmals Tastatur, anstelle der Schalter wie bei Altair
- Fernseher wird als Video-Terminal genommen
- Der gesamter Rechner befindet sich auf einer Platine
- Da es keine billige RAMs verfügbar sind, wird ein Shift-Register benutzt um den Text auf dem Bildschirm darzustellen
- 8 KByte DRAM, 4 KB für BASIC, 4 KB stehen dem User zur verfügung
- Da es BASIC Programme laufenlassen kann, sorgt der Apple 1 für viel Aufregung und wird über 220 mal verkauft

Apple I



Apple II

- Im Jahre 1977 erscheint die nächste Generation, Apple 2
- Farb- und Soundunterstützung wurden eingebaut
- Per Slots erweiterbar
- Auch Speicher auf 48 KB zu erweitern wird möglich
- Programmer's Aid #1

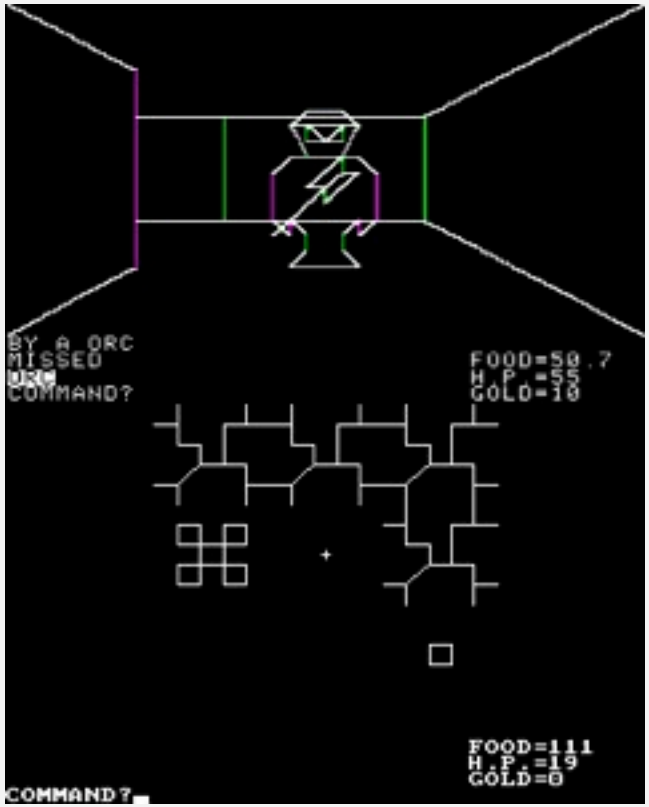
Apple II





Apple II

Akalabeth



Ultima II



C64



- Bis heute der bestverkaufter Heimcomputer mit über 17 Millionen Geräten
- Äußerlich an VIC20 angelehnt
- Auflösung 320x200
- 16 Farben
- 64 KByte RAM
- SID (Sound Synthesizer Chip) ist ein absolutes Highlight
- 1986 C64C (in Deutschland C64-II)
- August 1987 C64G
- Die Unterschiede zum ersten Modell sind minimal

C64



Commodore VIC20/VC20 Spiele

Pacman



Pharaohs Curse



Amiga 1000

- 1982 beginnt eine kleine Firma Amiga Inc. auf Basis von 68000 eine Videospielkonsole zu entwickeln
- Amiga kann 1984 keine neue Investoren finden und steckt in finanziellen Schwierigkeiten
- Commodore kauft dem ehem. Chef Tramiel die Amiga vor der Nase weg
- Der neuer Rechner AMIGA 1000 wird am 23.7.85 vorgestellt



C64



- Mit C64 landet Commodore einen Hit; in nur zwei Jahren verkauft sich der Computer über 4 Millionen mal weltweit
- Bis heute der bestverkaufter Heimcomputer mit über 17 Millionen Geräten
- Äußerlich an VIC20 angelehnt
- Auflösung 320x200
- 16 Farben
- 64 KByte RAM
- SID (Sound Synthesizer Chip) ist ein absolutes Highlight
- 1986 C64C (in Deutschland C64-II)
- August 1987 C64G
- Die Unterschiede zum ersten Modell sind minimal

ZX80



ZX80



- ZX80 ist für Personen die Programmieren lernen wollen
- Es wird sofort ein Erfolg
- Vermarktung in den USA
- Bis September 1980 über 20.000 Geräte verkauft
- 16 KByte Erweiterung

ZX81

- Im März 1981 vorgestellt
- Neue Chiptechnologie ermöglicht Sinclair Preis auf £69,95 bzw. £49,95 als Bausatz zu senken
- Besitzt Gleitkommafunktionalität, was jedoch Simuliert wird
- ZX81 hat verbesserte Videodarstellung
- Bis Januar 1983 werden 300.000 Stück weltweit verkauft
- Mitte 1982 darf die Firma Times Clone für amerikanischen Markt produzieren
- Dies führt zu Produktion von hunderten Programmen, Bücher, Speichererweiterung etc. für den Sinclair

ZX81



Spectrum

- Spectrum erscheint in zwei Varianten: 16KByte RAM für £125 und 48KByte RAM für £175
- 8 Farben in zwei Helligkeitstufen
- 1 Kanal-Soundgenerator
- 156 × 192 Auflösung
- Verbessertes BASIC
- Spectrum kann mit allen anderen Homecomputer auf dem Markt konkurrieren
- Bis Februar 1983 werden 200.000 Stück verkauft; nach Ostern sind es 12–15.000 pro Woche allein in England
- 1984 erscheint Spectrum+ mit verbesserter Tastatur
- 1985 Spectrum 128 mit 128 KByte Speicher und Dreikanal-Soundchip

Spectrum

Spectrum

Spectrum+





Atari: Die Modelle: 400 und 800

- 1978 bring Atari gleich zwei neue Computer heraus: Atari 400 und 800
- Basiert auf den Mikroprozessor 6502 mit 1,8 MHz
- 400 mit 16 KByte und Folientastatur
- 800 mit 48 KByte und Schrebmaschinentastatur
- Beide besitzen 10 KByte ROM
- 16 Farben in 16 Helligkeitsstufen
- 40 Spalten x 24 Zeilen in 320x192 Pixeln
- Und (!) 4-Kanal-Sound

Modelle 400 und 800

Hier die englische Werbung aus dem Jahre 1983.



Die Atari-XL-Serie

- Im Jahr 1983 bringt Atari drei neue Computer heraus: 1200XL, 600XL und 800XL
- Die 600XL und 800XL sind aufgrund ihrer Kompatibilität zu 400 und 800 erfolgreicher
- 600XL hat 16 KByte RAM (auf 64 erweiterbar)
- 800XL hat 64 KByte
- Der Unterschied zur Vorgänger liegt in verbesserten Betriebssystem

Die XL-Serie

Screenshots aus den Spielen Jumpman und The Goonie.



Geschichte





- 1983 J. Lanier (Atari Research Center) entwickelt Datenhandschuh
- 1984 Globale Beleuchtungs-Simulation mit Radiosity (Goral, Nishita)
- 1984 Gründung Wavefront Technologies für Animations-Software

Geschichte

- 1985 Gründung von VLP durch J. Lanier, zur Entwicklung erster kommerzieller Virtual-Reality-Produkte
- 1985 GKS – erster grafischer ISO-Standard (1988 GKS-3D)
- 1986 Gründung von Pixar durch Ed Catmull und A. R. Smith nach Abspaltung von Lucas Film

Geschichte

- 1986 „Luxo Jr.“ von Pixar: Erster Computergrafik-Kurzfilm, der für einen OSCAR nominiert wird
- 1986 Pixars RenderMan wird Industrie-Standard
- 1987 „Rendez-vous à Montréal“ Marylin Monroe und Humphrey Bogart werden digital wiederbelebt





Geschichte

- 1986 „Flight of the Navigator“ von Randal Kleiser, erstmalig Nutzung des Reflection Mappings
- 1988 „Tin Toy“ von Pixar – erster Computergrafik-Film, der einen Oscar gewinnt, Kategorie: Bester animierter Kurzfilm





Geschichte

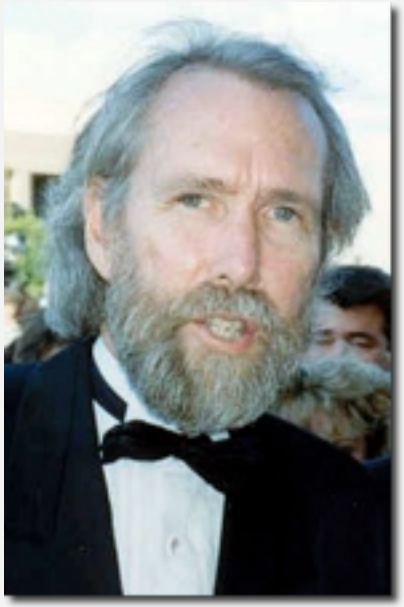
- 1989 „The Abyss“ von James Cameron
- ILM stellt dabei die Szene mit der Wasser-Kreatur her, welche die Gesichter der Mannschaft imitiert
- 1989 „Indiana Jones und der letzte Kreuzzug (Indiana Jones and the Last Crusade)“:
- Erster Kinofilm, in welchem Compositing vollständig digital ist

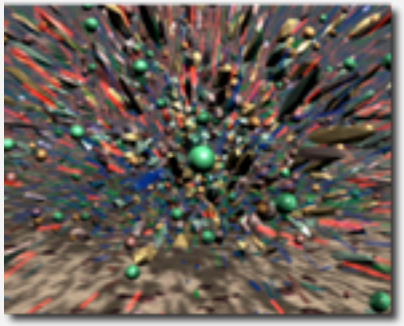




Geschichte

- 1989 Einführung von Motion Capturing durch Jim Henson, Nutzung mechanischer Eingabegeräte für Computeranimation
- 1991 „Panspermia“ von Karl Sims: Bahnbrechende prozedurale Animationen; Wälder und Partikelsysteme





Geschichte

- 1991 „Terminator 2“, computergenerierte Spezialeffekte bei Animation des „T1000“
- 1992 „Death Becomes Her (Der Tod steht ihr gut)“: Fotorealistischer Austausch von Haut und Körpern





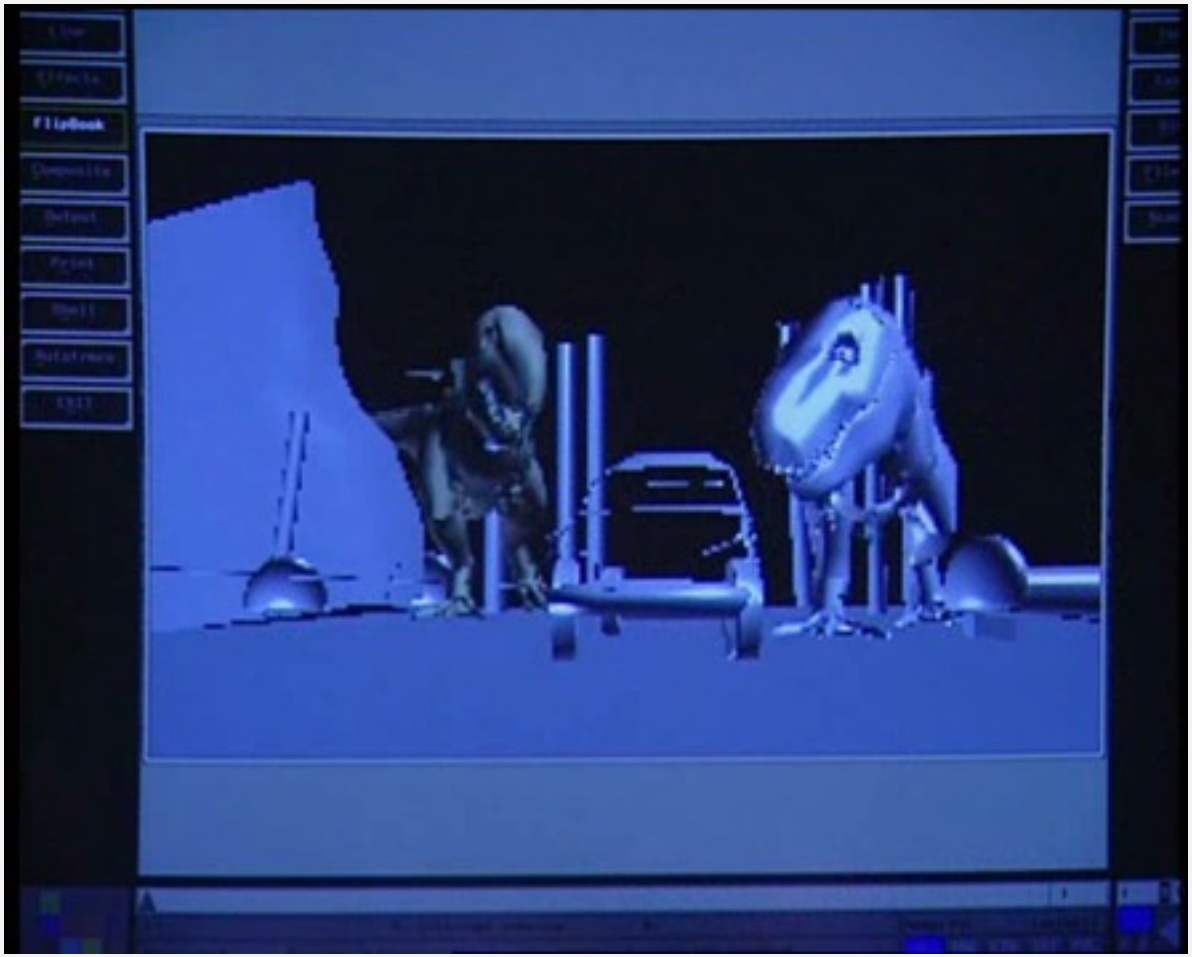
Geschichte





- 1993 „Jurassic Park“
  - von Steven Spielberg
  - anstelle der ursprünglich geplanten Animatronik werden Computeranimationen für die Dinosaurierszenen eingesetzt

Geschichte



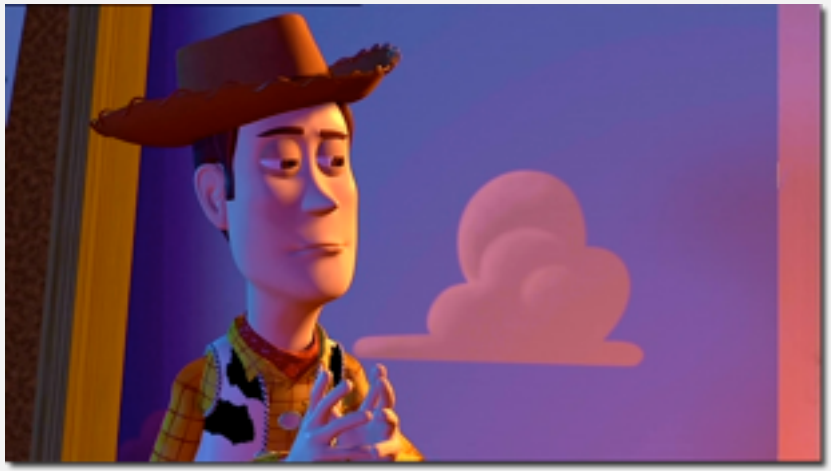
Geschichte: Toy Story

- 1995: „Toy Story“, Pixar
  - erster vollständig computeranimierter Film
  - Rendering: 800.000 Stunden Berechnungszeit für 70 Minuten Film auf 177 Sun Sparc 20





Geschichte: Toy Story





- Problem: Überzeugende Ausleuchtung
- Lösung: Gestalten mit Licht
- Zahlreiche Herausforderungen, wie z. B. überzeugende Pflanzen

Geschichte: Toy Story

- „Toy Story“ ist 79 Minuten lang,
- besteht aus 114.240 gerenderten Frames (Einzel-Bildern),
- jeder Frame hat eine Auflösung von 1.526 × 922 Pixeln,
- insgesamt besteht der Film aus 1.635 Kamera-Shots,
- es wurden mehr als 400 Modelle für Charaktere, Häuser, Autos, etc. erstellt,
- das fertige Werk verfügt über 160 Mrd. Pixel,
- das macht über 600 Mrd. Bytes;

Geschichte: Toy Story

- insgesamt wurden 34 Terabytes Renderman-Files (Modelldaten) zum Rendern benötigt,
- 4,5 Millionen Zeilen Code beschreiben die Szenenobjekte,
- 25.000 Storyboard-Zeichnungen bildeten die Grundlage für den Film,
- 300 vernetzte Sun Workstations rechneten 24 Stunden am Tag insgesamt 800.000 Maschinenstunden,
- 110 Leute bei PIXAR arbeiteten an dem Film, davon 28 Animateure.

Geschichte





- 1996:
  „Dragonheart“ (Universal):
  - bahnbrechende Computergrafik-Character-Animation
  - Lippen-Synchronisation
  - Grimassieren analog zum Synchronschauspieler Sean Connery

Geschichte

- 1995 „Jumanji“ (Tri-Star): Weiterentwicklung im Bereich Haarsimulation, Integration Realfilm/Computergrafik
- 1996 „Twister“: Wetter-Effekte und Tornados





Geschichte





- 1997 „Titanic“:
  - Einsatz von Motion-Capturing für Menschenmengen-Simulationen
  - Einsatz einer so genannten Rendering-Farm, erstmalig auf PC-Basis unter Linux

Geschichte

- 1997 „Geri's Game“:
  - Erste Anwendung von Subdivision-Surface-Technologie
  - Oscar für den besten animierten Kurzfilm 1997



Geschichte

- 1998 „Antz“ (PDI): Software zur Facial Animation, Crowd-Simulation, Wasser-Effekte; Abstimmung der digitalen Charaktere mit Synchronschauspielern
- 1999 „Das große Krabbeln (A Bug’s Life)“: Prozedural erzeugte Pflanzen, Windsimulation





Geschichte

- 2001 aufwändige und sehr überzeugende Gesichtsanimationen von virtuellen Charakteren im Film „Shrek“





Geschichte

- 2001 „Final Fantasy – The Spirit Within“
  - erster komplett computeranimierter Spielfilm mit realistischen virtuellen Charakteren
  - Produktion des Films dauerte ca. 4 Jahre
  - etwa 170 Computeranimatoren
- 2005 „Final Fantasy – Advent Children“





Quellen

- John Dingliana: *Ray-tracing*. University of Dublin, Department of Computer Science, Power-Point-Präsentation, Dezember 2006. https://www.cs.tcd.ie/John.Dingliana/Lab4.ppt.
- Foley, van Dam, Feiner, Hughes: *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Heidrun Schumann: *Computergraphik 1*. Universität Rostock, Vorlesungsskript, Juli 2005.
- *ACM SIGGRAPH Education Commitee*. WWW-Präsentation, Dezember 2006. http://education.siggraph.org/.
- Nadia Magnenat-Thalmann und Daniel Thalmann: *Handbook of Virtual Humans*. Wiley & Sons, 2004. ISBN: 0470023163.
- *The Internet Movie Database*. WWW-Präsentation, Dezember 2006. http://www.imdb.com.
- *Wikipedia-Lexikon*. WWW-Präsentation, Oktober 2006. http://de.wikipedia.org/.
- Swen Jurić: *Workstations.* MEDIADESIGN-Hochschule, Power-Point-Präsentation, März 2007.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik und –animation

Peripheriegeräte





Peripheriegeräte

Der PC als grafisches System

Graphisches System

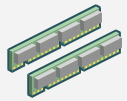


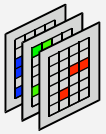
Eingabegerät



Prozessor

Hauptspeicher



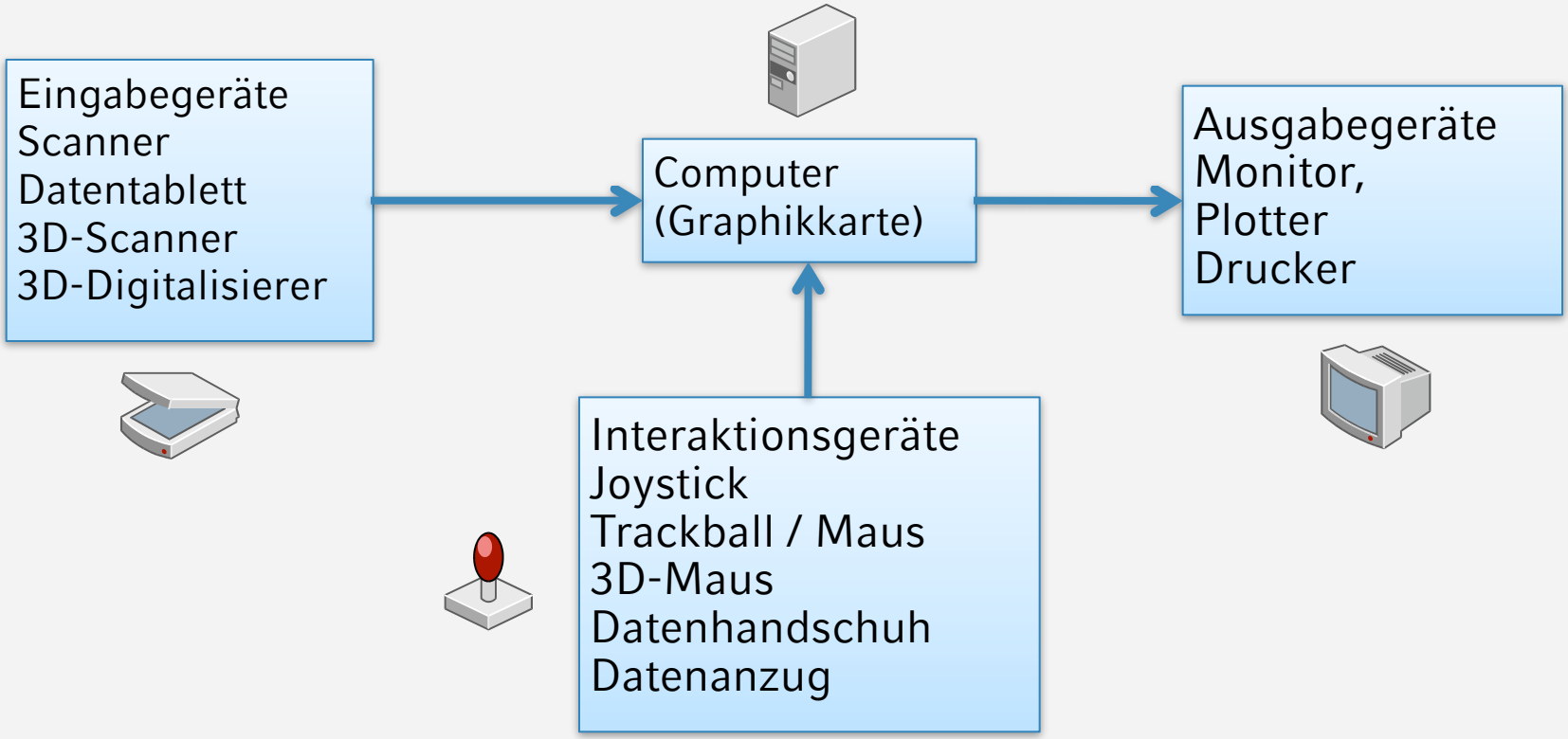


Framebuffer



Ausgabegerät

Graphik-Hardware (Peripheriegeräte)





Peripheriegeräte

Eingabegeräte für die Interaktion

Eingabegeräte für die Interaktion

- Geräte, um einem Computer Informationen zukommen zu lassen
- Jedes Gerät, welches einen Computer mit Daten versorgt, zählt als Eingabegerät
- Lochkarten und Schalter waren die erste bekannte Eingabemöglichkeit für den Computer

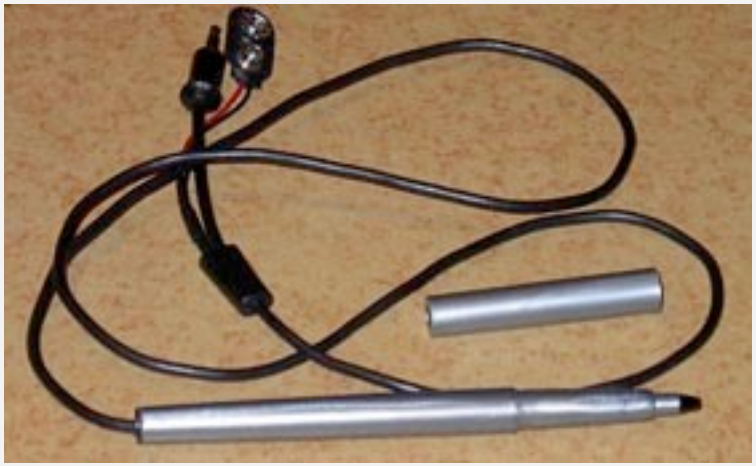
Klassifikation

- Einteilung nach Interaktionsaufgaben:
  - alphanumerische Eingabegeräte
  - Positionsgeber
  - Wertgeber
  - Auswähler
  - 3D-Eingabegeräte
  - andere (Spracherkennung etc.)

Positionsgeber

- Verarbeitung der Koordinaten
  - absolut
  - relativ
- Art und Weise der Interaktion
  - direkt
  - indirekt
- Signalübertragung
  - diskret
  - kontinuierlich

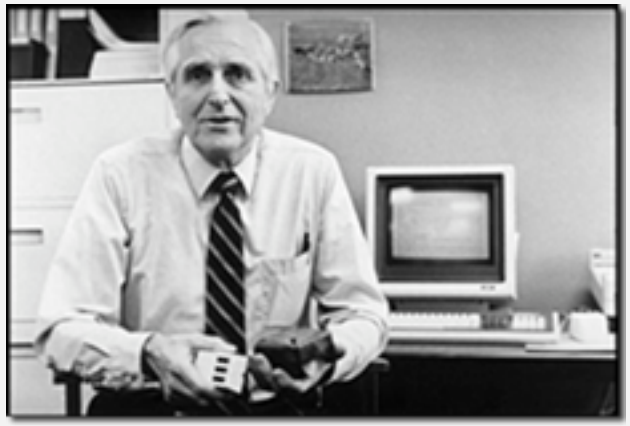
Lichtgriffel



- Computer-Eingabegerät zum direkten Arbeiten auf einem Röhrenbildschirm
- Wurde 1949 am Lincoln Laboratory entwickelt

Maus

- 1963 erfand Douglas C. Engelbart die erste Computermaus
- Anregungen durch das Militär
- Fand erst nach Patentierung 1970 Beachtung





Maus

- Ursprünglicher Name war „X-Y-Positions-Anzeiger für ein Bildschirmsystem“
- 1970 wird die Maus im PARC weiterentwickelt
- 1971 English ehemaliger Mitarbeiter Engelbarts erfindet erste Kugelmaus
- Wird zum ersten mal im Xerox Alto eingesetzt

Maus

- Erste Maus für den Xerox Alto
- Entwicklung der ersten GUIs (GUI = engl. Graphical User Interface)
- Grundlage für grafische Benutzungsoberflächen und Interaktionen mit dem Rechner bis heute

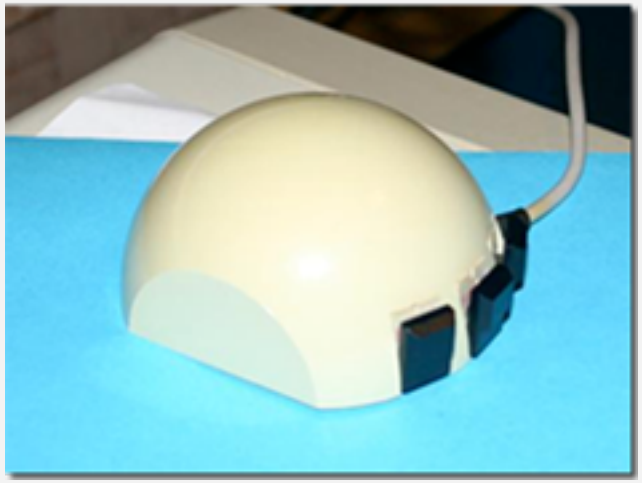




Maus

- auch am Forschungszentrum von Xerox in Palo Alto/ Kalifornien (PARC) entwickelt
- Maus für den Lilith Computer, der 1980 von Niklaus Wirth gebaut wurde





Maus

- Ab 1981 sollte die Maus kommerzieller und gezielter im Xerox „Star“ eingesetzte werden
- Apple lässt sich nach dem Scheitern des „Star“ die Technik für den Apple „Lisa“ lizenzieren



Maus

- 1983 Apple
- Maussteuerung für den „Lisa“-Computer
- beispielgebend für Benutzer-Interaktion





Maus



- Mit der Einführung der PS/2 Systeme durch IBM im Jahre 1987 werden Mäuse mit PS/2-Anschluss vorgestellt
- Maus wurde von IBM immer als Pointing Device bezeichnet

Maus

- 1980 Beginn der Entwicklung optischer Mäuse
- Steven Kirsch bei Mouse Systems und Richard Francis Lyon bei Xerox entwickelten unterschiedliche Ansätze
- 1990 Optische Mäuse durch Einführung neuer Bildverarbeitungschips

Maus

- Bereits 1984 stellt Logitech eine drahtlose Kugelmaus vor die auf Infrarottechnologie basiert – zu diesem Zeitpunkt ist die Technik noch nicht ausgereift, so dass ein Einsatz nicht möglich ist
- 1991 kabellose Mäuse über Radiowellen
- Ende 2002 wird von Microsoft und Logitech eine Maus vorgestellt, die über Bluetooth per HID-Profil mit dem angeschlossenen PC kommuniziert
- 2003 Verschlüsselung

Trackball

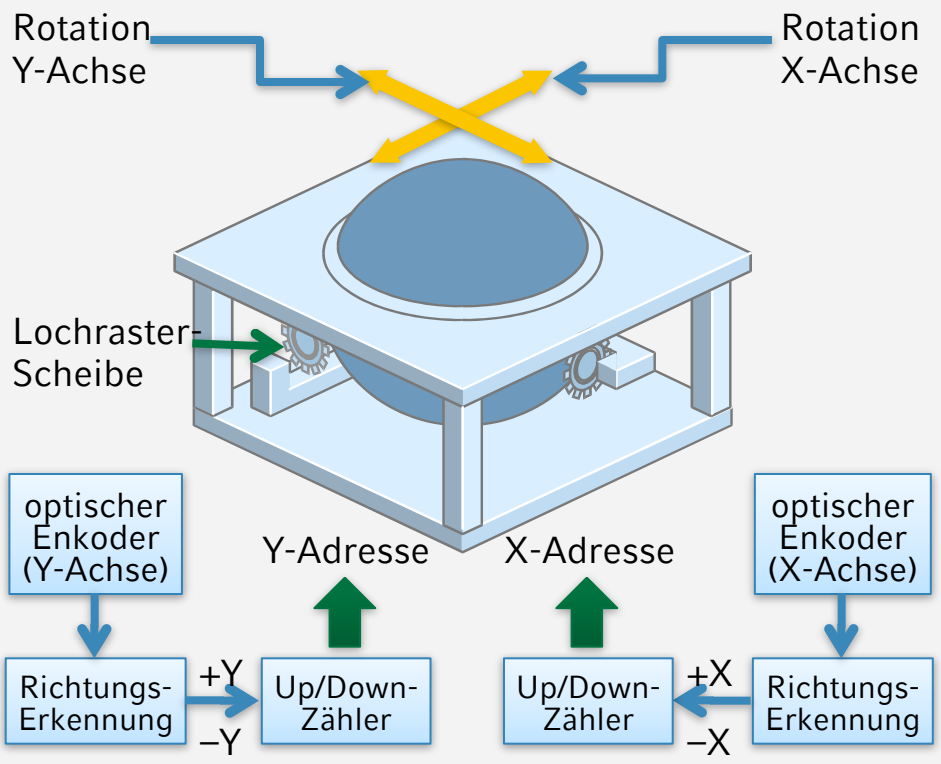


- Ein Trackball (deutsch Rollkugel) ist ein Eingabegerät, das aus einem Gehäuse besteht, in dem eine Kugel angebracht ist
- Kugel wird mit den Fingern bewegt
- im Gehäuse erfassen mechanische oder optische Sensoren die Bewegung der Kugel entlang zweier Achsen.
- Trackball funktioniert ähnlich wie eine umgekehrte Maus

Trackball / Maus (mechanisch)

Funktionsprinzip:

Abhängig von der Rotations-Richtung treibt ein Gummiball zwei Lochraster-Scheiben an, die im Winkel von 90° versetzt angeordnet sind



Maus (optisch)

- Bewegliche Teile (Kugel etc.) werden durch einen optischen Sensor (CMOS) und einen Digitalen Signalprozessor (DSP) ersetzt
- Infrarot-Sensor emittiert rotes Licht unter die Maus und nimmt 1.500 mal pro Sekunde ein hochaufgelöstes Bild auf.
- Bilder dann durch den DSP verglichen und in Bewegungen des Cursors umgesetzt
- Image Correlation Processing mit 18 MIPS
- Vorteile
  - sanftere, genauere Bewegung
  - weniger anfällig gegenüber Verschmutzung
  - funktioniert auf vielen Oberflächen

Joystick



- Der Joystick ist dem Steuerknüppel eines Flugzeugs oder dem Gangschaltungs-Hebel eines Autos nachempfunden
- dient zur Bewegung von Steuerelementen

Touchpad



- Der Begriff Touchpad bzw. Tastenfeld bezeichnet eine berührungs-empfindliche Fläche, die als Maus- und Tastenersatz in Notebooks meistens unterhalb der Tastatur angebracht ist.

Grafiktablett

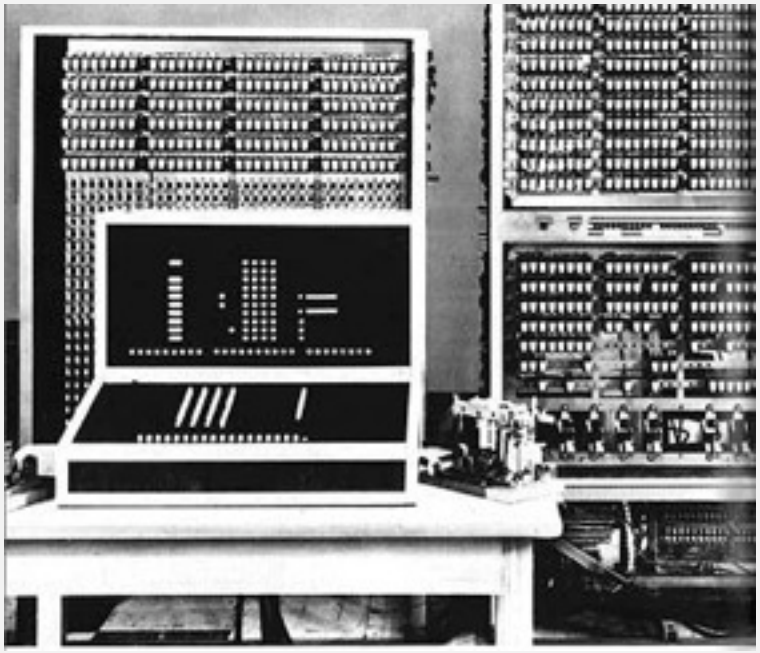


- Ein Grafiktablett ist ein Zeigegerät für Computereingaben.
- Die Spitze eines Stiftes wird auf einer Platte bewegt.
- Der Stift sendet Daten über Stiftdruck und gedrückte Tasten an das Tablett, die so gewonnenen Bewegungsdaten werden vom Grafiktablett an den Computer übermittelt.

Künftiges

- Es gibt noch einige Neuerungen in diesem Bereich, die teilweise noch mit gesunder Skepsis genutzt werden und wieder andere die jetzt schon im Sprung zu Kommerziellen erfolgen sind.
- Ein paar dieser Neuheiten sind die Lasertastatur, LCD-Tastatur oder vielen bekannt die Wii- Controller die jetzt auch für den PC in Serie gehen.

Tastatur



- Die erste Tastatur die an einem Computer Verwendung fand: 1944 in der Z3 von Conrad Zuse
- sie löste zudem das veraltete Lochkartensystem als Eingabemöglichkeit ab

Tastatur

- Als späteres Grundgerüst wurden Fernschreiber mit Schreibmaschinentastatur zur Eingabe verwendet. Diese wurden aber recht schnell durch rein Elektronische Tasten ersetzt
- Arten:
  - Dvoràk
  - alphabetisch
  - QWERTZ-Tastaturen

Tastatur

- Spezialtatsturen
- Beispiele:
  - Commodore Tastatur
  - Sinclair ZX Spectrum





Tastatur

- 83-Tasten-PC/XT-Tastatur (Merkmal: Funktionstasten F1 bis F10 links außen vertikal, gemeinsame Cursorsteuerung auf Zehnertastatur)
- 102 Tasten-MF2-Tastatur





Tastatur



- Apple Tastatur
- Spezialfunktionen für Hardware, etwa Einschalter

Tastatur

- Die Tastaturen neuerer Bauart sind wie andere Eingabegeräte auch kabellos und mit USB Anschluss erhältlich
- Konzentration auf Design und Funktionalität

Tastatur



- Die Lasertastatur funktioniert, indem der Sensor der das Bild projiziert, misst, wo das Laserfeld unterbrochen wird
- anschließend wird das entsprechende Zeichen oder die Aktion zugeordnet

Ausblick



- berührungs-empfindliche Monitore als Tastatur
- etwa „iPhone“

Scanner

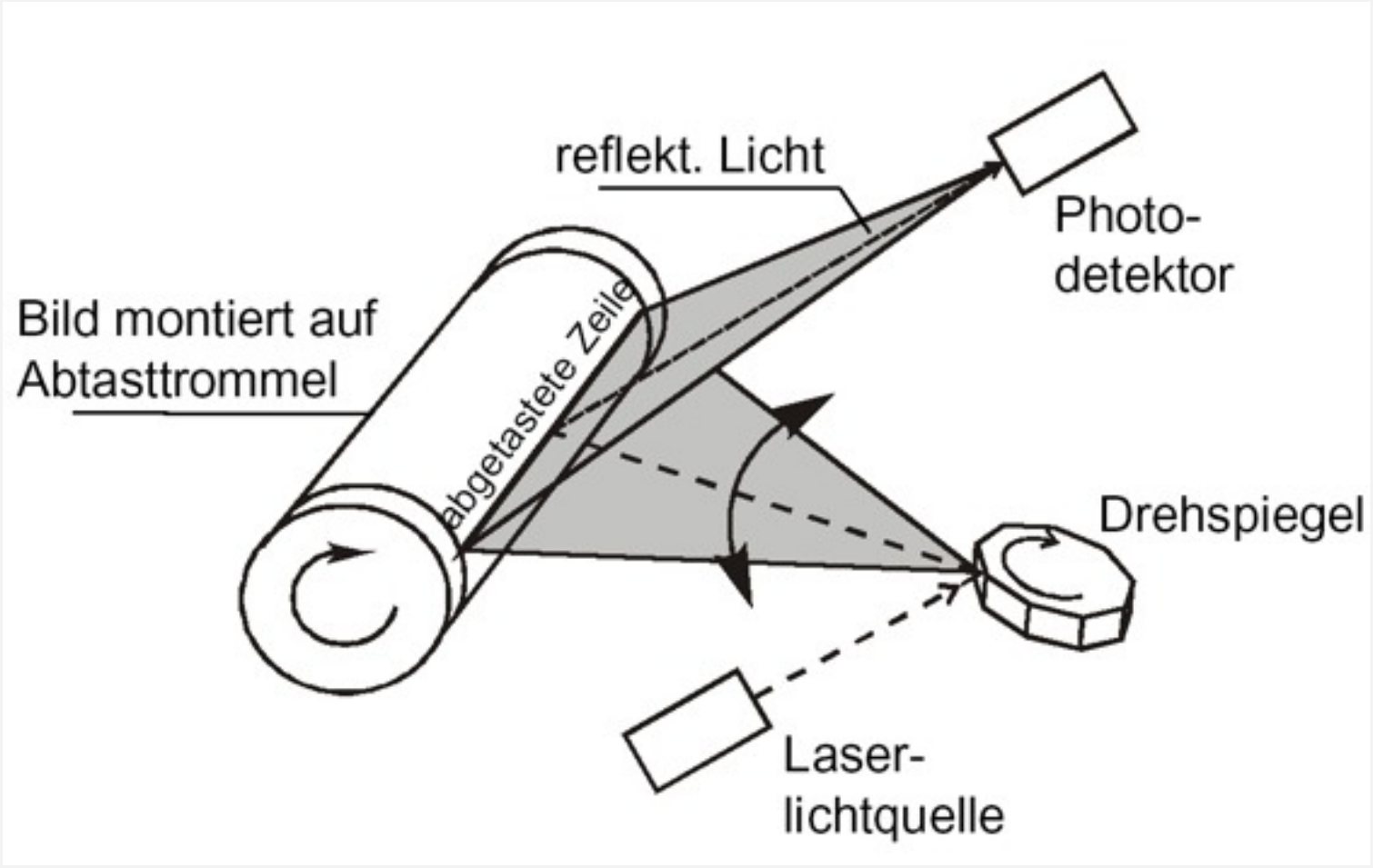


- im Scanner werden die analogen Daten der physikalischen Vorlage mit Hilfe von Sensoren aufgenommen und anschließend mit AD-Wandlern in digitale Form übersetzt
- So können sie z. B. mit Computern verarbeitet, analysiert bzw. visualisiert werden

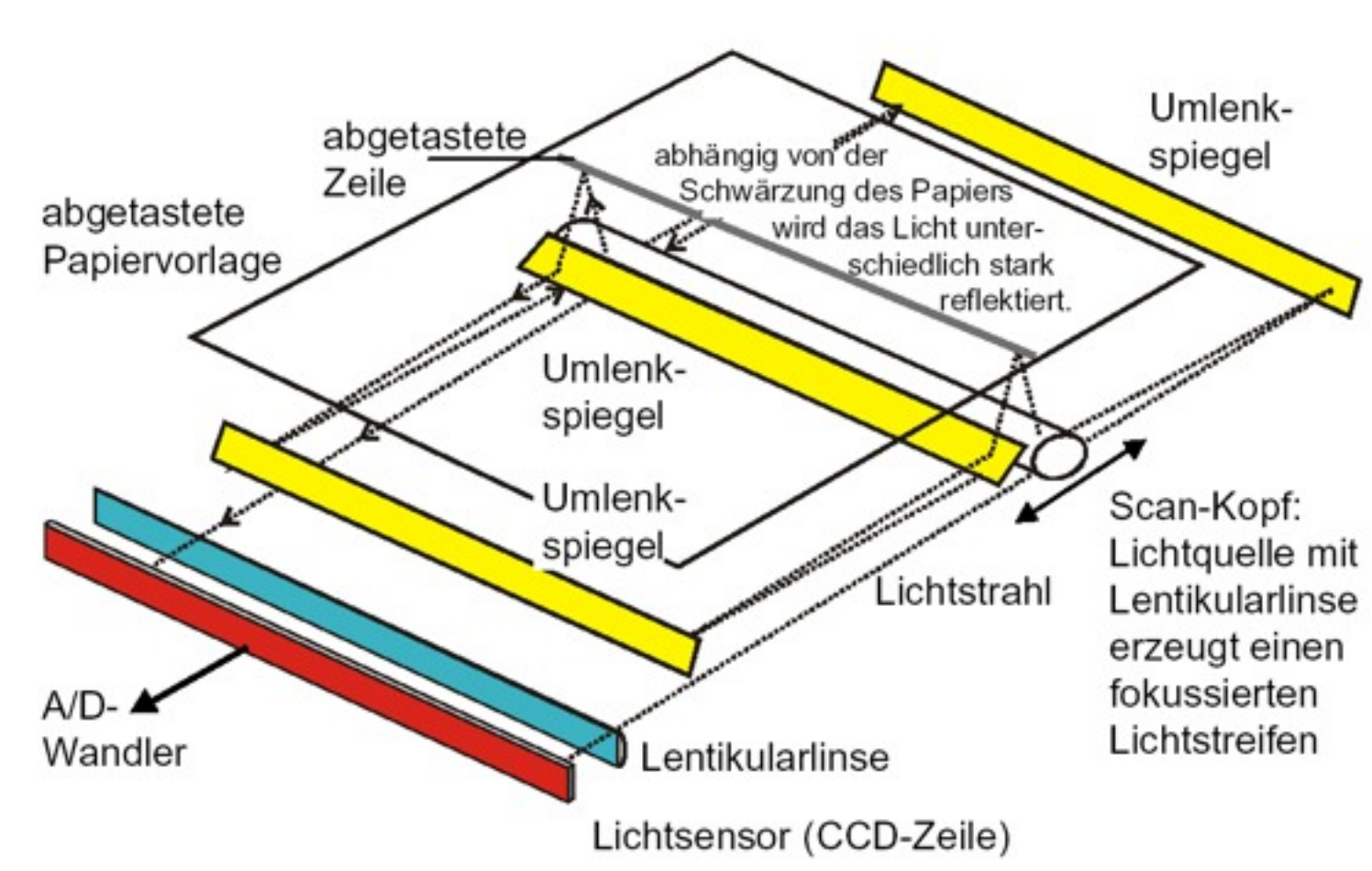
Scanner

- Eingabe von Bild-Daten
- dabei Analog-Digital-Wandlung
- Arten von Scannern:
  - Highend-Trommelscanner, auch für reflektierende Bilder und Folien, 35mm Dia bis bin zu 16 foot × 20 in Material, hohe Auflösung (≥ 10.000 dpi)
  - Dokumentscanner, für OCR und Document Management
  - Photoscanner (Photo wird über eine stationäre Lichtquelle bewegt)
  - Folien-/Diascanner mit Durchlicht
  - Handscanner

Trommelscanner



Flachbettscanner



Alternative Eingabegeräte



- Ähnlich wie der Wii-Controller werden in nächster Zeit alternative Eingabegeräte für interaktive virtuelle 3D-Welten den PC erobern.

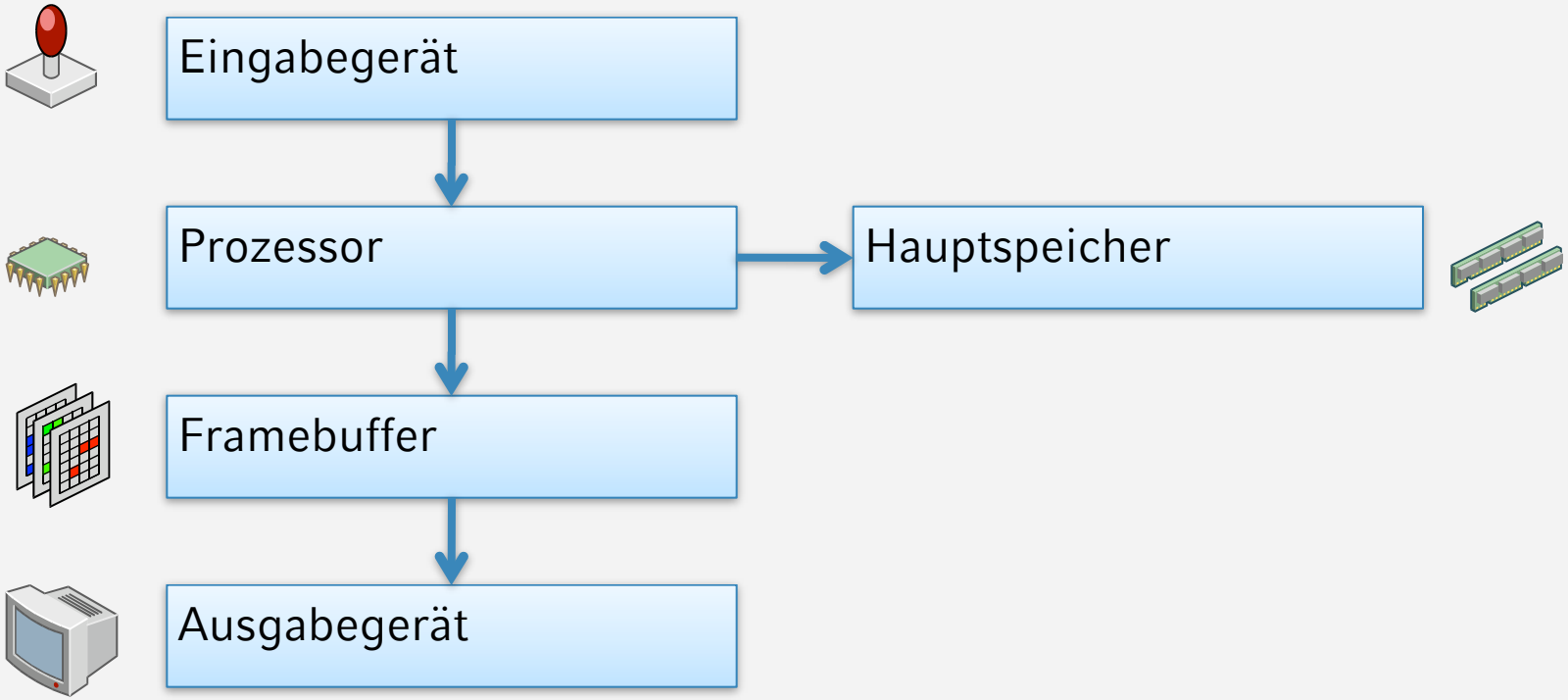
LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik und –animation

Peripheriegeräte für die Ausgabe



Graphisches System





Peripheriegeräte

Bildschirme



Bildschirme

- Bildschirme
  - CRT-Bildschirme
  - LCD-Bildschirme
  - Plasma-Bildschirme
  - LED-Anzeigen
  - OLED-Anzeigen

Monitore (Datensichtgeräte)

- Standard heute: Kathodenstrahlröhren-Monitore (CRT)
  - übliche Auflösungen 640 × 480 bis 1600 × 1200 Pixel
  - Monochrom-Monitore nur noch selten
  - üblich: 16 Bit oder 32 Bit Farbtiefe
- Laptops und auch Standalone: LCD-Displays
- Plasmadisplays für Spezialanwendungen und große Displayflächen

Röhrenbildschirm

- CRT, engl: Cathode Ray Tube
- 1897 von Karl Ferdinand Braun entwickelt, deshalb auch braunsche Röhre gennant
- Braunsche Röhre: Fernsehgeräte, Monitore und Oszilloskope

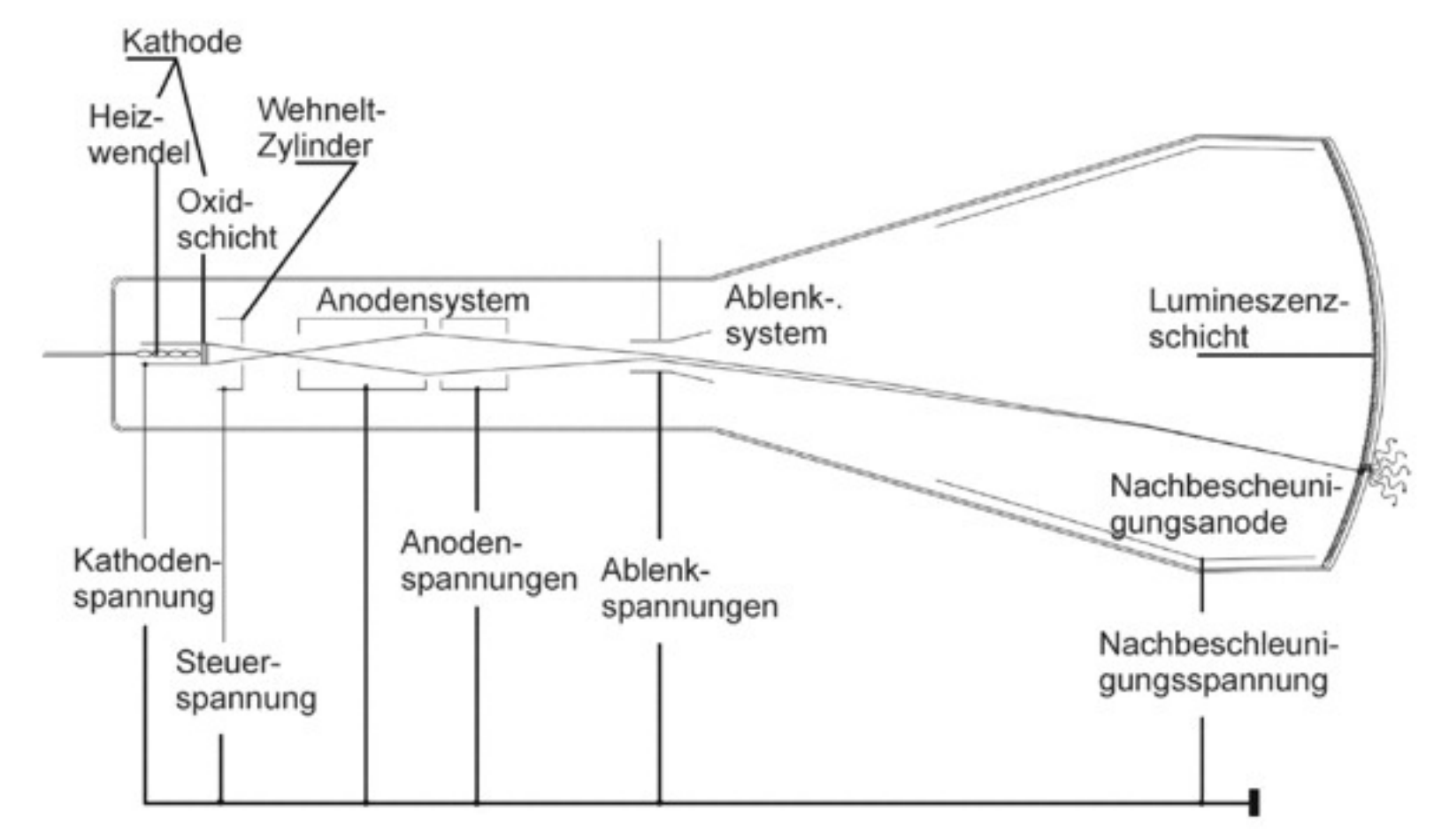




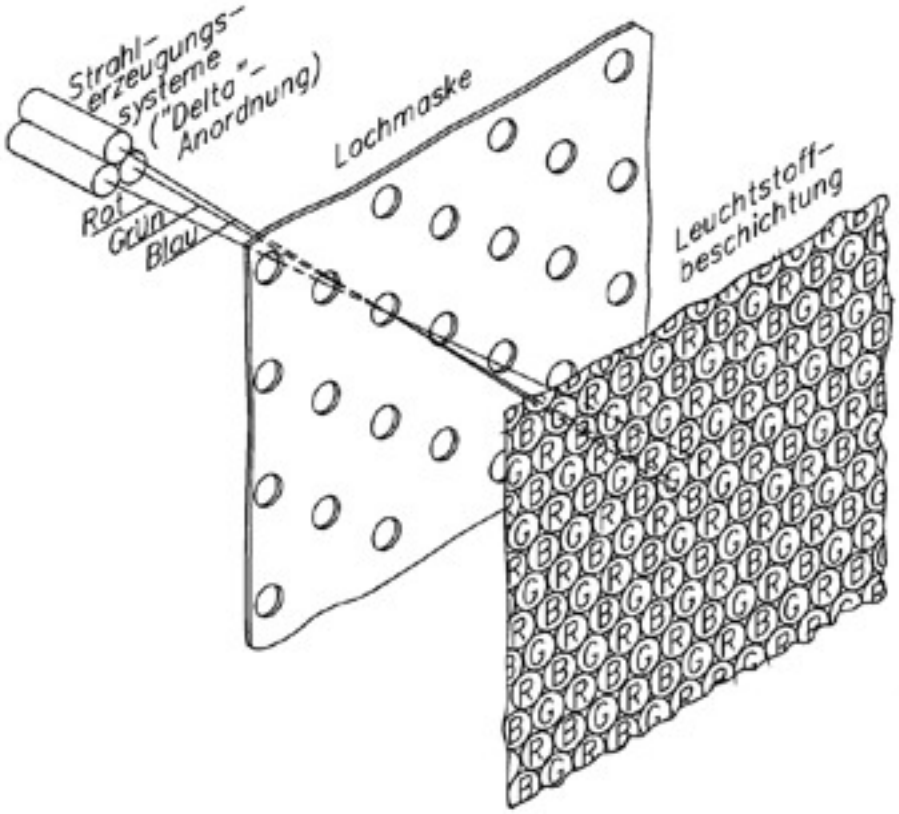
Fernsehen

- 1923: Zusammen mit der Kathodenstrahlröhre ist der Weg zum vollständig elektronischen Fernsehen geebnet; August Karolus stellt in Leipzig sein erstes Fernsehgerät vor
- 1927: Philo Farnsworth gelingt die Übertragung eines Bildes auf rein elektronischem Weg
- 1941 gelingt es John Logie Baird, erste farbige Fernsehbilder zu übertragen
- 1954: In Amerika wird die NTSC-Norm für Farbfernsehen eingeführt
- 1963: Am 1. April schlägt Walter Bruch das PAL-Verfahren für das Farbfernsehen vor

Aufbau Kathodenstrahlröhre

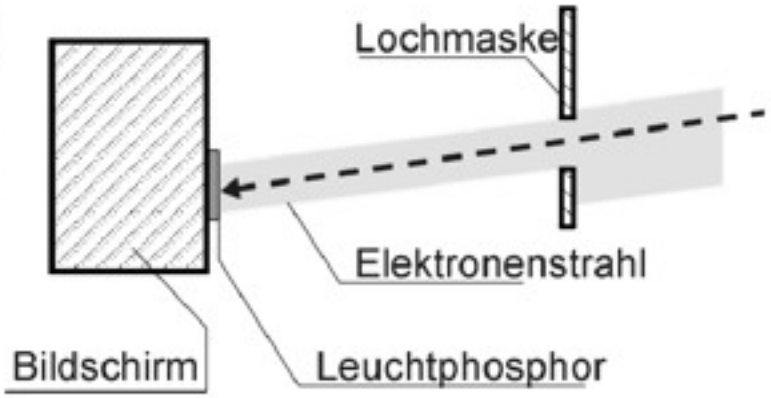


Delta-Bildröhre

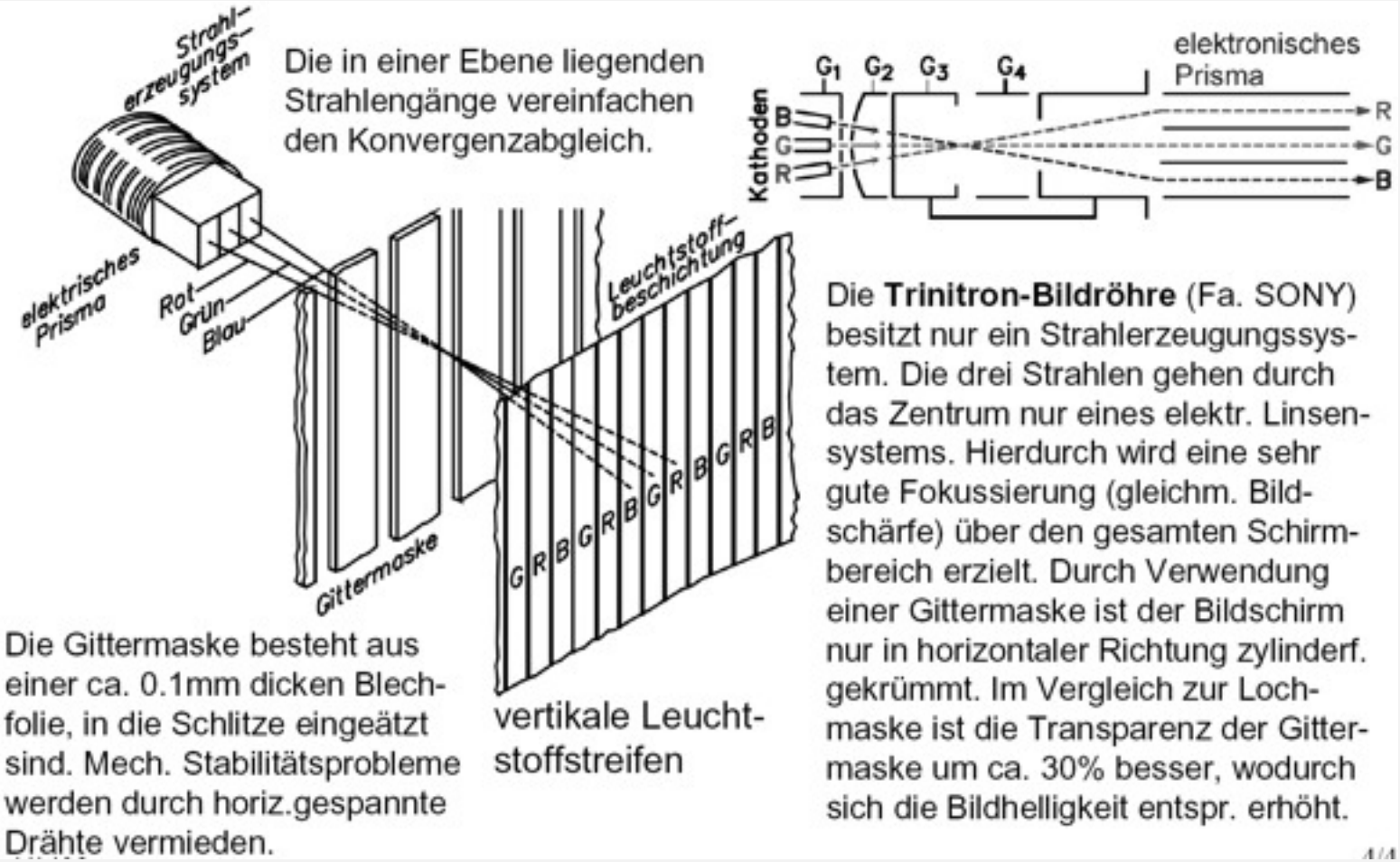


Wirkungsweise der Lochmaske

Die Auftreffgebiete der Elektronenstrahlen überdecken die Abblendlöcher der Maske. Nur die Anteile, die auf die korrespondierenden Leuchtphosphorpunkte auftreffen, werden durchgelassen.



Trinitron-Bildröhre



Vor- und Nachteile von CRT-Displays

Vorteile	Nachteile
hohe Auflösung,	schwer, sperrig,
volle Farbtüchtigkeit, sehr guter Kontrast,	hohe Leistungsaufnahme,
niedriger Preis bei hoher Zuverlässigkeit.	flimmernd, verzerrend,
	Analogtechnik,
	Röntgenstrahlung.

LCD-Bildschirme

- engl: Liquid Crystal Display (LCD)
- Flüssigkristallbildschirm
- Hintergrundbeleuchtung durch Leuchtstoffröhren (Lebensdauer 100.000 Stunden)
- oder seit neustem LEDs – dadurch weniger Energieverbrauch

LCD-Bildschirme

- Aktive-Matrix-Display
  - aktuelle Technik bei LCD-Bildschirmen
  - LCDs werden durch eine Matrix aus Dünnschichttransistoren angesteuert
  - engl: thin film transistor (TFT)
  - Diese Bauart von Displays ist als Matrix-LCDs bekannt, wird aber fälschlicherweise umgangssprachlich häufig auch als TFT-Display bezeichnet.

LCD-Bildschirme

- Vorteile LCD gegenüber CRT:
  - geringere Stromaufnahme
  - Strahlungsarm
  - flimmerfreies, verzerrungsfreies, scharfes Bild
  - geringes Gewicht
  - geringe Einbautiefe
  - gegenüber Magnetfeldern unempfindlich

LCD-Bildschirme

Nachteile LCD gegenüber CRT

- nur die native Auflösung ist wirklich scharf
- andere Auflösungen sind unscharf
- Darstellung von schwarz als dunkelgrau
- produktionsbedingte Pixelfehler

Nachteile Plasma gegenüber LCD

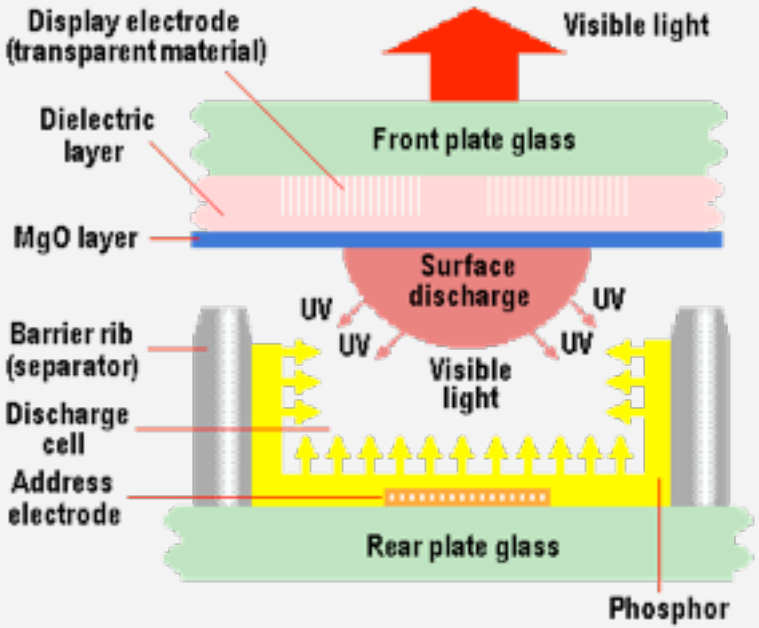
- Einbrenneffekte
- Transport nur stehend, da sonst Schwingungen zu Schäden führen können

LCD-Bildschirme

- Samsung LE-52F96BD:
  - Bildschirmdiagonale: 52 Zoll (132 cm)
  - Auflösung: 1.920 × 1.080 Pixel
  - Kontrast: 500.000:1
  - Helligkeit: 450 cd/m²
  - Energiebedarf: 310 Watt
  - Gewicht: 42 kg
  - Tiefe: 32 cm
  - Preis: 2669,00 €

(Quelle: www.hardwareschotte.de, Online: 8. Mai 2008)

Plasmadisplay



- Wirkprinzip: Anlegen einer Hochspannung an ein Gas mit niedrigem Druck erzeugt Licht
- Plasmadisplay: Matrix kleiner „Leuchtzellen“
- Anlegen einer Spannung: Ionisieren des Gases (Übergang zu Plasma)
- UV-Licht wird emittiert, bringt Phosphorschicht zum Leuchten
- Jede Zelle hat drei „Subzellen“ (RGB)

Plasmabildschirme

Vorteile Plasma gegenüber LCD

- hoher Kontrast (15.000 : 1)
- schnelle Ansteuerung der Bildpunkte
- beliebige Größe (theoretisch)
- bessere Darstellung von Schwarz

Nachteile Plasma gegenüber LCD

- Einbrenneffekte
- Transport nur stehend, da sonst Schwingungen zu Schäden führen können

Plasmabildschirme

- Samsung PS-50Q91H:
  - Bildschirmdiagonale: 50 Zoll (127 cm)
  - Auflösung: 1.365 × 768 Pixel
  - Kontrast: 15000:1
  - Helligkeit: 1300 cd/m²
  - Energiebedarf: 460 Watt
  - Gewicht: 69 kg
  - Tiefe: 8,9 cm
  - Preis: 1089,00 €

(Quelle: www.hardwareschotte.de, Online: 8. Mai 2008)

LED

- engl. Light-Emitting Diode (LED)
- Henry Joseph Round, 1907
- LEDs in den Farben RGB
- Verwendung bei sehr großen Anzeigetafeln im Außenbereich

LED





OLED

- engl. Organic Ligth-Emitting Diode (OLED)
- bestehend aus mehrere Schichten
- Kathode injiziert Elektronen, Anode stellt Löcher bereit
- In der Rekombinationsschicht werden Photonen freigesetzt
- Farbe hängt vom Energieabstand zwischen angeregtem und Grundzustand ab
- Farbe kann durch Farbstoffmoleküle gezielt geändert werden

OLED

Vorteile

- günstige Produktionskosten durch Drucken
- Blickwinkel bis zu 170°
- hohe Schaltgeschwindigkeit (0,001 Sekunden)

Nachteile

- noch zu kurze Lebensdauer, nur 150.000 Stunden bei blauen OLEDs – also etwa 17 Jahre
- rote, blaue und grüne OLEDs altern unterschiedlich schnell
- hohe Temperaturen verkürzen Lebensdauer



Peripheriegeräte

Projektoren

CRT-Projektoren

- Projektoren
  - CRT-Projektoren
  - LCD-Projektoren
  - DLP-Projektoren
  - Laser-Projektoren

Auf- und Rückprojektion

Vorteile Aufprojektion

- günstig
- geringer Platzbedarf

Vorteile Rückprojektion:

- qualitativ besser
- heller
- Betrachter kann keinen Schatten erzeugen

Projektoren: Röhrenprojektoren



- Die ersten stark verbreiteten Videoprojektoren verwendeten spezielle Kathodenstrahlröhren zur Darstellung des Bildes.
- Diese Röhren wurden auf eine sehr hohe Helligkeit getrimmt, um genügend Licht für die Projektion zu liefern.
- Für Farbprojektoren werden drei Röhren – eine für jede Grundfarbe – verwendet, die in der Regel getrennte Objektive hatten.

LCD-Projektoren

- engl: Liquid Crystal Display (LCD)
- Flüssikkristalldisplays



LCD-Projektoren

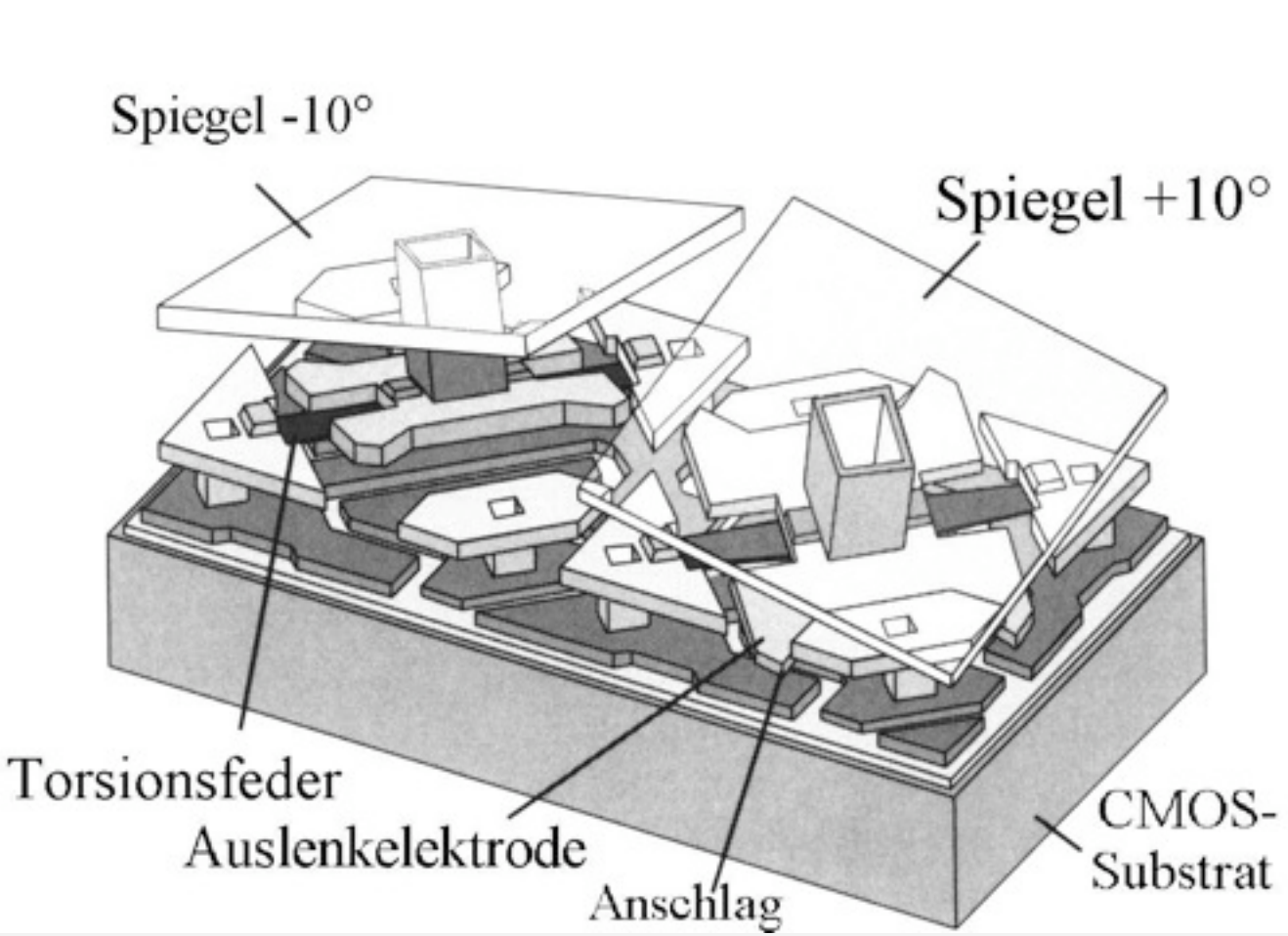
- Sony VPL-VW50 Full-HD LCD-Projektor:
  - Auflösung: 1.920 × 1.080 Pixel
  - Kontrast: 15.000:1
  - Helligkeit: 900 ANSI Lumen
  - Energiebedarf: 300 Watt
  - Gewicht: 11 kg
  - Tiefe: 47,1 cm
  - Preis: 1.879,00 €

(Quelle: www.hardwareschotte.de, Online: 8. Mai 2008)

DLP-Projektoren

- Digital-Light-Processing (DLP)
- Basis ist ein Mikrospiegelarray, engl. Digital Micromirror Device (DMD)
- DLP™ ist die Handelsmarke von Texas Instruments, 1987
- Unterscheidung:
  - 1-Chip-DLP, mit Farbrad
  - 3-Chip-DLP, ein Chip pro Farbe
- aktuelle höchste Auflösung: 2.048 × 1.080 Pixel

DLP-Projektoren



DLP-Projektoren

- Vorteile DLP gegenüber LCD:
  - kein Einbrennen
  - kein polarisiertes Licht, dadurch höhere Lichtleistung
  - weichere Rasterung
  - höherer Kontrast
  - keine Nachzieheffekte
  - für Stereoprojektion

DLP-Projektoren

Nachteile

- bei Ein-Chip-DLP: Regenbogeneffekt
- bei Ein-Chip-DLP: Pixelfehler

Fazit

- 3-Chip-DLP-Projektoren werden überwiegend für professionelle Projektionen verwendet
- Beispiel: Digitales Kino

DLP-Projektoren

- Samsung SP-A800B DLP:
  - Auflösung: 1.920 × 1.080 Pixel
  - Kontrast: 10.000:1
  - Helligkeit: 1000 ANSI Lumen
  - Energiebedarf: 300 Watt
  - Gewicht: 9,8 kg
  - Tiefe: 47,9 cm
  - Preis: 4.299,00 €

(Quelle: www.preisroboter.de, Online: 8. Mai 2008)

LCoS-Projektoren

- Liquide Crystal on Silicon (LCoS)
- Unterscheidung 1-Chip-LCoS und 3-Chip-LCoS
- kombiniert Vorteile aus DLP und LCD
- bestehend aus Siliziumfolie, dünne Beschichtung aus Flüssigkristallen, Glasscheibe

Laser-Projektoren

- Diodengepumpter Festkörperlaser
- generiert RGB-Farben
- Lichtleitfaser führt Licht zum Projektionskopf
- 2-achsiger-Spiegelscanner lenkt Laser um

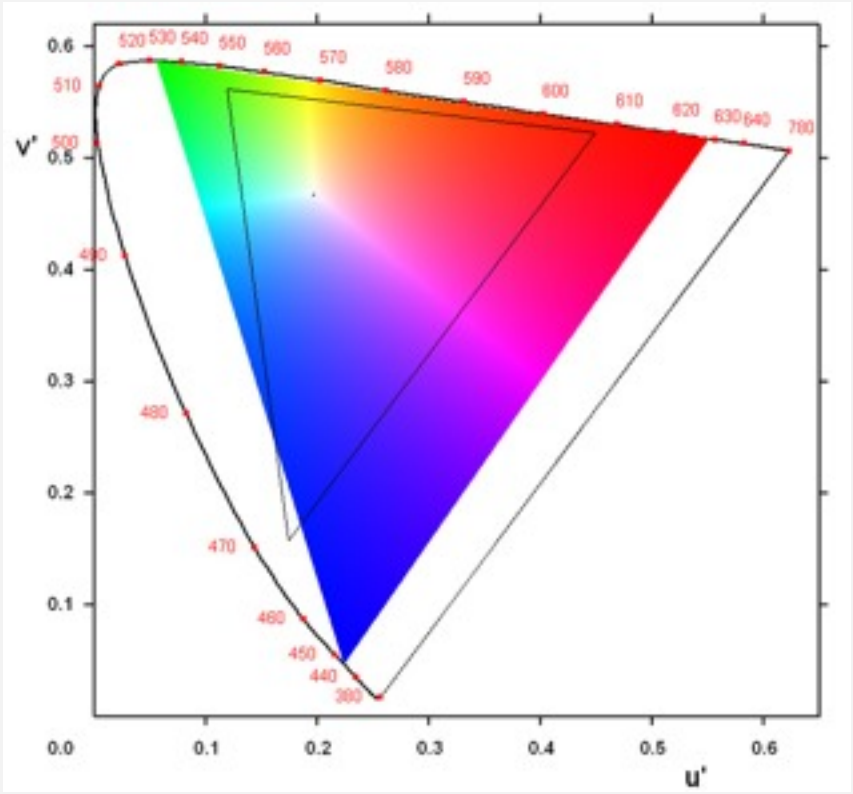
Laser-Projektoren





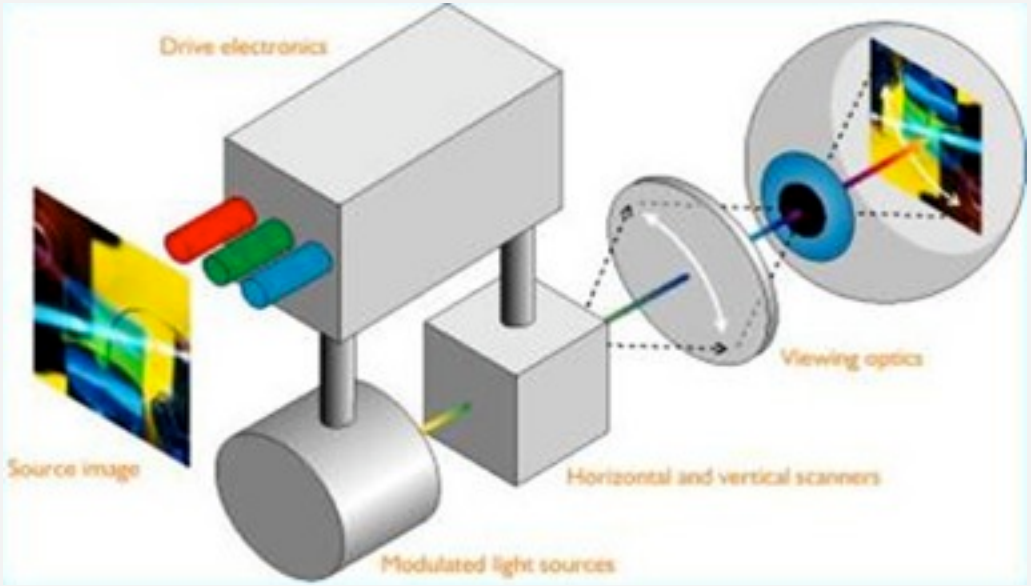
Laser-Projektoren

- Vorteile von Laser Projektoren:
  - hoher Kontrast (50000:1)
  - brillante Farben
  - unbegrenzte Schärfentiefe
  - Mehrkanalfähigkeit
  - kleines Objektiv

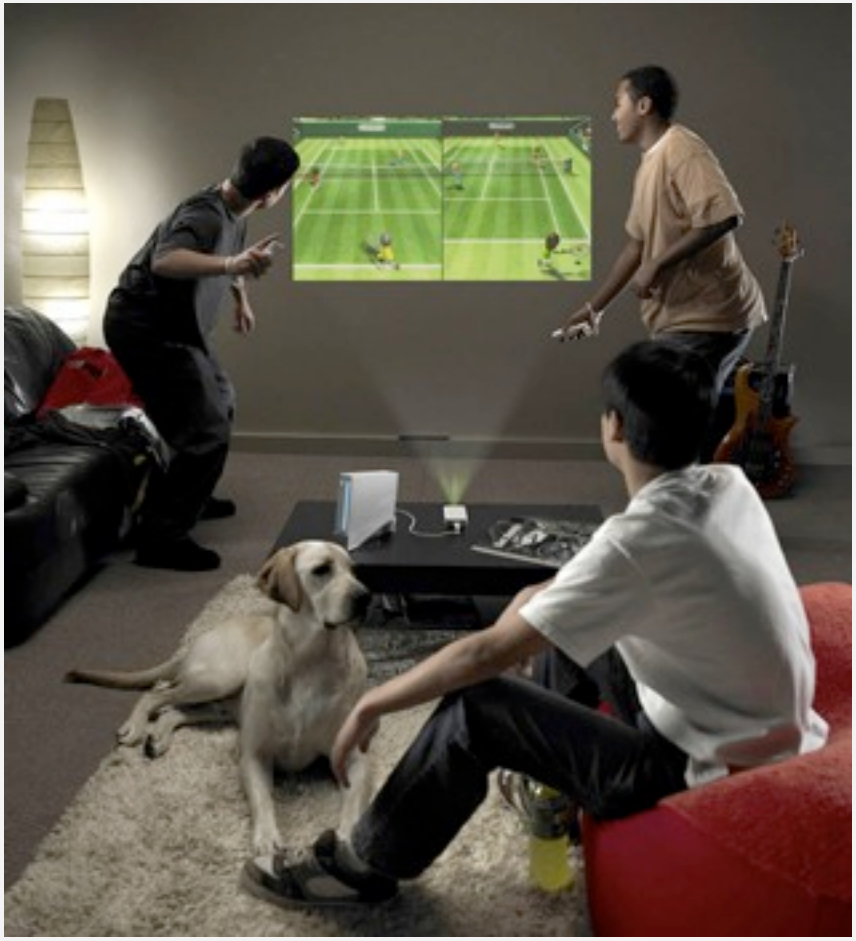


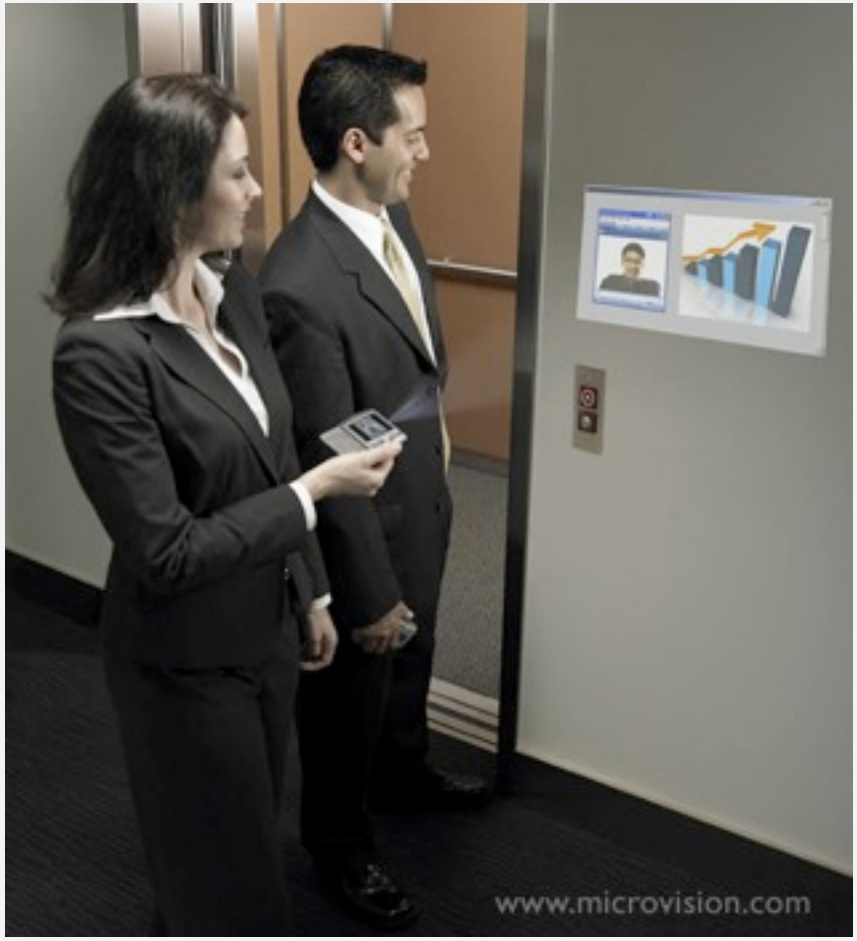
Virtual Retinal Display

- Laserprojektion direkt ins Auge
- führender Hersteller: Microvision
- vom Hersteller sind nur Fotomontagen erhältlich



Virtual Retinal Display





Exkurs: Spezifikationen

- VGA
- RGB HV
- HDTV
- DVI
- HDMI
- DisplayPort

VGA

- VGA (Anschluss), engl: Video Graphics Array
- 15-poliger Mini-D-Sub-Stecker
- Kabellänge je nach Abschirmung: 5-30m
- VGA für Auflösung über SXGA (1.280 × 1.024) nur bedingt geeignet, da störanfällig



RGB HV

- RGB HV, engl: Red, Green, Blue, H-Sync, V-Sync
- BNC-Stecker für analoge Übertragung für lange Stecken
- Reichweite bis ca. 100 Meter
- Auflösung mindestens bis zu 2.048 × 1.536

HDTV

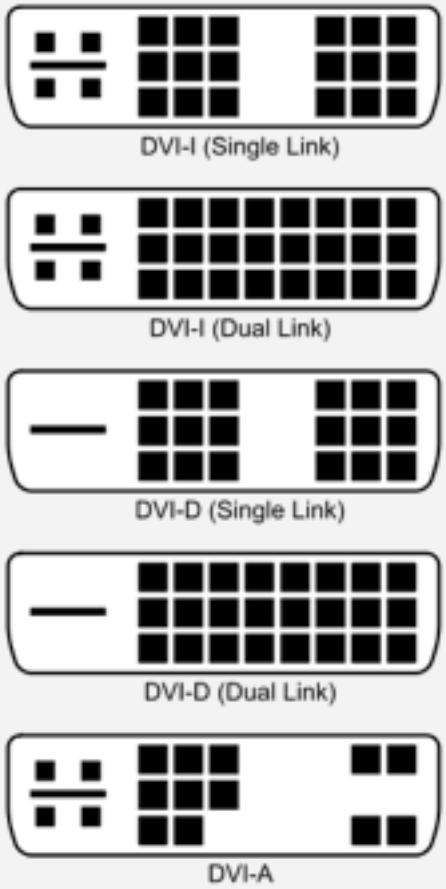
- HDTV, engl: High-Definition TV
- Angabe: [Zeilenzahl] + [Bildschirmaufbau] + [Bildwiederholrate]
  - 1.080i60 – 1.920 × 1.080, 60 Halbbilder pro Sekunde
  - 1.080p24 – 1.920 × 1.080, 24 Vollbilder pro Sekunde
  - 720p50 – 1.280 × 720, 50 Vollbilder pro Sekunde

DVI

- DVI, engl: Digital Visual Interface
- maximale Länge: 15 Meter
- Übertragungsrate: 3,7 Gbit/s
- Single Link Übertragung: 1.915 × 1.436 Pixel
- kompatible zu HDMI

DVI – Stecker und Buchsen

- DVI-I, bei Dual Link Aulösung von QXGA möglich, 2560 × 1600 bei 60 Hz
- DVI-A, maximale Aulösung von WUXGA möglich, 1920 × 1200, analog



HDMI

- HDMI, engl: High Definition Multimedia Interface
- Kopierschutz HDCP, engl: High-bandwith Digital Content Projection, in der HDMI-Spezifikation vorgesehen
- maximale Länge: 15 Meter
- Übertragungsrate: 5 Gbit/s (Typ A), 10 Gbit/s (Typ B)



DisplayPort

- DisplayPort, VESA genormter universeller Verbindungstandard
- Standard von Mai 2006
- von der Industrie akzeptiert
- kompatible zu VGA, DVI, HDMI
- verschlüsselt Signale
- Übertragungsrate: 10,8 Gbit/s (vier Leitungen)

Force Feedback

- Erstmals als Erweiterung für Nintendo 64 Controller
- Force Feedback – Vibration im Controller als Rückmeldung



Quellen

- Foley, van Dam, Feiner, Hughes: Computer Graphics, Principles and Practice. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim: Computergraphik 1. Universität Magdeburg, Vorlesungsskript, Juli 2005.
- Beate Betz: Hardcopygeräte. MEDIADESIGN-Hochschule, Power-Point-Präsentation, März 2007.
- Jena Optik, WWW-Präsentation, Mai 2008. www.jenaoptik-laserdisplays.de
- Timo Hanisch: Eingabegeräte. MEDIADESIGN-Hochschule, Power-Point-Präsentation, März 2007.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Computergrafik I

Farbe und Sichtbeschreibung

Letztes Mal

- Rendering-Pipeline
- Koordinatensysteme
- Transformationen

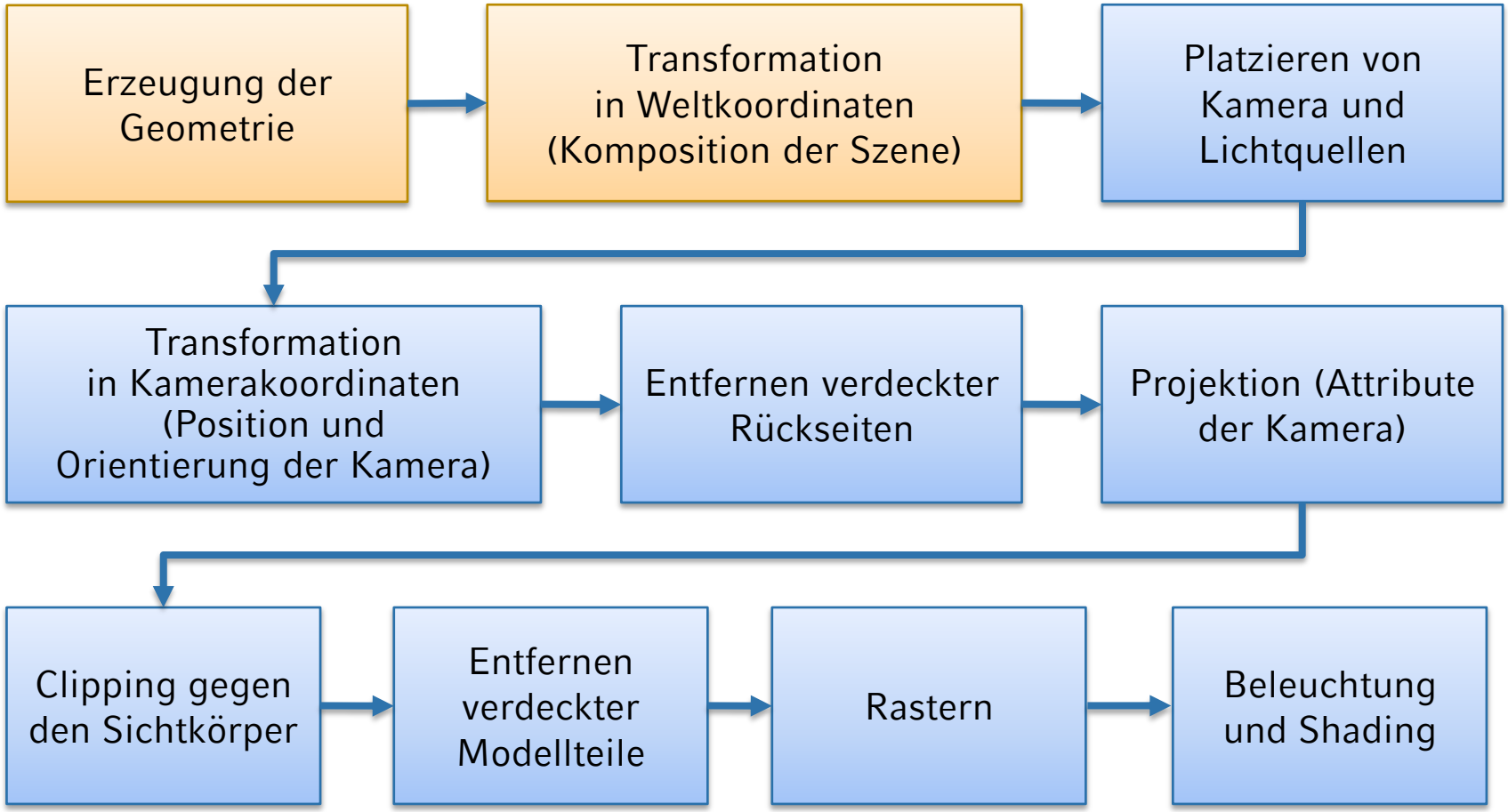
Geometrische Modelle: Polygonale Modelle

- Definition eines Polygons:
  1. Angabe der Koordinaten der Eckpunkte
     - *x*-, *y*-, *z*-Koordinatentripel
     - Verbindung der Eckpunkte zu einem Polygon wird implizit hergestellt:
       - Verbinde den *i*-ten Punkt mit dem (*i* + 1)-ten
       - Verbinde den letzten mit dem ersten zum Schließen des Polygons

Geometrische Modelle: Polygonale Modelle

2. Auflistung aller Eckpunkte (vertices) in einer Liste
   - Verbindung wird angegeben durch eine Liste von Indizes (Kanten definiert als indizierte Eckpunktliste)
   - Vorteil: kompakte Speicherung (keine Redundanz), Projektion/Transformation des Modells: Iteration über alle Eckpunkte
3. Weitere komplexere Modellierungstechniken

Rendering-Pipeline



Fahrplan für heute

- Farbe
- Zusammenbau der Szene

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik und -animation

Farbe

Farbe in der Computergraphik

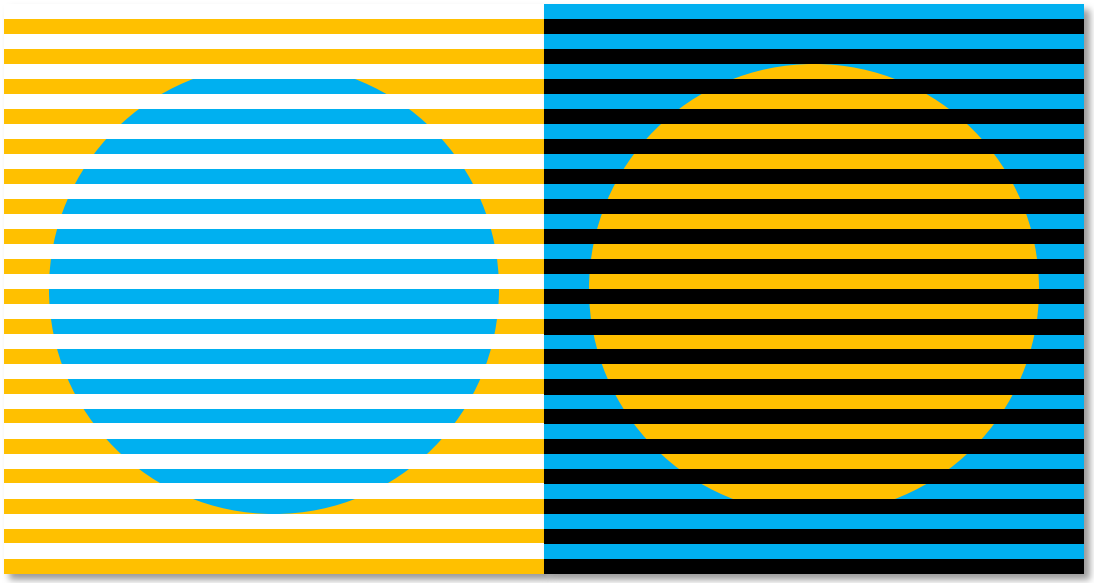
Motivation:

- Realistische Farbdarstellungen (Nachbildung der Lichtausbreitung)
- Farbmodelle zur intuitiven Spezifikation
- Farbmodelle zur Konvertierung von Farben zwischen unterschiedlichen Medien

Farbe in der Computergraphik

- Grundlagen: Physik (Optik, Lichtausbreitung), Physiologie und Psychologie
- Visuelle Wahrnehmung von Farben und Farbunterschieden: verschiedene Theorien
- Wahrgenommene Farbe eines Objektes ist abhängig von:
  - Lichteinfall,
  - Reflexions- und Transmissionseigenschaften,
  - Farbe umgebender Objekte und
  - visueller Wahrnehmung (Unterschiedlich bei jungen und alten Menschen, Wahrnehmungsstörungen).

Farben – Wahrnehmung



- Die blauen und orangen Farben sind in beiden Bildern exakt gleich

Farben – Wahrnehmung

- Die Wahrnehmung von Farben wirkt auf zweierlei Art psychologisch:
  1. Hervorrufen von Assoziationen– also Vorstellungen, meistens Erinnerungen an Dinge wie
     - Feuer (Rot),
     - Gras (Grün),
     - Zitrone (Gelb), siehe die Tabelle oben.

Farben – Wahrnehmung

2. Gefühle (Farbgefühl, Gefühlston, Anmutungsqualität, Gefühlscharakter) kann hervorgerufen werden bei Verwendung von Eigenschaften

   - gefährlich (Rot),
   - giftig (Grün),
   - frisch (Gelb).

Farben – Wahrnehmung

- Assoziationen und Gefühle in Folge von Farbwahrnehmung gehen in die Schöpfungen der Kultur(en) ein.
- Nach der „Empiristischen Theorie der Gefühlswirkung von Farben“ werden die Farbgefühle individuell und implizit (unbewusst, nicht erinnerbar) gelernt: Gefühle, die der Mensch auf Grund seiner ererbten Triebstruktur und Daseinsthematik ursprünglich gegenüber bestimmten überall vorkommenden „Universalobjekten“ bzw. „Universalsituationen“ entwickelt

Farben – Wahrnehmung: Universalobjekte

- Der blaue Himmel,
- das Wasser,
- die grüne Vegetation,
- das rote Feuer,
- Blut,
- die gelbe Sonne,
- der braune Erdboden,
- Fäkalien,
- der graue Felsen,
- die schwarzen Brandreste.

Farben – Wahrnehmung: Universituationen

- Situationen, in denen sich der Mensch täglich befindet,
- etwa die dunkle (schwarze) Nacht und
- der helle (weiße) Tag

Farbe in der Computergraphik

Was ist Licht?

- Sichtbarer Teil des elektromagnetischen Spektrums.
- Wellenlängenbereich von 380 bis 780 nm (bei Tieren unterschiedlich).
- Frequenz des sichtbaren Lichtes entsprechend: ~10^{15} Hz.
- Charakterisiert durch (physikalische Größe) Licht-Intensität und die (wahrgenommene Größe) Helligkeit.
- Licht breitet sich sehr schnell in alle Richtungen aus. Es wird an Oberflächen reflektiert und teilweise gebrochen bzw. von strahlenden Oberflächen emittiert.

Zusammenhang: Farbe und Licht

- Farbe ist die Wahrnehmung der spektralen Zusammensetzung des Lichtes (welcher Anteil des einfallenden Lichtes hat welche Wellenlänge)

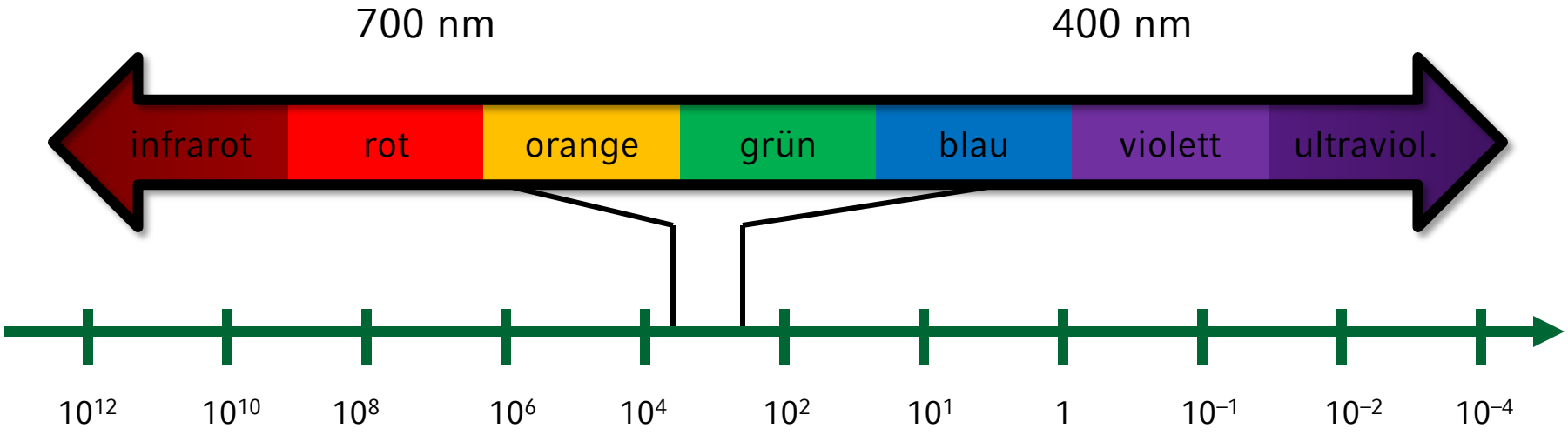
Farbwahrnehmung

Ausgangspunkt:

- Wellentheorie des Lichtes → Farbwahrnehmung elektromagnetischer Strahlung im Wellenlängenbereich zwischen 380 und 780 nm.
- Lichtwahrnehmung in zwei Schritten:
  1. Reizaufnahme durch Rezeptoren auf der Retina
     - Stäbchen: für Schwarz-Weiß-Sehen auch bei geringer Intensität (≈ 120 Millionen)
     - Zapfen: für Farbwahrnehmung (≈ 6,5 Millionen)
  2. Verarbeitung der Reize in mehreren Stufen (Kontrastverstärkung am Ausgang der Retina, Interpretation im visuellen Kortex des Gehirns)

Farbwahrnehmung; Zapfen für das Farbsehen

- Drei Arten mit unterschiedlicher Wellenlänge (Rot, Grün, Blau im Verhältnis 10:10:1) → Wichtig für Gestaltung von Farbskalen!
- Breite der Zapfen etwa 3 µm ⇒ Ortsauflösung 50 Bogensekunden



Farbwahrnehmung

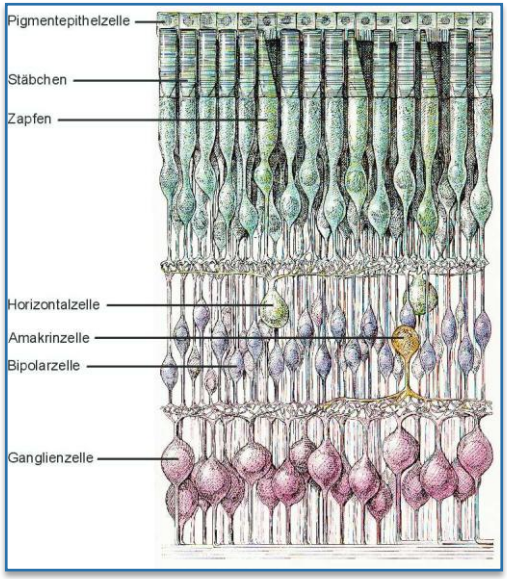
- Farben, die durch die Zerlegung weißen Lichts entstehen, werden als Spektralfarben bezeichnet.
- Spektrum enthält nur einen kleinen Teil der Farben – nämlich nur reine Farben, und von diesen nicht diejenigen auf der sogenannten „Purpurlinie“ zwischen Violett und Rot.
- Alle anderen, die durch Farbmischung entstehen, werden als „Mischfarben“ bezeichnet.
- Bestimmte Mischfarben können dem menschlichen Auge durch Metamerie als identisch mit bestimmten Spektralfarben erscheinen.

Farbwahrnehmung

Farbe	Wellenlänge
Rot	≈ 625–740 nm
Orange	≈ 590–625 nm
Gelb	≈ 565–590 nm
Grün	≈ 520–565 nm
Cyan	≈ 500–520 nm
Blau	≈ 450–500 nm
Indigo	≈ 430–450 nm
Violett	≈ 380–430 nm
Magenta	Mischfarbe
Braun	Mischfarbe

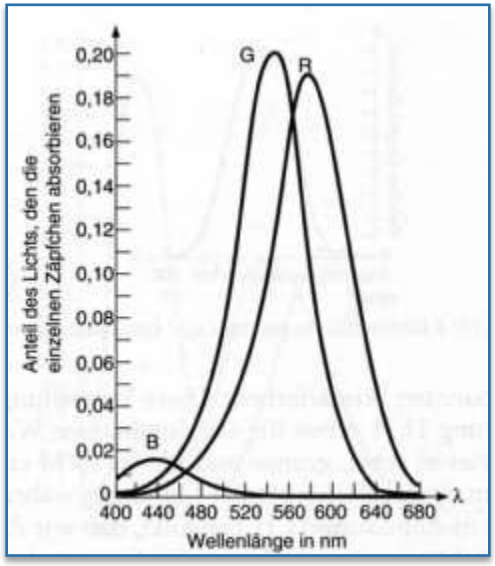
Farbwahrnehmung - Auge

- Bild wird auf Netzhaut „projiziert“
- Photorezeptorzellen wandeln Lichtimpulse in elektrische Impulse um
- Stäbchenzellen:
  - Hell-Dunkel
  - 498 nm
- Zapfenzellen
  - Farbe
  - 3 Empfindlichkeiten



Farbwahrnehmung - Auge

- Experimente zum Farbempfinden
- Spektralreaktion
- Reaktion auf blaues Licht schwächer



Farbwahrnemung

Objektive Farbmerkmale

- **Dominante Wellenlänge**: Die Wellenlänge aus dem Spektrum, bei der die höchste Leistung abgestrahlt wird.
- **Reinheit**: Physikalisches Maß, das für ein gegebenes Licht angibt, in welchem Verhältnis weißes Licht zu einem monochromatischen Licht zu mischen ist, um das gegebene Licht zu erzeugen.
- **Luminanz**: Beschreibt die Strahlungsenergie; gibt Intensität bezogen auf den Flächeninhalt eines unendlich kleinen Flächenelementes an, das sich auf der Lichtquelle befindet.

Farbwahrnemung

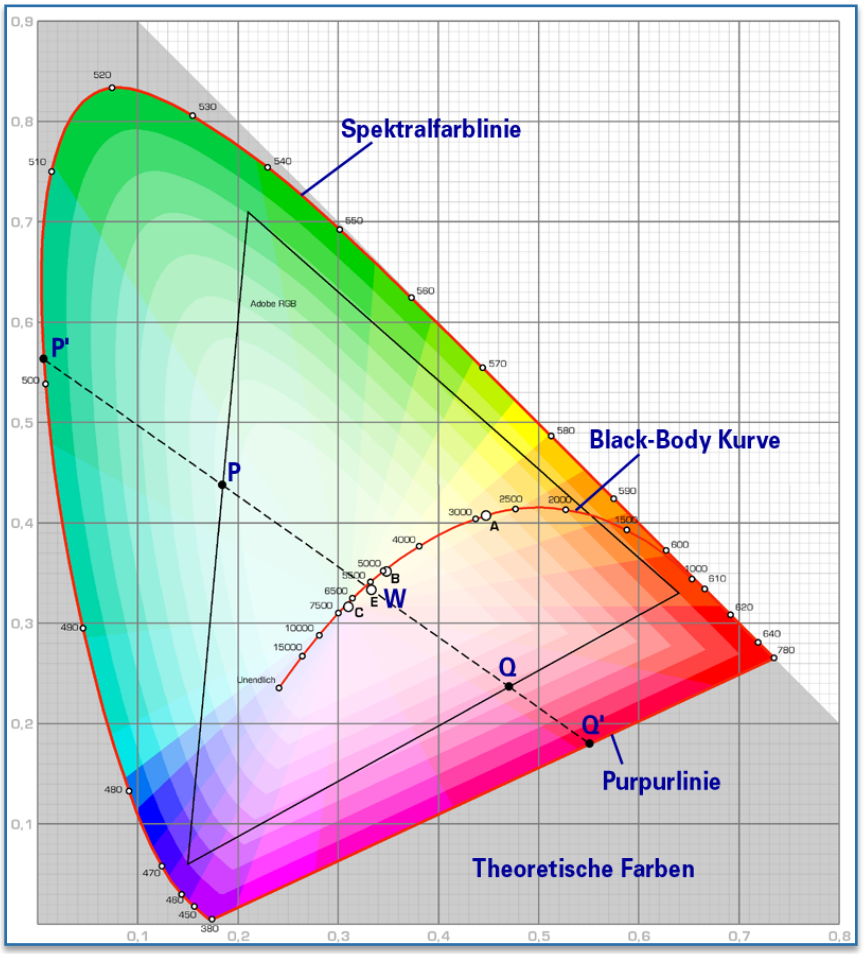
- **Subjektive Farbmerkmale**
- Helligkeit: Physiologisch-psychologisches Maß für die Stärke des durch einen Beobachter wahrgenommenen Gesamtenergieflusses
  - Lightness: Helligkeit eines reflektierenden Objektes
  - Brightness: Helligkeit eines selbstleuchtenden Objekte (Lampe, Sonne, Bildschirm)
- Farbton (Hue): physiologischer Begriff zur Unterscheidung verschiedener Spektralmuster; unterscheidet zwischen reinen Farben (rot, gelb, grün, blau, usw.)
- Sättigung (Saturation): physiologisches Maß für den Grad, in dem der wahrgenommene Farbton eines Lichtes von dem Farbton eines weißen Lichtes gleicher Luminanz abweicht.

Farbspezifikation und Farbräume

- 3 Arten von Photorezeptoren → Abbildung jeder Farbe ist durch Mischung aus 3 Primärfarben (mit bestimmten Gewichten möglich)
- CIE = Commission Interlationale de l'Eclairage
- Abbildung einer spektralen Leistungsverteilung auf 3D-Koordinaten in einem Farbraum
- CIE-Diagramm zur Farbstandardisierung (1931)

Farbspezifikation und Farbräume

- Farbmodell: Spezifikation eines 3D-Koordinaten-systems und einer Untermenge davon, in der alle sichtbaren Farben eines bestimmten Farbbereiches (Gammut) liegen.
- CIE-Diagramm und Bildschirmgammut



Farbspezifikation und Farbräume

- Hardwareorientierte Farbmodelle:
  - Motiviert durch die Charakteristika von Ausgabegeräten.
  - Beispiele: RGB- und CMY-Modell
- Perzeptionsorientierte Farbmodelle:
  - Gleiche Abstände im Farbraum korrespondieren zu (etwa) gleichen Abständen in der Farbwahrnehmung.
  - Nutzung von physiologischen Größen: Farbton, -sättigung, -helligkeit
  - Beispiele: HLS- und HSV-Modell
- Hardwareorientierte Modelle sind unerlässlich; perzeptionsorientierte für die Farbeingabe wünschenswert. → Transformation notwendig.

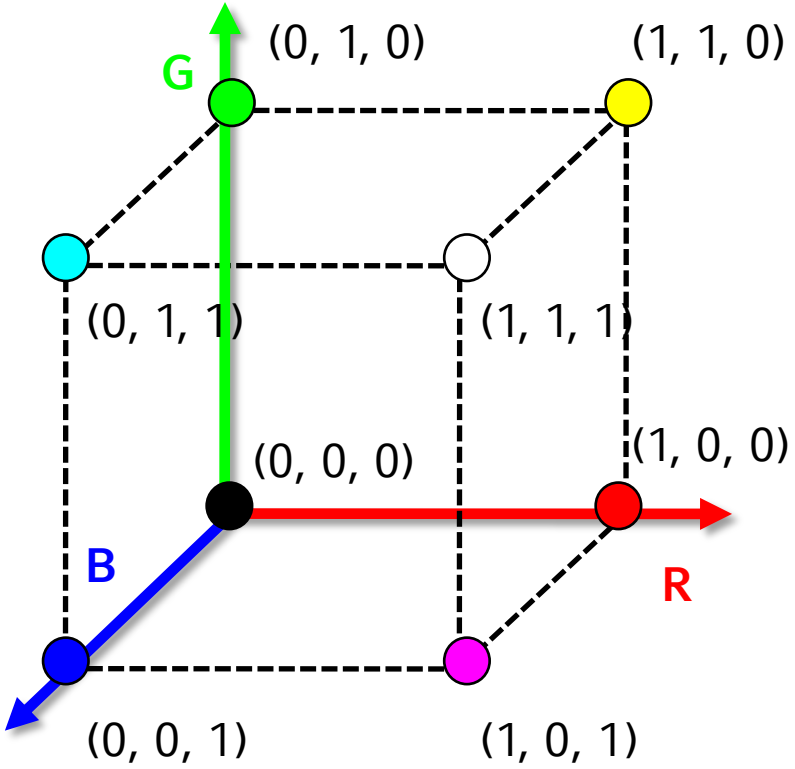
Farbspezifikation und Farbräume

Hardwareorientierte Farbmodelle:

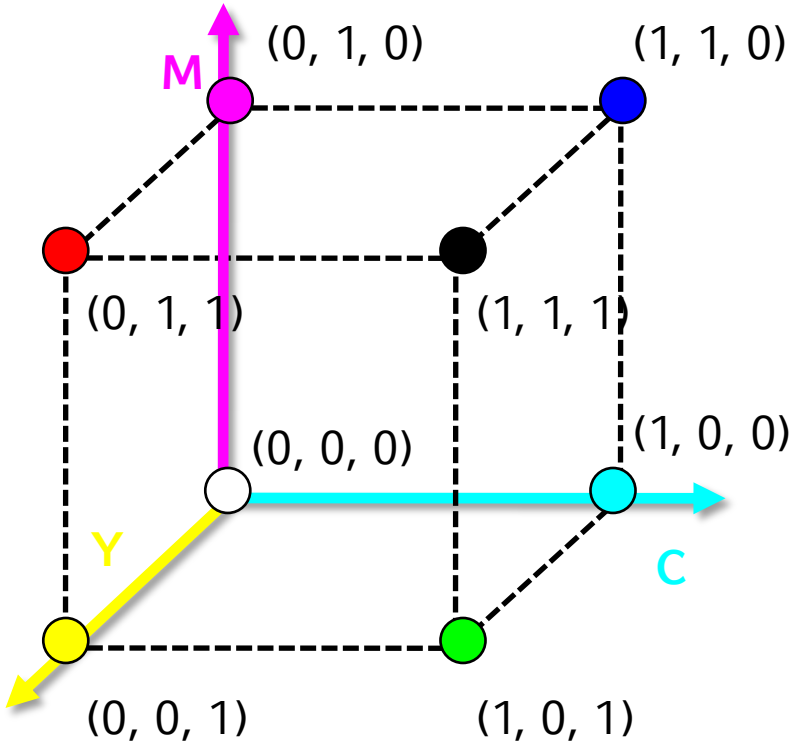
- **RGB:** Additives Farbmodell, bei dem Farben als Linearkombination ihres Rot-, Grün-, Blau-Anteils beschrieben werden. Genutzt für Ansteuerung von Monitoren.
- **CMY:** Subtraktives Farbmodell, bei dem Farben als Linearkombination ihres Cyan, Magenta, Yellow-Anteils beschrieben werden. Genutzt für Ansteuerung von Druckern.
- Beim Druck: Hinzunahme von reinem Schwarz.

Hardwareorientierte Farbmodelle

RGB-Farbmodell

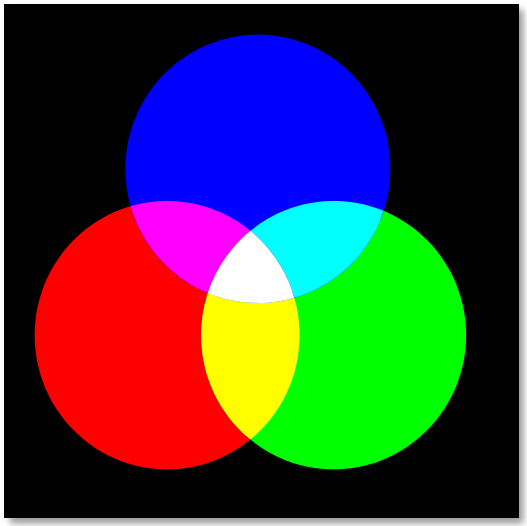


CMY-Farbmodell

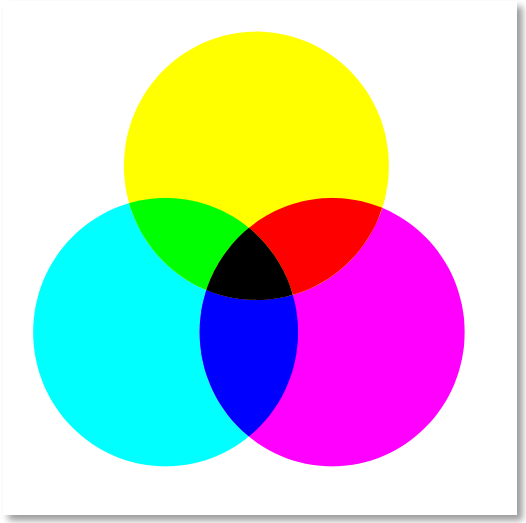


Hardwareorientierte Farbmodelle

additive Farbmischung



subtraktive Farbmischung



Farbspezifikation und Farbräume

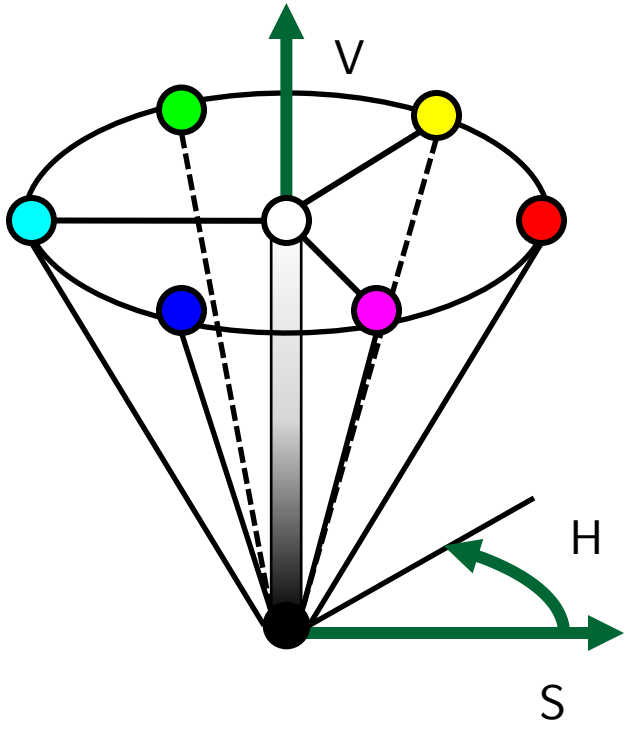
- Konvertierung: RGB ↔ CMY
- Einfaches, angenähertes Verfahren

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} C \\ M \\ Y \end{pmatrix}$$

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

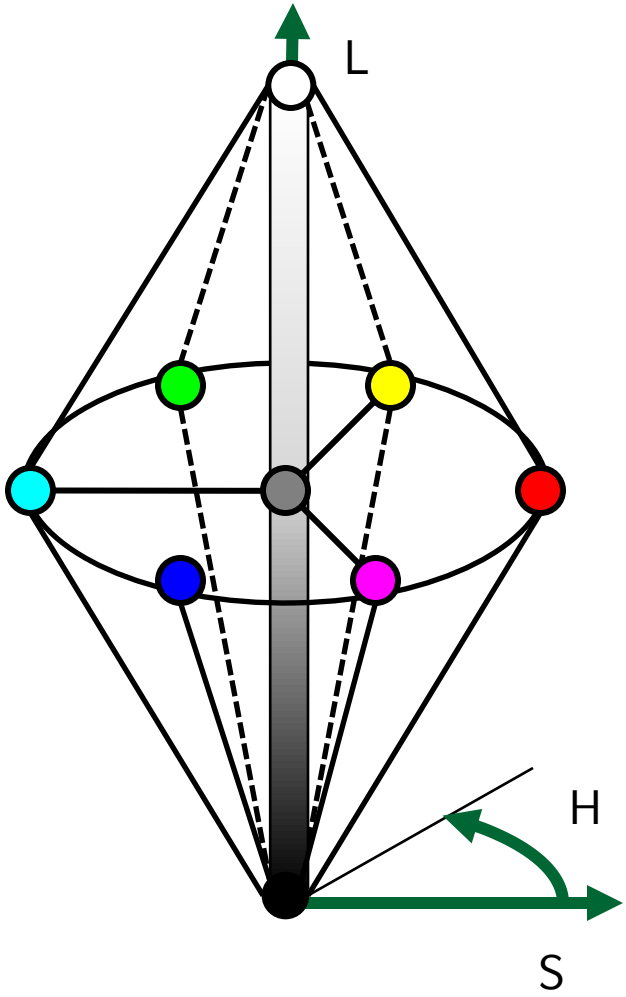
Perzeptionsorientierte Modelle: HSV

- Hue, Saturation, Value
- Hue – Winkel um vertikale Achse, 0° entspricht rot
- Änderung der Sättigung
  - 0 … 1
- Helligkeit entspricht dem Schwarzanteil
- Beispiel: gesättigtes dunkelblau:
  - H = 225°,
  - S = 100 %,
  - V = 100 %.



Perzeptionsorientierte Modelle: HLS

- Hue, Lightness, Saturation
- Strategie von Malern: nimm reines Pigment (H), Weiß dazu (S), Schwarz dazu (1 – L)
- Komponenten nicht unabhängig voneinander
- Graustufen: S = 0
- Voll gesättigte Farben:
- L = 0,5, S = 1



Konvertierung: HLS in RGB

$$M_1 = S\sin(H)$$

$$M2 = S\cos(H)$$

$$I = \frac{L}{\sqrt{3}}$$

$$(R,G,B) = (M_1, M_2, I) \cdot \begin{pmatrix} \frac{2}{\sqrt{6}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{pmatrix}$$

Konvertierung: RGB in HLS

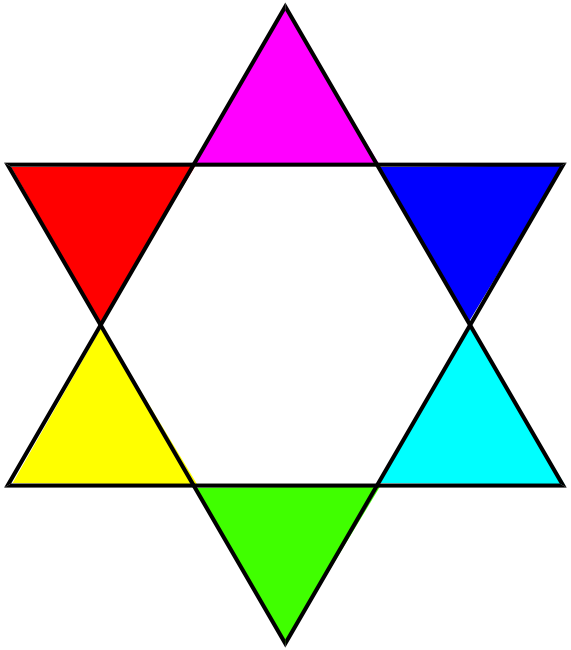
$$(M_1, M_2, I) = (R, G, B) \cdot \begin{pmatrix} \frac{2}{\sqrt{6}} & 0 & \frac{1}{\sqrt{6}} \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ -\frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \end{pmatrix}$$

$$H = \arctan\left(\frac{M_1}{M_2}\right)$$

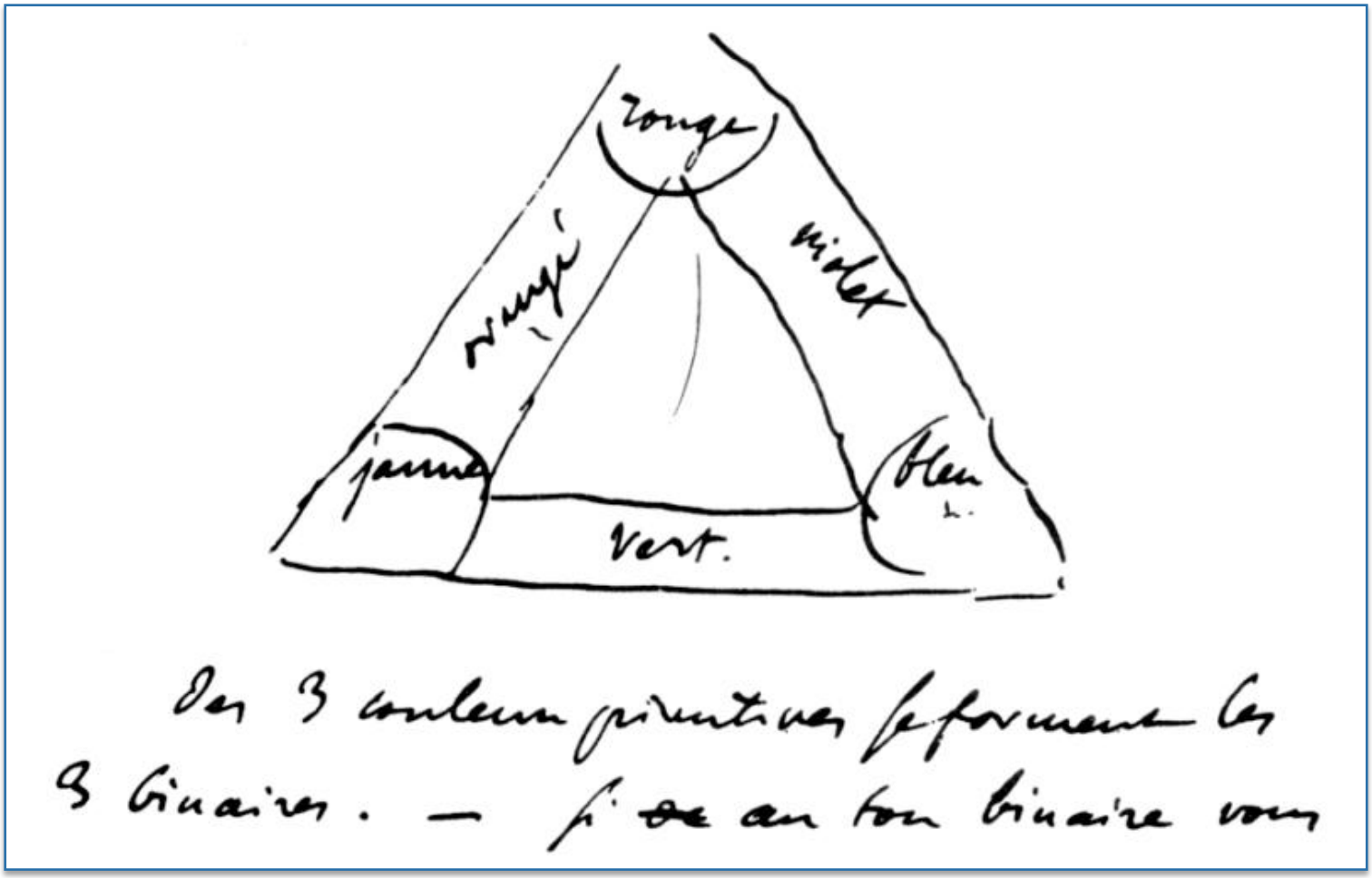
$$S = \sqrt{M_1^2 + M_1^2}$$

$$L = I_1\sqrt{3}$$

Farben – Exkurs: Zusammenfassung



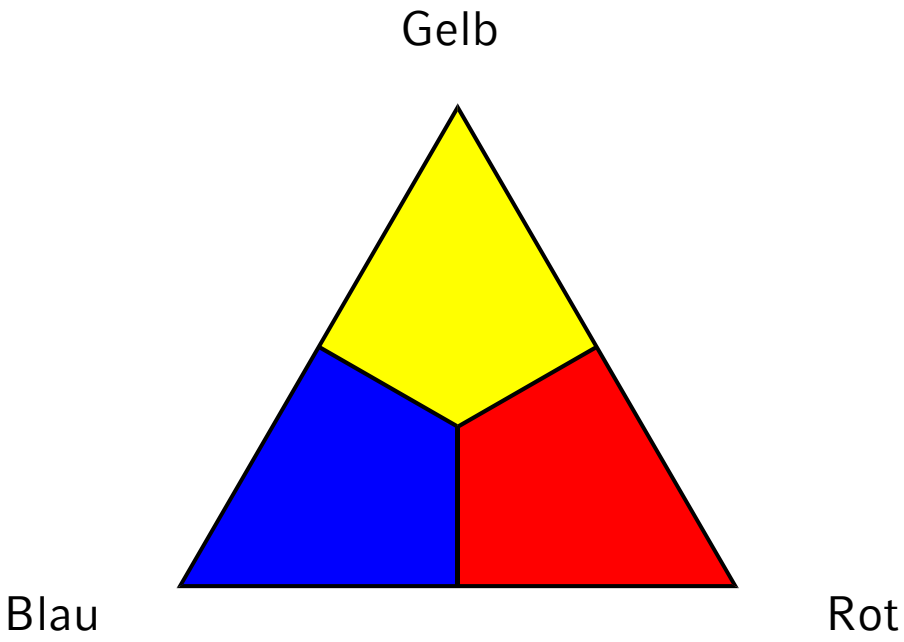
Farben – Exkurs: Delacroix



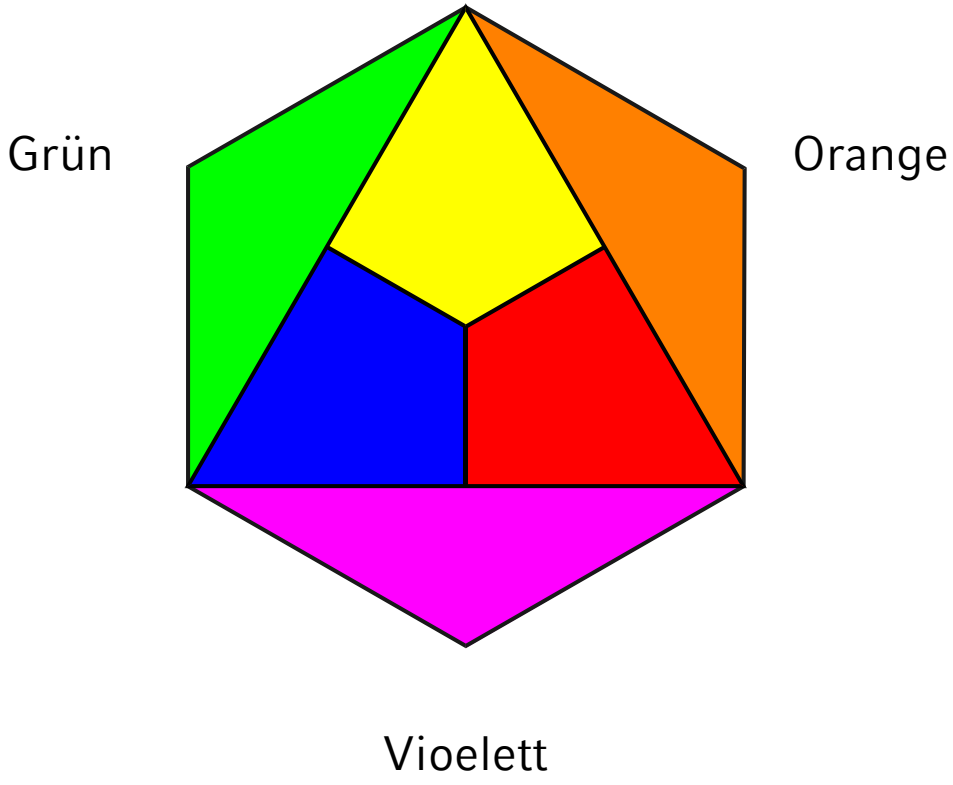
Farben – Exkurs:

- Zahlreiche Persönlichkeiten beschäftigten sich mit Arbeiten zur Theorie der Farben:
- Leonardo da Vinci – „Ein Künstler, welcher ohne Theorie ans Werk geht, ist wie ein Steuermann, der sich ohne Steuerruder und Kompass aufs Meer wagt.“ aus: „Traktat von der Malerei“,
- Isaac Newton,
- Johann Wolfgang Goethe,
- Philip Otto Runge,
- Johannes Itten, „Kunst der Farbe“, 1962.

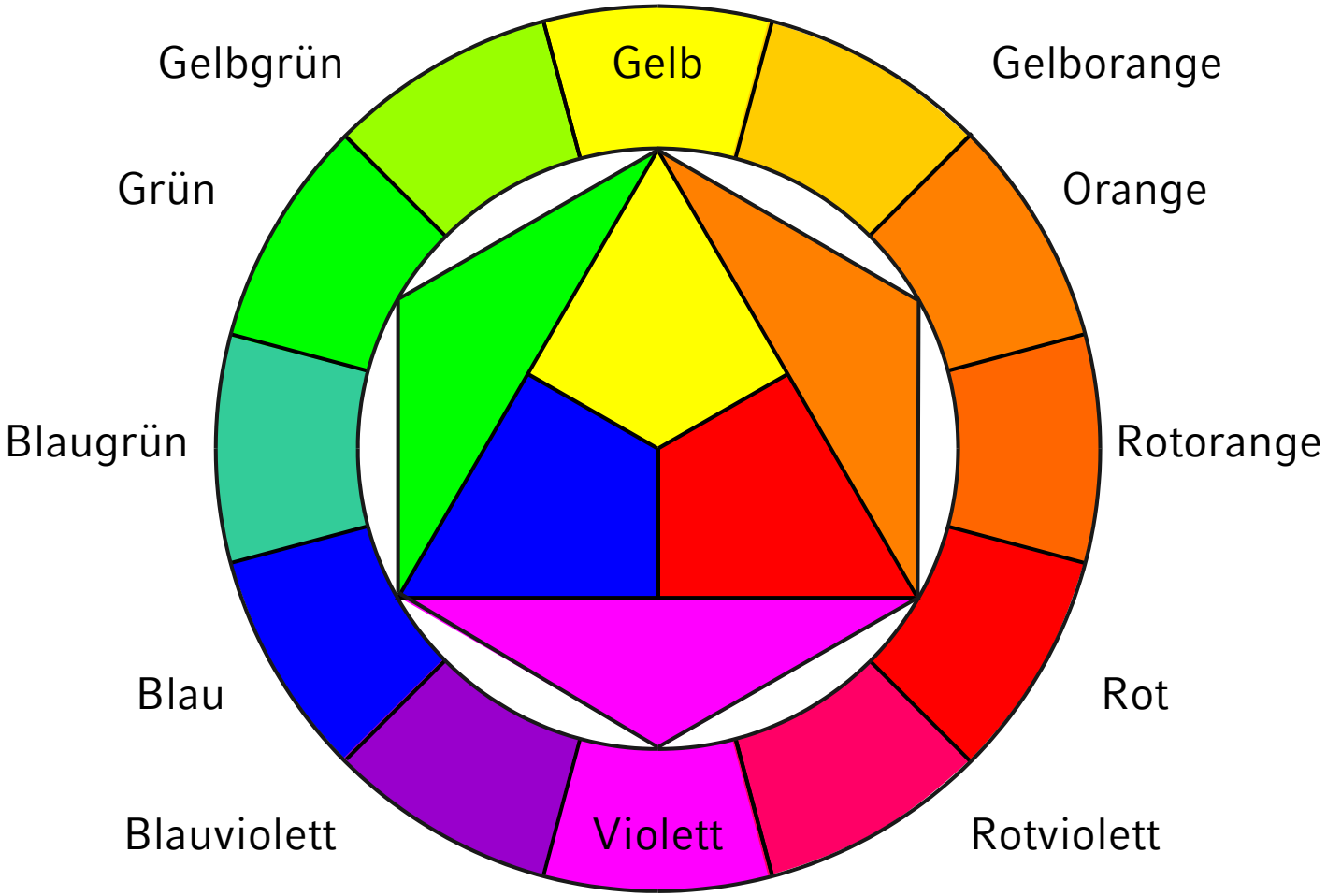
Farben – Exkurs: Farben 1. Ordnung



Farben – Exkurs: Farben 2. Ordnung



Farben – Exkurs: Farben 3. Ordnung



Farben – Exkurs: Farbkontraste nach Itten

- Von *Kontrast* spricht man, wenn zwischen zwei zu vergleichenden Farbwirkungen deutliche Unterschiede oder Intervalle festzustellen sind.
- Ist Unterschied maximal:
  - *polarer Kontrast*,
  - Farbwerte heißen dann *Pole* des Kontrastes

Farbkontraste: Farbe-An-Sich-Kontrast

- „Buntheit“ mehrerer Farben nebeneinander
- beschrieben durch mindestens drei Farben
- Maß ist Abstand der Farben untereinander auf dem Farbkreis
- Rot, Blau und Gelb ergeben maximalen Buntkontrast

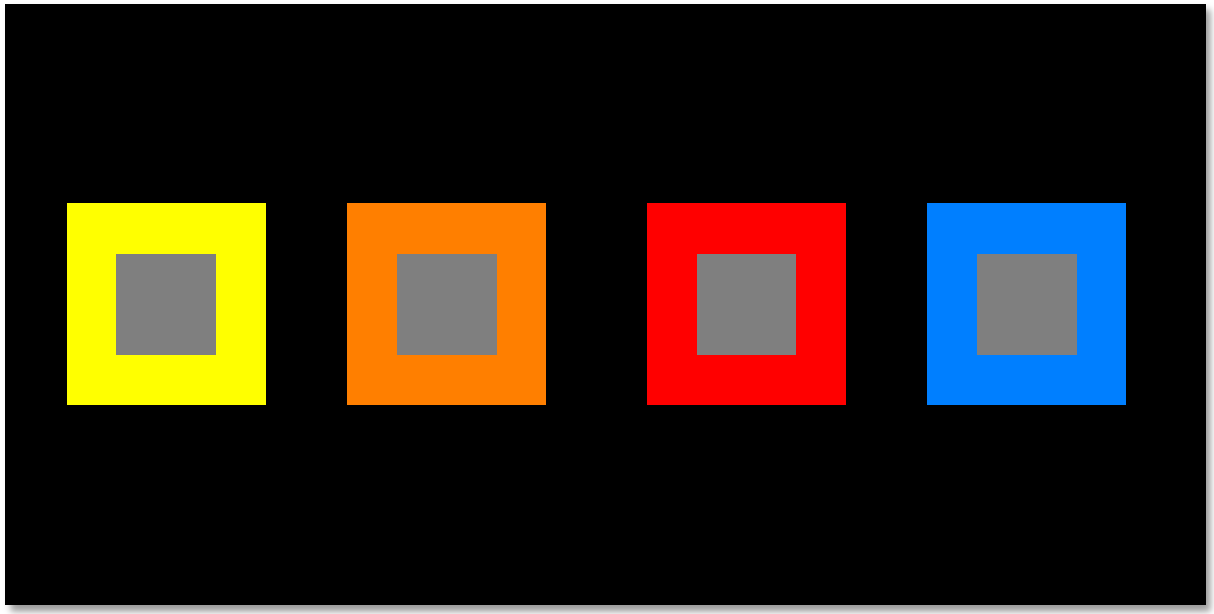
Farbkontraste: Hell-Dunkel-Kontrast

- Tonwertkontrast
- Abstand der Farben auf der Tonwertskala
- Pole:
  - Schwarz und
  - Weiß



Farbkontraste: Simultan-Kontrast

- basiert auf der Erscheinung, dass das menschliche Auge zu einer Farbe die Komplementärfarbe generiert
- Wirkung einer Farbe verändert sich durch umgebende Farbflächen
- Grau erscheint auf gelb hellviolett, auf orange bläulich usw.



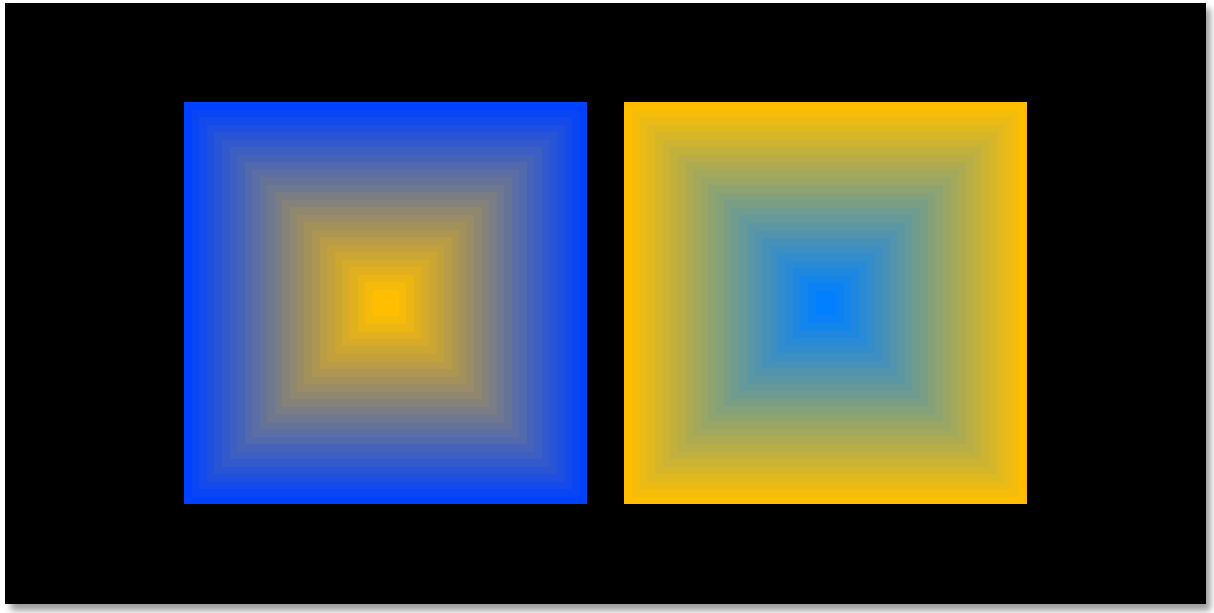
Farbkontraste: Simultan-Kontrast

- Streifen in der Mitte ist gleichmäßig gefüllt
- Kombination von Hell-Dunkel- und Simultankontrast



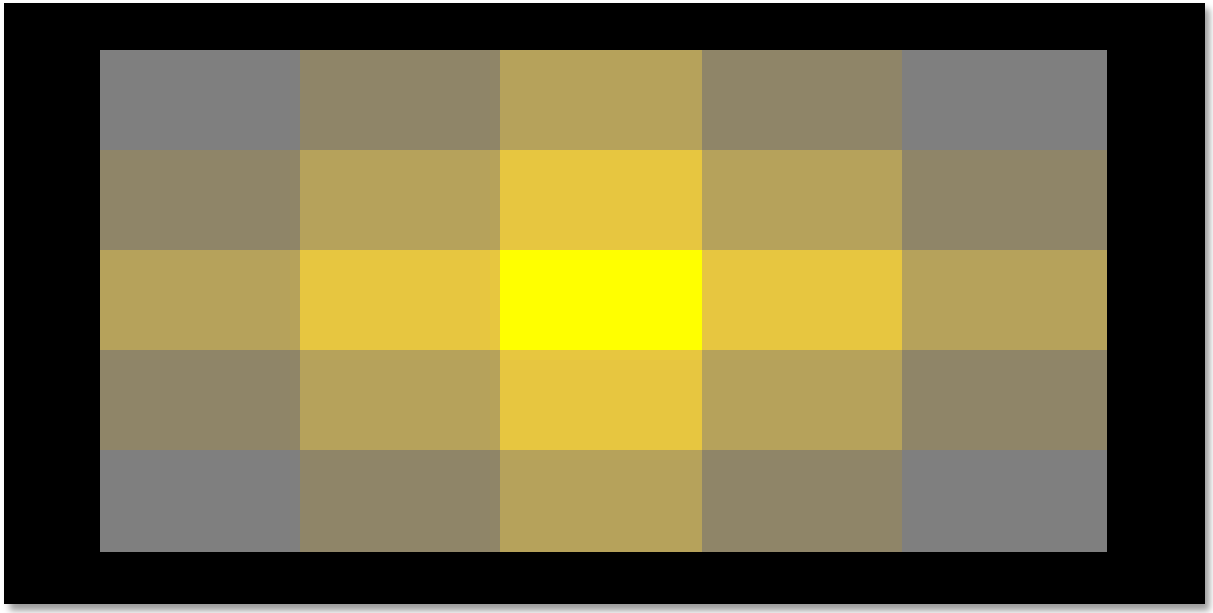
Farbkontraste: Warm-Kalt-Kontrast

- maximaler Kontrast zwischen Blaugrün (kalt) und Rotorange (warm)
- Abstand zweier Farben auf dieser Skala ist Maß für den Kontrast
- Teil des Warm-Kalt-Kontrastes ist die Wirkung, dass warme Farben räumlich näher wirken als kalte; deshalb auch Perspektivkontrast genannt



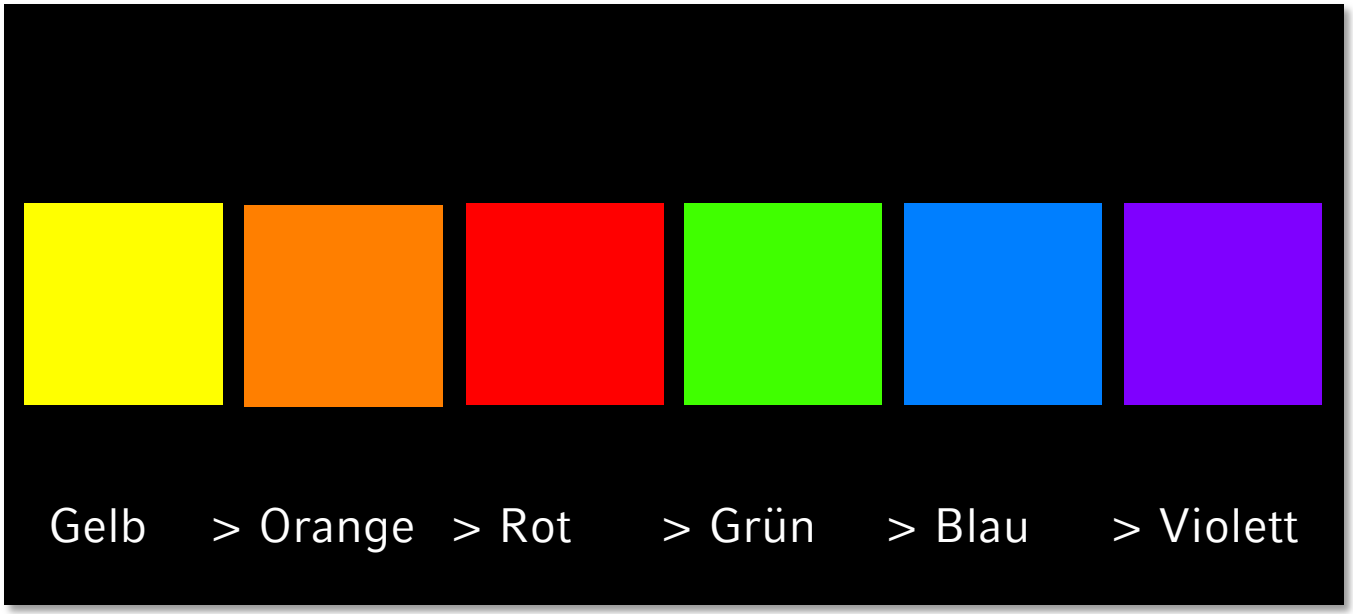
Farbkontraste: Qualitäts-Kontrast

- Unterschied zwischen Leucht- und Buntkraft von Farben
- beschrieben durch Relationen der Farben zueinander (Satt-Trübe, Leuchtend-Stumpf)
- Intensität einer Farbe kann gesteigert werden, indem sie in die Nähe einer Farbe mit schwächerem Tonwert gebracht wird



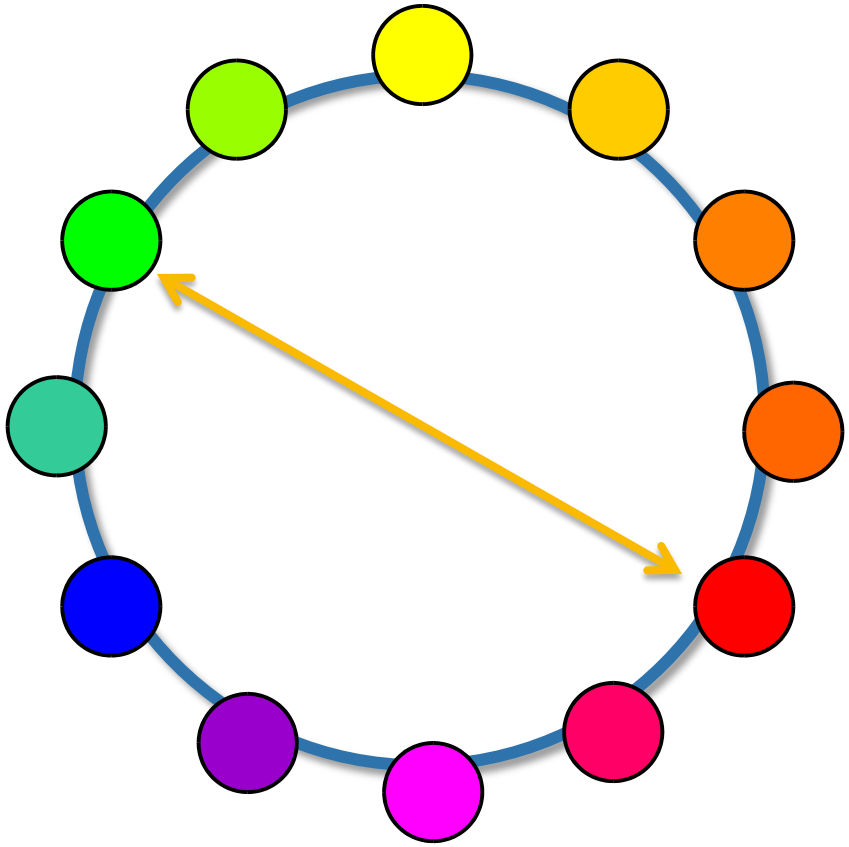
Farbkontraste: Quantitäts-Kontrast

- „Kräfteverhältnis“ von Farben in ihrer Wirkung bezüglich ihrer Ausdehnung
- etwa so wirkt eine orange Fläche größer als eine gleichgroße blaue
- Goethe: Farben haben „Gewichte“



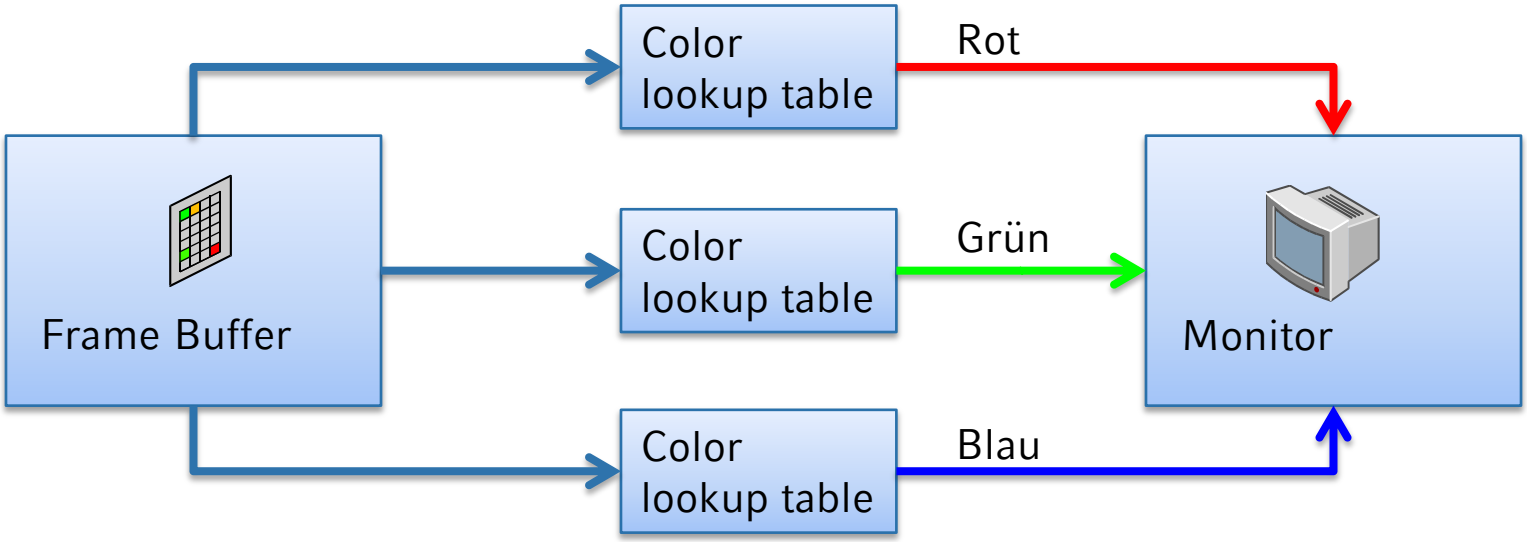
Farbkontraste: Komplementär-Kontrast

- Komplementärfarben liegen einander im Farbkreis gegenüber
- steigern sich gegenseitig zu höchster Leuchtkraft



Farbspezifikation

- Indizierte Farben/Paletten – Motivation:
  - Bei hoher räumlicher Auflösung stehen nur wenige Farben zur Verfügung (begrenzter Framebuffer)
  - Typisch: 8 Bit (256 Werte)
  - Werte werden als Indizes in eine Farbtabelle genutzt, die je 8 Bit R-, G-, B-Werte angeben.

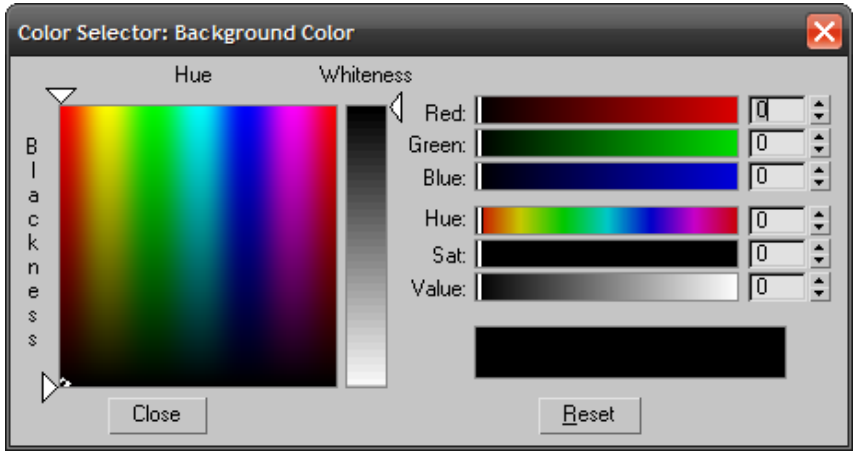


Farbspezifikation durch den Benutzer

- Auswahl aus einem Menü (Palette)
  - nur sinnvoll bei geringer Farbanzahl
  - Farben auf kleinen Flächen schwer zu erkennen
- Namentliche Nennung („gelblich-grün“, „blaugrau“)
  - mehrdeutig und subjektiv
  - Abhilfe: Color Naming Scheme (CNS, international standardisiert).

Farbspezifikation durch den Benutzer

- Koordinatenangaben in einem Farbraum
  - textuell
  - Rollbalken
  - Interaktion mit graphischer Darstellung des Farbmodells



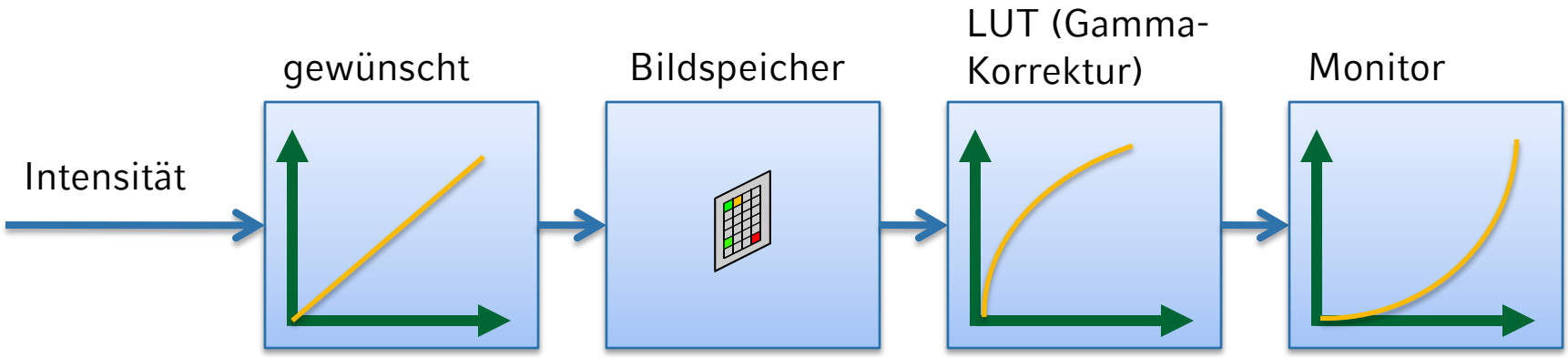
Gamma

- visuelles System reagiert nicht-linear auf Intensitätsänderungen
- Nichtlineare Reaktion des Elektronenstrahls im Monitor auf Spannungsänderungen
- Intensität des „Lichts“ am Monitor ist abhängig von der Eingangsspannung des Elektronenstrahls: $I = \text{constant}(V)\gamma$
- Eingangsspannung für eine gewünschte Intensität (für Farbmonitore typischerweise: $2{,}3 \leq \gamma \leq 2{,}6$):

$$V = \left(\frac{I}{\text{constant}} \right)^{\frac{1}{\gamma}}$$

Gamma und Gammakorrektur

- Kompensation für die Nicht-Linearität des Elektronenstrahls
- Reaktion des visuellen Systems und Reaktion des Elektronenstrahls sind nahezu invers zueinander.
- Gammakorrektur kalibriert nur die Intensität der Anzeige, nicht die Farben.



Ein kleiner Test zum Abschluss



- Aufgabe: Sagen Sie die Farbe, mit dem der Text gefüllt ist, so schnell vor wie möglich!
- Konflikt, linke Gehirnhälfte versucht, Farbe zu sagen und rechte Gehirnhälfte versucht, das Wort zu sagen

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

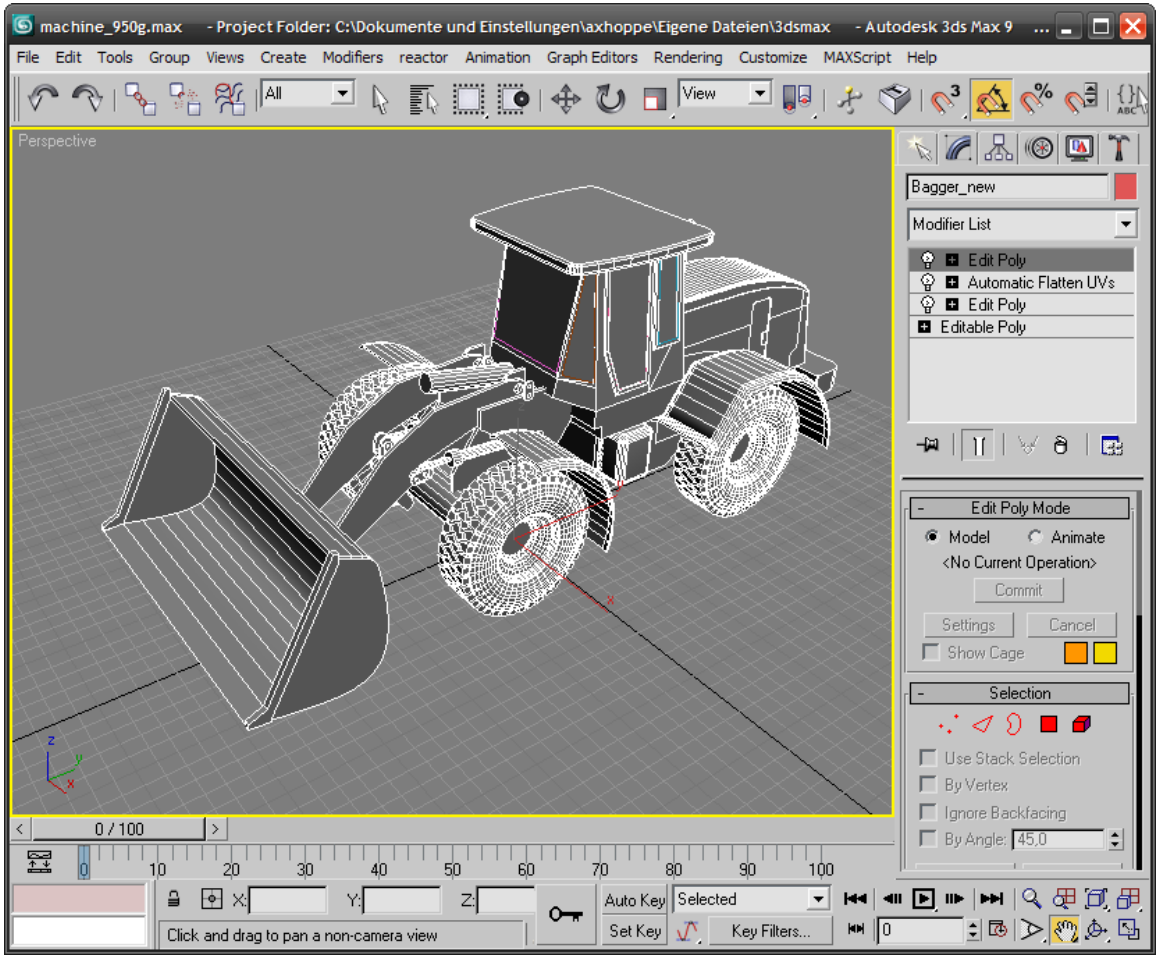
Vom Modell zum Bild

Datenstrukturen für die Erstellung von Szenen

Zusammenbau der Szene

- Bisher sind Objekte in *lokalen* Koordinaten gegeben, d.h. jedes Objekt hat sein eigenes Koordinatensystem.
- Berechnungen, die mehrere Objekte einbeziehen, sind schwierig
- daher ist Transformation in ein gemeinsames Koordinatensystem notwendig
- *Weltkoordinaten:*
- Platzieren der Objekte in einem globalen Koordinatensystem, in dem auch Kameras und Lichter sowie Oberflächeneigenschaften der Objekte definiert werden

Zusammenbau der Szene

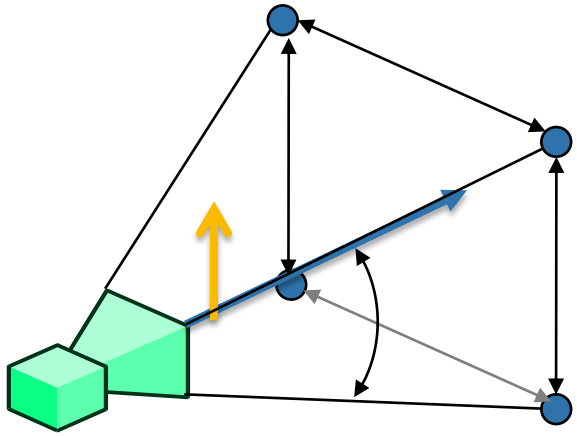


Zusammenbau der Szene

- Wie entsteht ein größeres geometrisches Modell?
- Durch explizite Modellierung verschiedener Teile (oft durch mehrere Personen parallel durchgeführt)
- Durch extensive Wiederverwendung von Teilen (auch aus anderen Modellen)
- Wiederverwendung durch Kopieren, Verschieben, Rotieren von Modellteilen (Beispiele: Räder und Türgriffe an Autos, Fenster, Türen an Gebäuden, etc.)
- Repräsentation großer Modelle:
  - Mehrfach verwendete Geometrie oft nur einmalig repräsentiert; zusammen mit mehreren Transformationen

Definition von Kameras und Lichtquellen

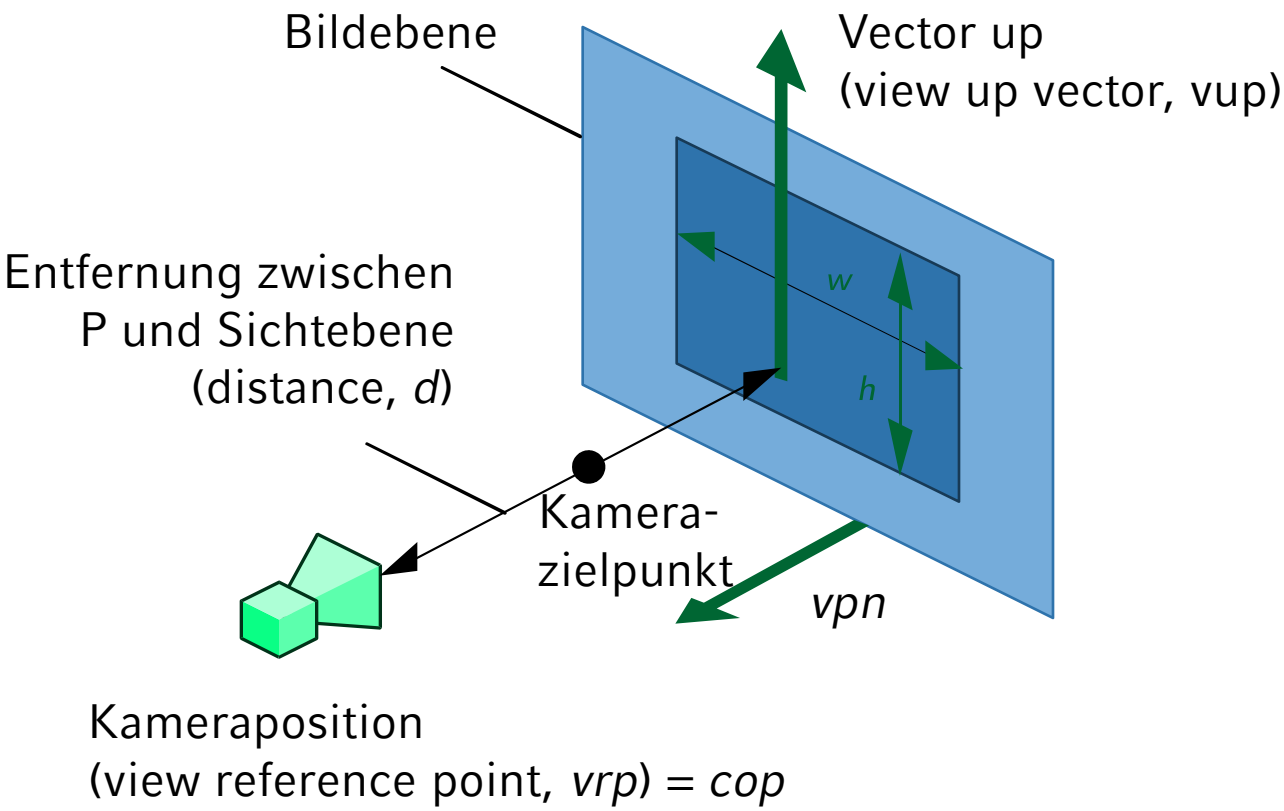
- Kameramodell:
  - Position und Richtung
  - die Abbildung bestimmende Parameter



Kameramodell

- Reale Kamera als Orientierung und Metapher für den Viewing-Prozess
- Parameter realer Kameras:
  - Position
  - Blickrichtung
  - Brennweite
  - Tiefenschärfe
  - Art des Objektivs (Weitwinkel, Tele) bestimmt den Öffnungswinkel → Projektionen sind teilweise nicht planar
- Parameter der entstehenden Bilder:
  - Größe (Breite, Höhe, bzw. Breite und Seitenverhältnis)
  - Typische Werte für Seitenverhältnis: 4:3 (Fernsehen), 16:9
  - Auflösung

Synthetische Kamera

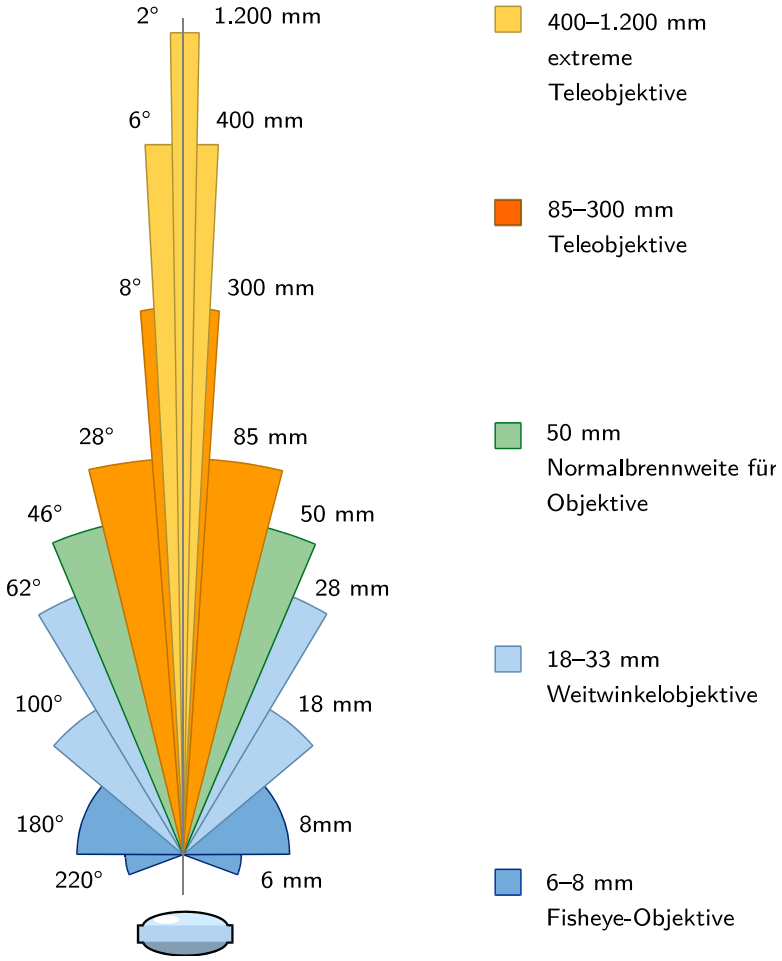


Kameramodell

Öffnungswinkel	Brennweite
76,0°	24 mm
67,6°	28 mm
56,4°	35 mm
41,4°	50 mm
24,9°	85 mm
15,8°	135 mm
10,7°	200 mm

Kameramodell

- Zusammenhang zwischen
  - Öffnungswinkel und
  - Brennweite



Kameramodell

- In der Computergraphik wird ein vereinfachtes Kameramodell genutzt:
  - Position – Punkt (Koordinaten in 3D)
  - Blickrichtung – Vektor (Koordinaten in 3D)
- Bildspezifikation:
  - Größe des Viewports in der Sichtebene (Pixel)
- Zusätzlich:
  - Schnittebenen, die die dargestellte Szene auf solche Objekte begrenzen, die hinter einer ersten Ebene (*near*) und vor einer zweiten (*far*) liegen.
    - Spezifikation als Abstände von der Kamera in Blickrichtung.
    - Objekte, die eine Ebene schneiden, müssen geclippt werden.

Kamerakoordinatensystem

- Koordinatensystem, bei dem die Kamera im Ursprung steht und entlang der (negativen) z-Achse ausgerichtet ist.
- Definition des Sichtkörpers
- Einige Operationen lassen sich hier sehr einfach ausführen, daher dieser „Zwischenschritt“
  - Optimierungen (Entfernen verdeckter Rückseiten)
  - Clipping gegen den Sichtkörper

Kamerakoordinatensystem

- Abbildung von Weltkoordinaten (des Modells) in Kamerakoordinaten: Model-View-Matrix.
- Matrix in homogenen Koordinaten, wird mit Projektionsmatrix multipliziert.

Definition von Lichtquellen (LQ)

- Üblich: Punktlichtquellen (LQ ohne Ausdehnung, die in alle Richtungen gleichstark abstrahlen) – vereinfachtes Modell
  - Position
  - Farbe des ausgestrahlten Lichtes
  - Helligkeit
- Weitere Typen von Lichtquellen möglich:
  - Flächige LQ
  - Gerichtete LQ (parallele Strahlen in eine Richtung)
  - Spotlights (Gerichtete LQ mit Öffnungswinkel)

Quellen

- Axel Hoppe. *Validierung und Nachbearbeitung von gerenderten Bildern.* Dissertation an der Otto-von Guericke-Universität Magdeburg, Shaker-Verlag, Aachen, 1999. ISBN 3-8265-4610-5
- Foley, van Dam, Feiner, Hughes. *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim. *Computergraphik 1*. Universität Magdeburg, Vorlesungsskript, Juli 2005.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Computergrafik I

Datenstrukturen für die 3D-Computergrafik

Rückschau

- Ausgabegeräte für die Computergrafik
- Erinnerung: alle relevanten Ausgabegeräte arbeiten rasterorientiert

Fahrplan für heute

- Vom Modell zum Bild: Vorüberlegungen für Geometrierepräsentation
- Rendering-Pipeline
- Transformationen in 2D
- Transformationen in 3D

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Vorüberlegungen

Vom Modell zum Bild

3D-Bildberechnung

- Gegeben:
  - möglichst genaues 3D-Modell in geeigneten Koordinaten,
  - Beschreibung für Oberflächen und Lichtverhältnisse sowie
  - eine Beschreibung der Sicht in die Szene
- Gesucht:
  - gerastertes (Pixel-)Bild, das das gegebene Modell realitätsnah darstellt.

3D-Szene

- für eine Szenenbeschreibung werden benötigt:
  1. Objektrepräsentation
  2. Beschreibungsmöglichkeit für Oberflächenanmutungen
  3. Modell für Interaktion der Oberflächen mit Lichtquellen
  4. Modell für ein Sichtbeschreibung in die 3D-Szene
  5. Möglichkeit, Objekte anzuordnen

Objekt-Repräsentation

1. Mächtigkeit (sinnvolle Menge von Objekten modellierbar)
2. Eindeutigkeit (Was wird repräsentiert?)
3. Genauigkeit (Approximation des Objektes)
4. Effizienz (Darstellung, Speicherplatz)
5. Abgeschlossenheit (Transformationen, Boolesche Mengenoperationen)
6. weitere Forderungen:
   - unmöglich, eine ungültige Repräsentation herzustellen
   - einfach, eine gültige Repräsentation herzustellen

Objekt-Repräsentation

- Volumenmodelle
  - Primitive Instancing
  - Constructive Solid Geometry
  - Sweep-Körper
  - Raumunterteilung
- Oberflächenmodelle
  - Polygone
  - Freiformflächen
  - Quadriken

Objekt-Repräsentation – Volumina

Kennzeichnung

- Unterscheidung zwischen innen/außen und Grenzflächen eines Modells häufig notwendig
- Modellierung der inneren Struktur eines Objektes
- Modellierung von Objekten als Volumina

Anwendungsgebiete

- CAD/CAM
- physikalische Simulation
- medizinische Visualisierung, etwa Operationsplanung

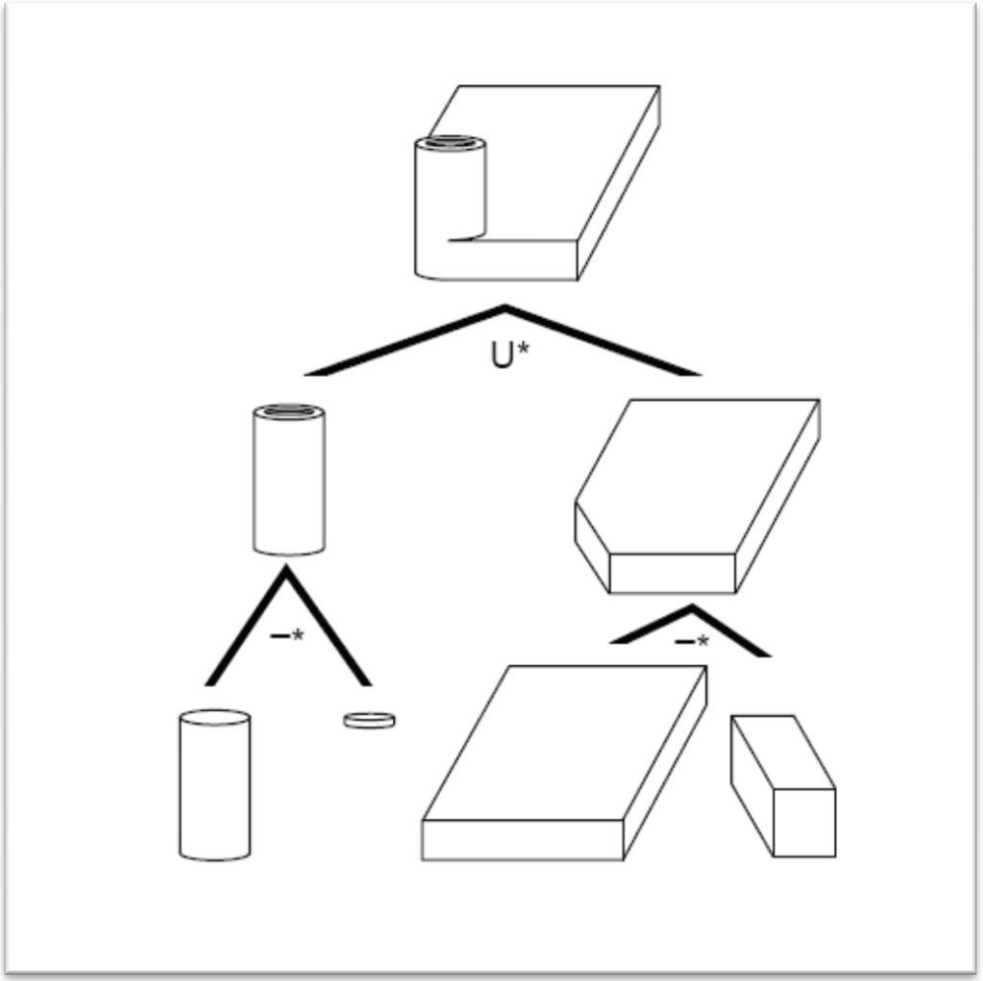
Primitive Instancing

- Modelliersystem definiert Menge von primitiven 3D-Volumina
  - anwendungsbezogen
  - Parametrisierung dieser Primitive
  - Komposition über Transformationen
- Eigenschaften
  - Zusammenbau zu „Baugruppen“
  - keine Kombination der Objektvolumina zu Gesamtvolumina

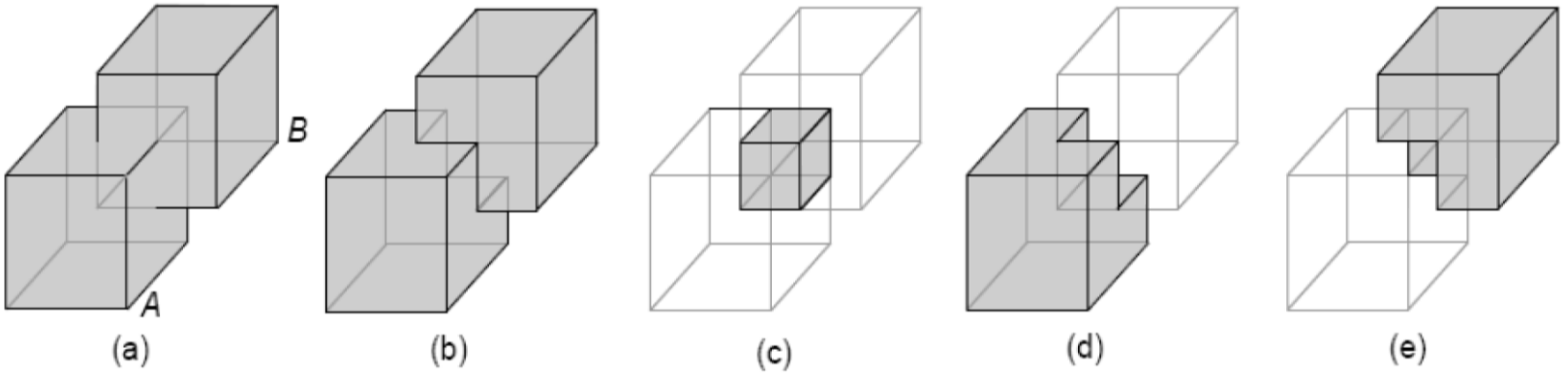
Constructive Solid Geometry (CSG)

- Herstellung komplexer Volumina über boolesche Operationen
- Primitive: geometrische Grundobjekte (Pyramide, Quader, Kugel, etc.)
- Ein komplexes Objekt wird über einen Baum beschrieben:
  - Knoten beschreiben die Verknüpfung der Nachfolger
  - Teilbäume repräsentieren Teilobjekte
  - pro Knoten gibt es zwei Nachfolger
- Volumina der Teilkörper werden verbunden zu Gesamtvolumen

Constructive Solid Geometry (CSG)



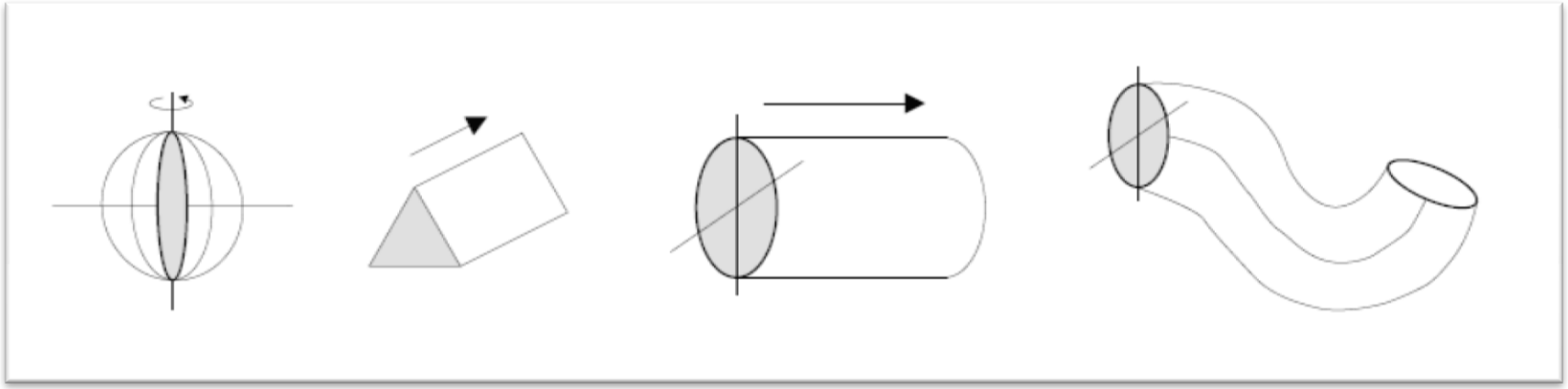
Boolsche Operationen



(b): $A \cup B$, (c): $A \cap B$, (d): $A - B$, (e): $B - A$

Sweep-Körper

- Objekt wird durch eine Fläche (Grundfläche) und eine Transformationsvorschrift beschrieben (=generalisierter Zylinder).
- problematisch:
  - Selbstüberschneidungen der Kurve (Was ist dann das Volumen?)
  - Kurve liegt in der Flächenebene (Leeres Volumen?)



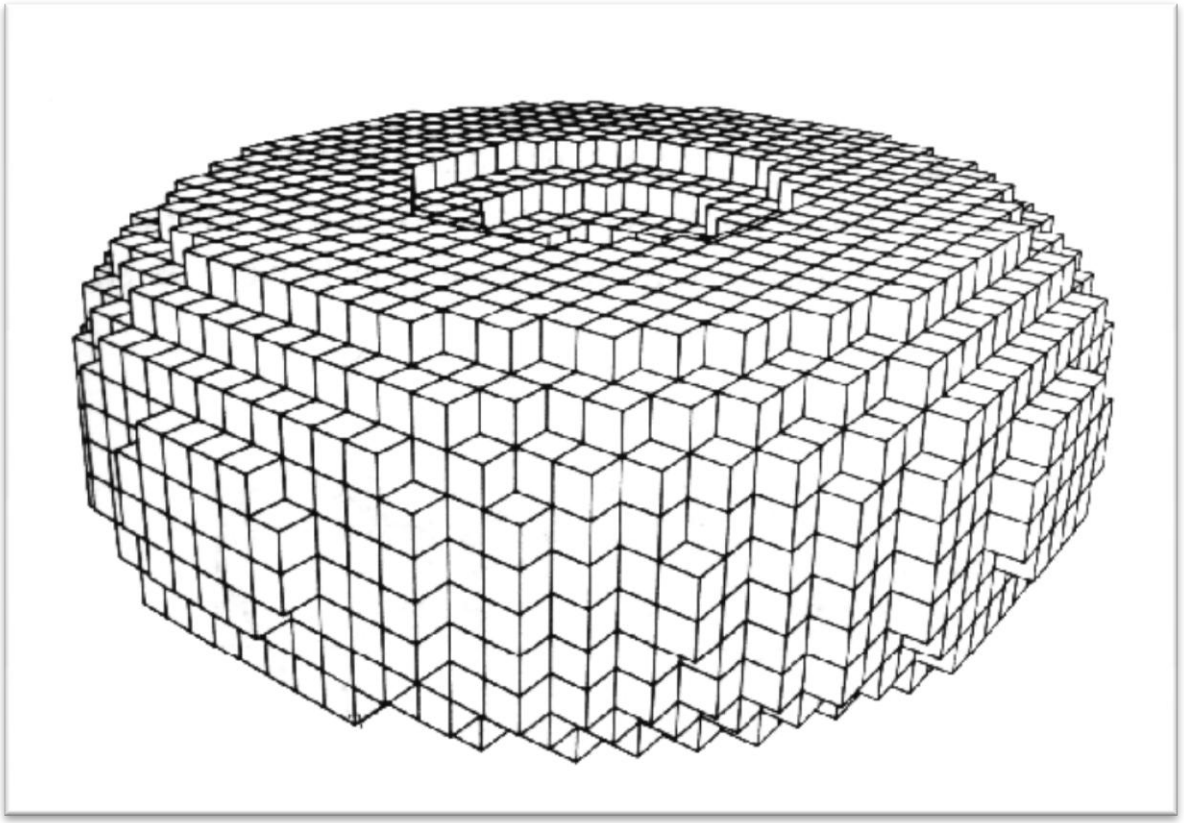
Raumunterteilung

- Unterteilung des Gesamtvolumens in Teilvolumina nach verschiedenen Strategien:
  - Zelldekomposition – Unterteilung in eine Menge unterschiedlicher Primitive
  - räumliche Aufzählung – Unterteilung in gleiche Voxel
  - Octrees – Unterteilung durch achsenparallele Ebenen
  - BSP-Trees – Unterteilung durch arbiträre Ebenen

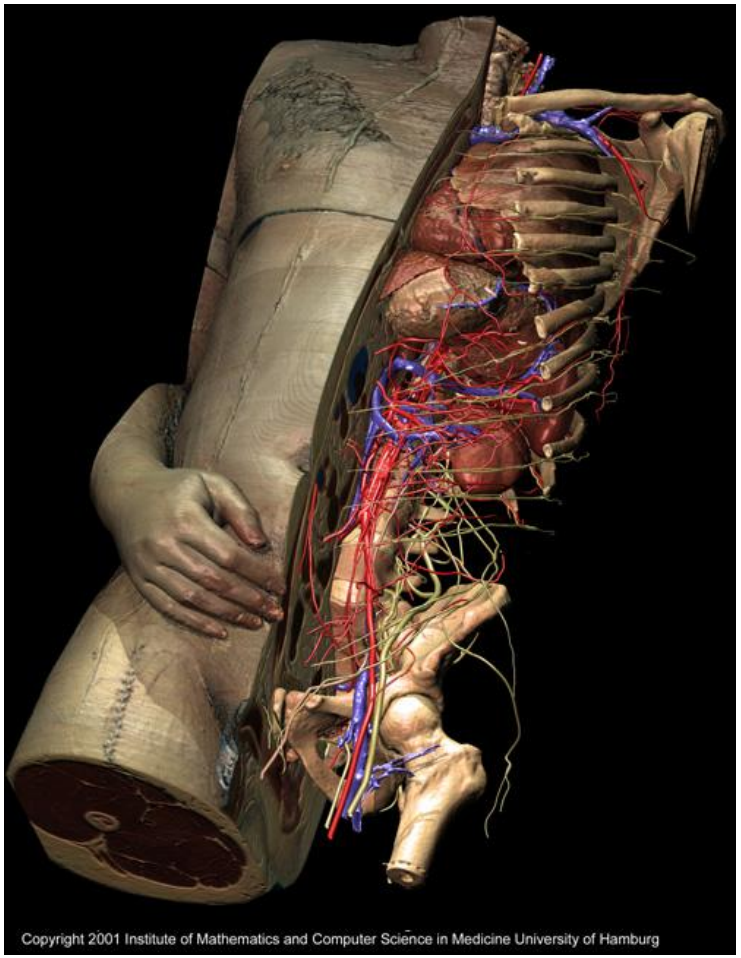
Voxel

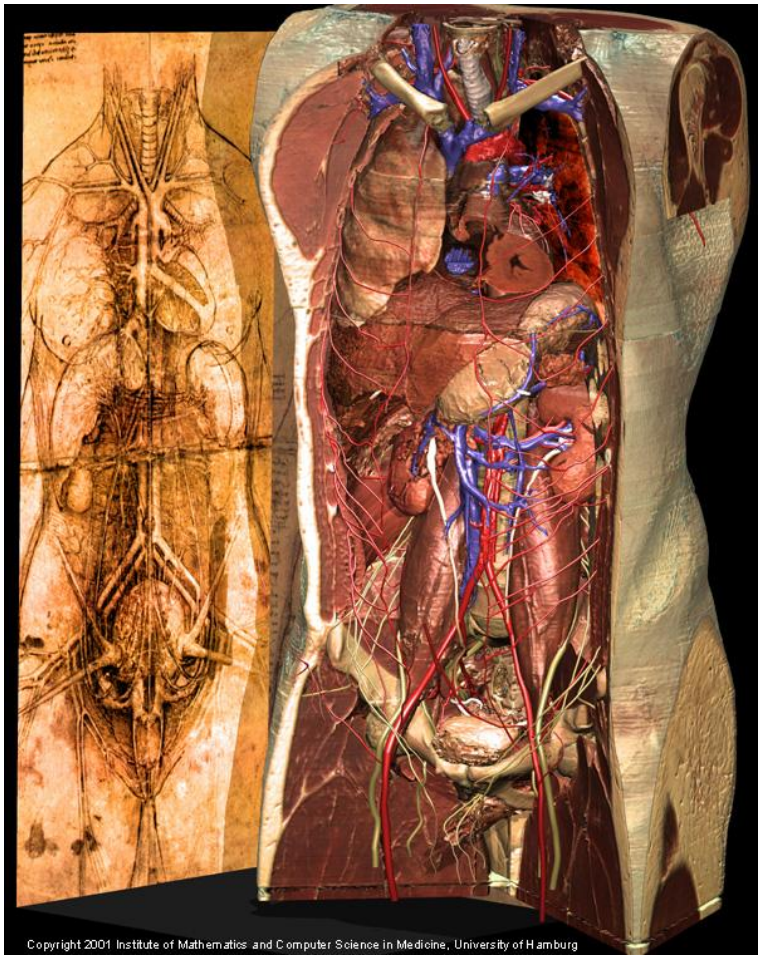
- Zelldekomposition in identische Zellen, die in einem regelmäßigen Gitter angeordnet sind
- *voxel* = *volume element*, dreidimensionale Pixel
- An- oder Abwesenheit des entsprechenden Voxels
- Möglichkeit der Speicherung weiterer Parameter
- Approximation führt zu Aliasing-Artefakten
- hoher Speicherplatzbedarf (n^3)

Voxel – Beispiel

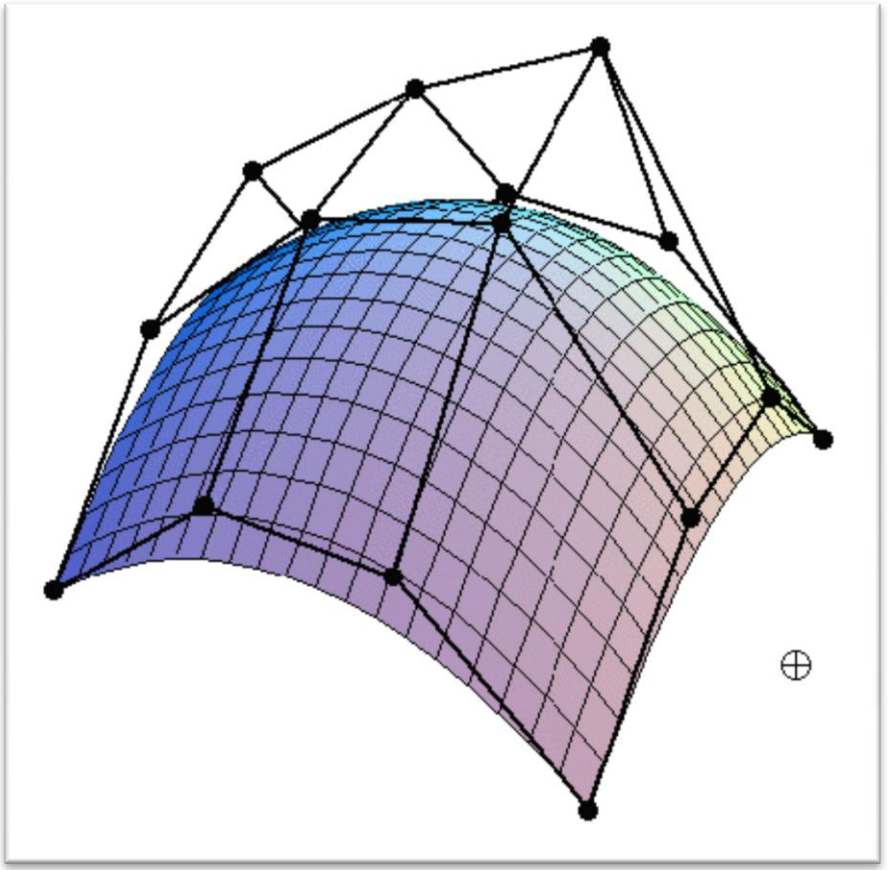


Voxel – Anwendungen





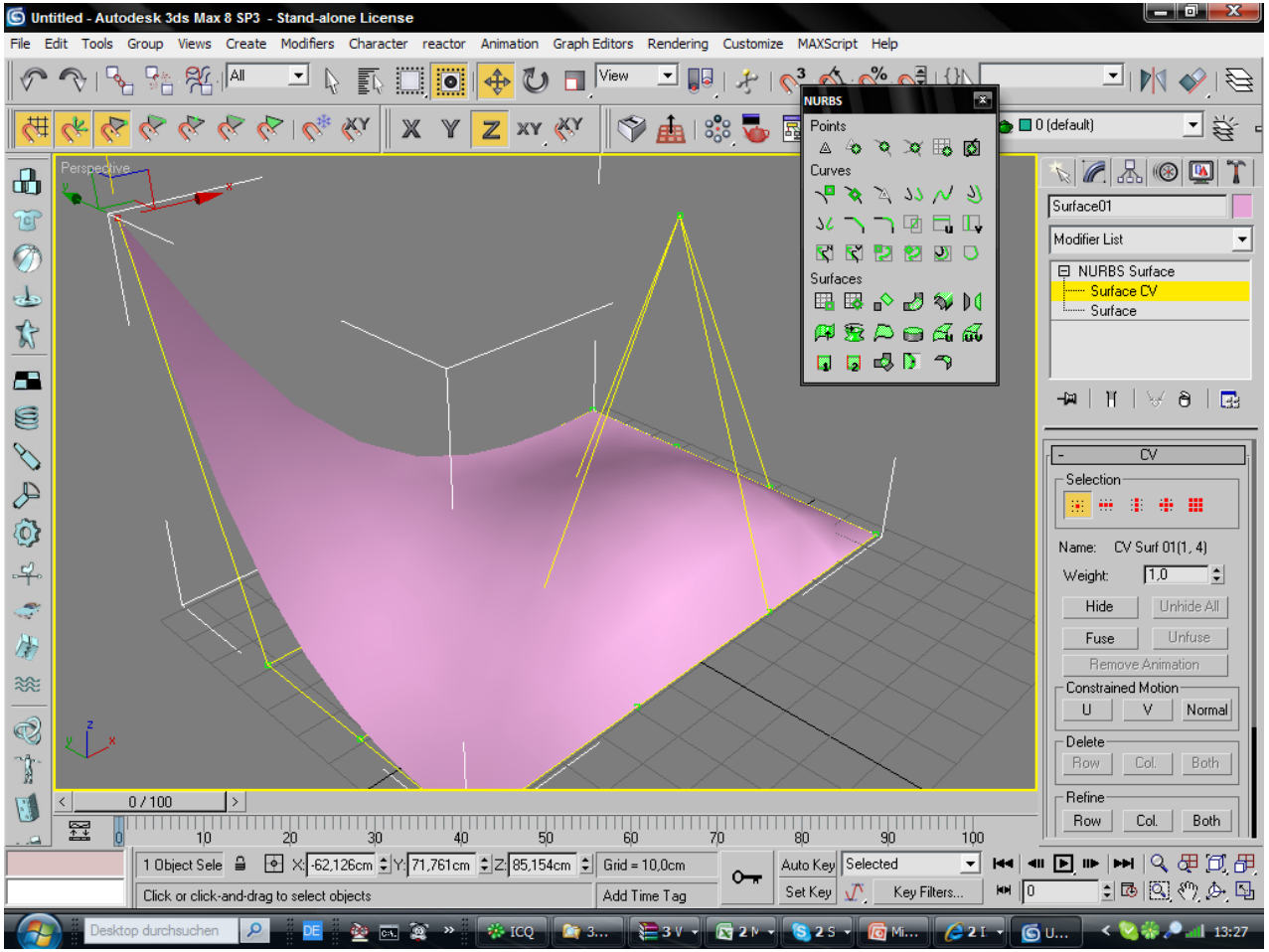
Freiformflächen



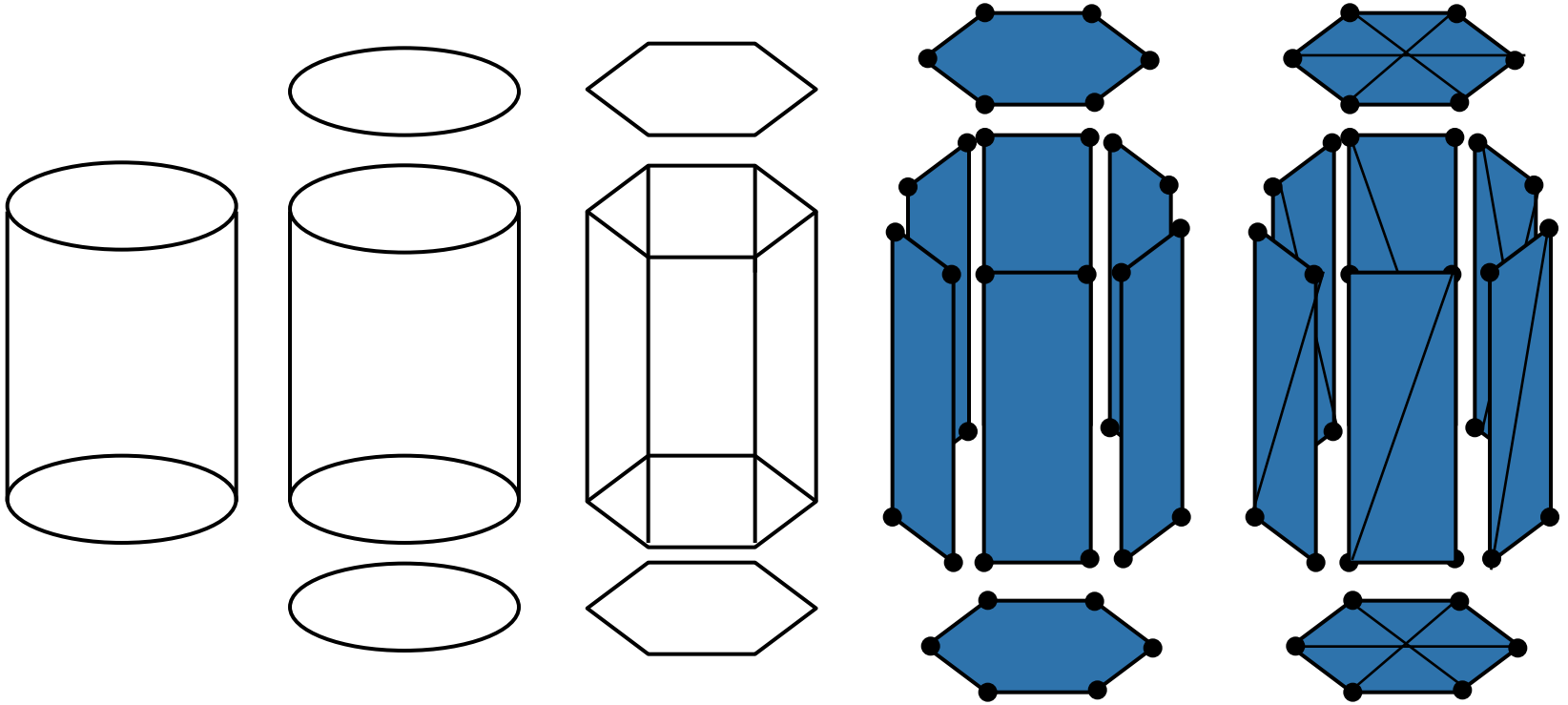
Freiformflächen – Eigenschaften

- sind invariant für projektive Transformationen
- gemeinsame mathematische Darstellung für sowohl analytische Standardformen (z. B. Kegelschnitte) als auch Freiformflächen
- reduzierter den Speicheraufwand für geometrische Objekte
- können durch numerisch stabile und präzise Algorithmen verhältnismäßig schnell ausgewertet werden
- sind Verallgemeinerungen von nicht-rationalen B-Splines und nicht-rationalen und rationalen Bézier-Kurven und -Flächen
- ermöglichen organische Anmutungen von Oberflächen

NURBS AUS SICHT DES ANWENDERS

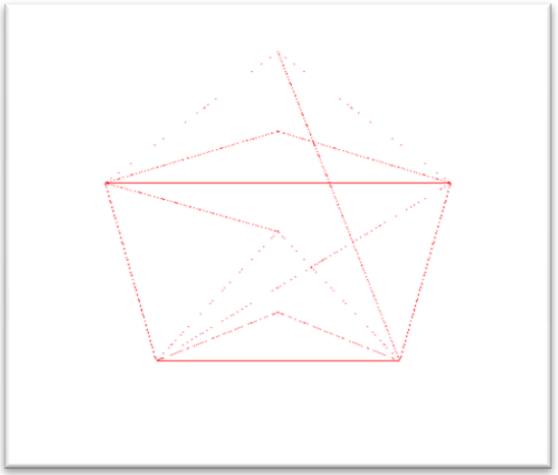
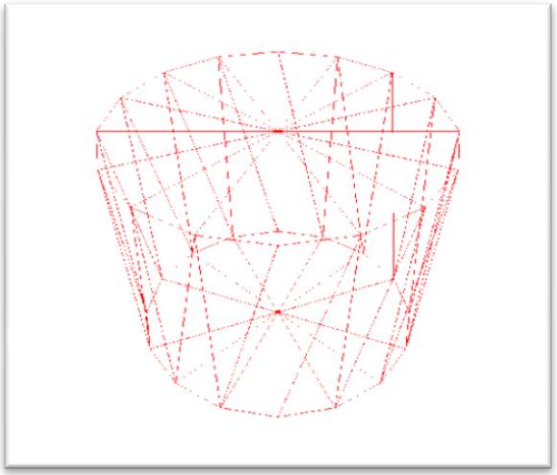
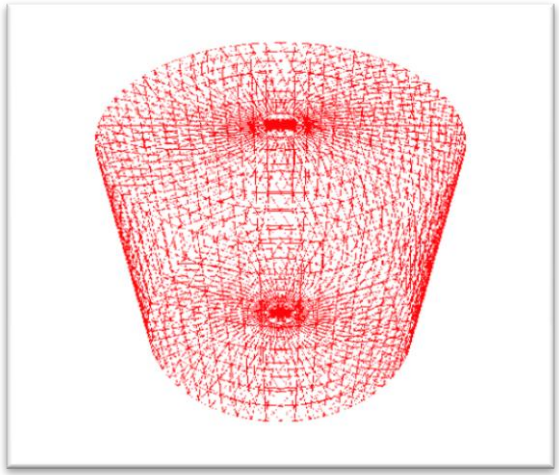


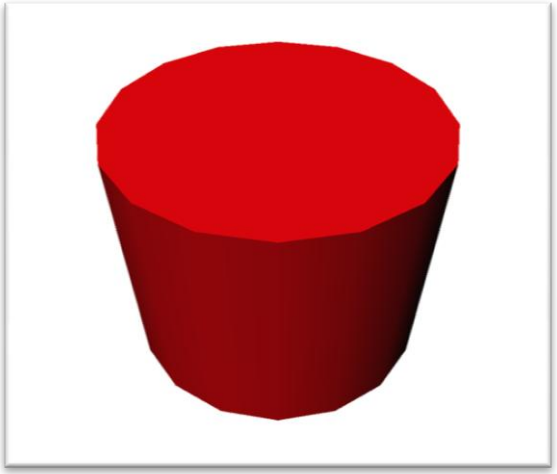
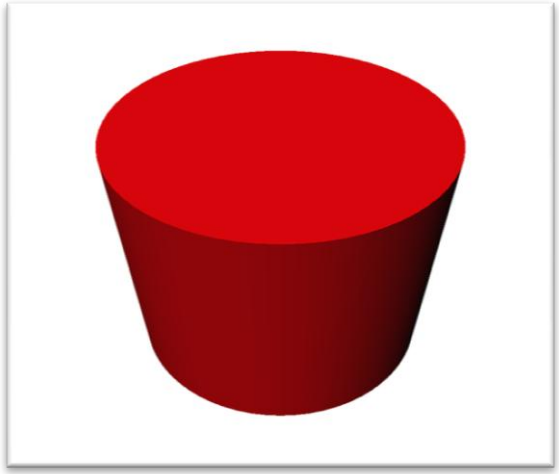
Polygone



ideales Objekt | Oberflächen | polygonale Oberfläche | Eckpunkte, Kanten für Polygone | Facetten (*Faces*)

Objekt-Repräsentation, Illustration

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

vom Modell zum Bild

Gestalterische Überlegungen

Ziel

- möglichst realistische, schöne Bilder
- gleiche Komplexität wie die komplexen Interaktionen zwischen Licht und Oberflächen in der Realität
- „Fotorealismus“
- Was ist der Ursprung der folgenden Bilder? Begründen Sie Ihre Aussage!

Real oder synthetisch?



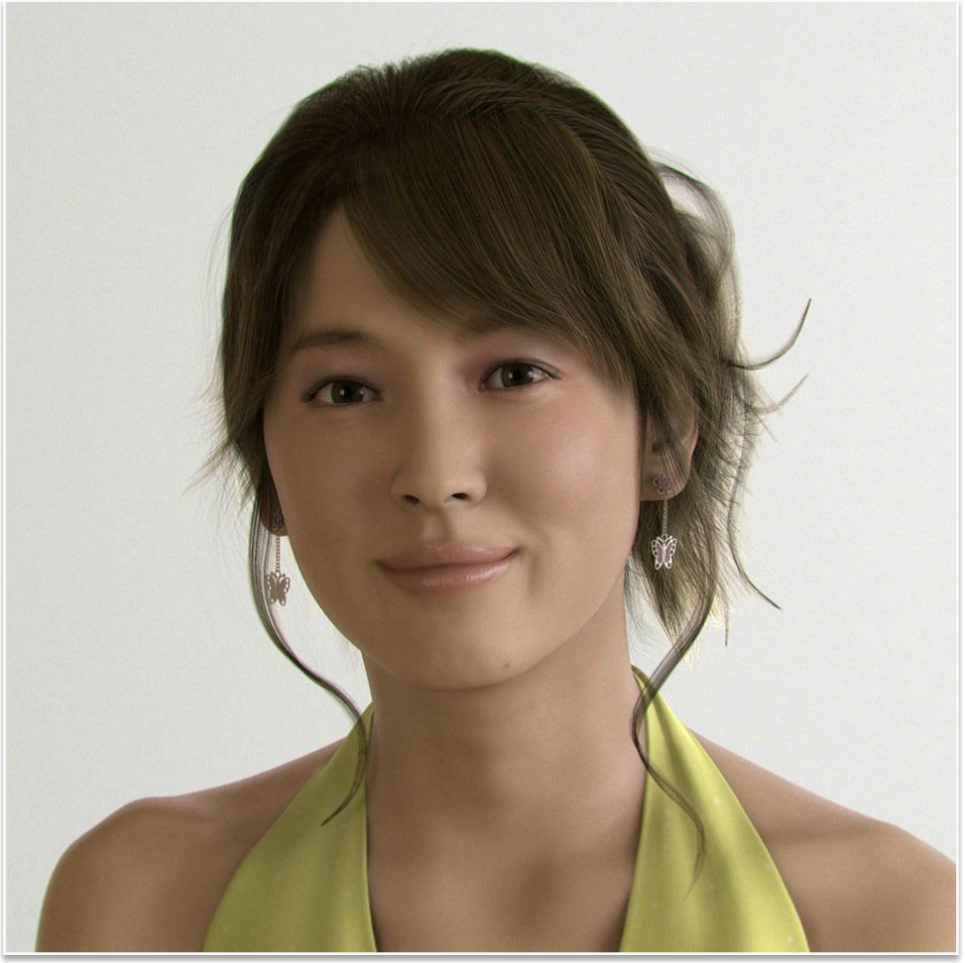
Real oder virtuell?



Real oder virtuell?



Real oder virtuell?



Real oder virtuell?



Objekt-Repräsentation

- Vorüberlegung: Ausreichend ist Betrachtung der Oberflächen
- daher Konzentration auf polygonale Oberflächenmodelle
- Analytische und implizite Darstellungen lassen sich zudem in explizite Darstellungen umwandeln (Polygonalisieren).

Vorteile polygonaler Modelle

- geometrischer Modelle
- Relativ einfache Algorithmen, daher Implementierung in Hardware möglich
- Wichtig: Polygone müssen in einer Ebene liegen (z. B. um zu bestimmen, was im Inneren liegt) – das ist bei Dreiecken immer der Fall
- Daher: (interne) Triangulierung

Zusammenfassung Polygone

Vorteile

- Einfache Darstellung der Objekte
- Einfache und einheitliche Handhabung bei Berechnungen
- Weit verbreitet – „kleinster gemeinsamer Nenner“ bei 3D-Modellen

Nachteile

- Polygonale Modelle approximieren die Oberfläche eines „runden“ Objektes.
- Je genauer diese Approximation sein soll, um so mehr Polygone werden benötigt – speicheraufwändig

Notation von Polygonen

- Polygone sind gekennzeichnet durch
  - Eckpunkte (engl. *vertex*, Plural *vertices*)
  - Kanten (engl. *edges*)
  - Merken: Polygone werden zusammengesetzt aus Facetten (engl. *faces*)

Notation von Polygonen

- Angabe der Koordinaten der Eckpunkte als Koordinatentripel
- Verbindung der Eckpunkte zu einem Polygon wird implizit hergestellt:
  - Verbinde den i-ten mit dem $(i + 1)$-ten Punkt
  - Verbinde den letzten mit dem ersten Punkt zum Schließen des Polygons

Notation von Polygonen

- Polygone werden nach 2D abgebildet
- Für weitere Betrachtungen reicht also zunächst:
  - Betrachtung 2D-Primitiven
  - Erörtern der Aufgaben der Computergrafik im 2D-Fall

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Bildberechnung

Rendering-Pipeline

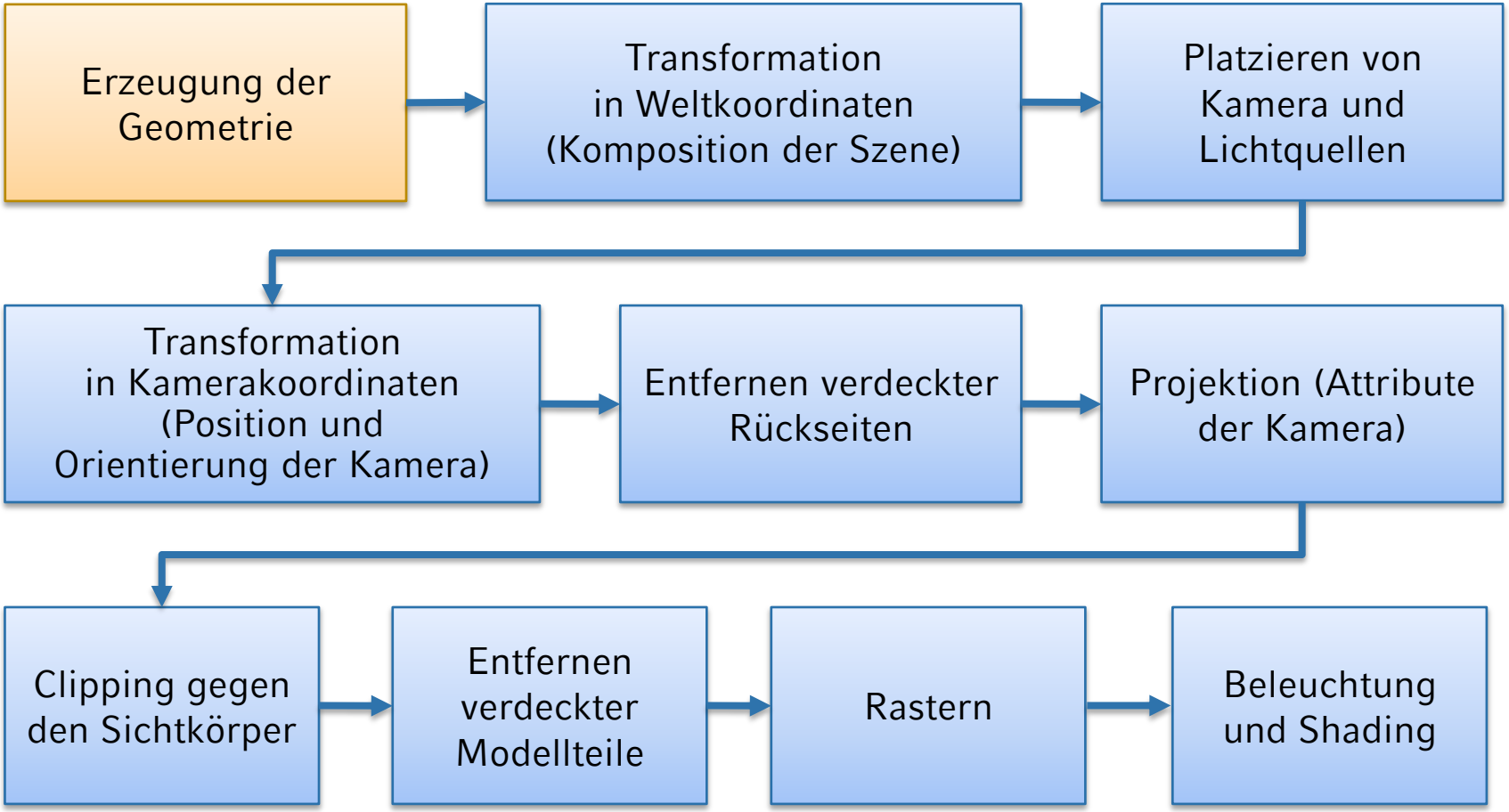
3D-Bildberechnung

- Ziel: Integration von einzelnen Techniken in ein System zur Bilderzeugung aus einem 3D-Modell
- Große Anteile:
  - Viewing,
  - Shading,
  - Rasterisierung
- Rendering von engl. *to render* = Bilderzeugung

3D-Bildberechnung

- Gegeben:
  - 3D-Modell in geeigneten Koordinaten,
  - Beschreibung für Oberflächen und Lichtverhältnisse sowie
  - eine Beschreibung der Sicht in die Szene
- Gesucht:
  - gerastertes (Pixel-)Bild, das das gegebene Modell realitätsnah darstellt.

Rendering-Pipeline



LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Spezifikationen in 2D und 3D

Koordinatensysteme

Vektoren und Vektorräume

- Vektor – Definition: Ein Vektor ist ein Element eines Vektorraumes.
- Vektorraum – Definition: Eine nichtleere Menge V heißt Vektorraum über einen fest gegebenen Körper K, falls gilt
  - auf V gibt es eine Verknüpfung +, so dass V mit dieser Verknüpfung eine abelsche Gruppe ist, d. h. für alle x, y, z aus V gilt:
    - $(x + y) + z = x + (y + z)$
    - Es gibt einen Nullvektor o mit: $x + o = x = o + x$.
    - Zu jedem x aus V existiert ein $-x$ aus V mit $x + (-x) = o$.
    - $x + y = y + x$
  - es gibt eine Skalarmultiplikation, d. h. eine Abbildung $K \times V \to V$, so dass für alle x, y aus V und alle r, s aus K gilt:
    - $1v = v$.
    - $r(v + w) = rv + rw$.
    - $(r + s)v = rv + sv$.
    - $(rs)v = r(sv)$.

Vektoren und Vektorräume

- Beispiele für Vektorräume:
  - Menge aller Polynome mit Koeffizienten aus K
  - Menge aller $n \times n$-Matrizen mit reellen Elementen
  - Menge aller geordneten Paare (x, y) (x, y reelle Zahlen)
  - Menge aller geordneten Tripel (x, y, z) (x, y, z reelle Zahlen)
  - Menge aller geordneten n-Tupel $(x_0, x_1, \ldots x_n)$ (x_i reelle Zahlen)
- besonders die letzten drei Beispiele für Computergraphik interessant $\rightarrow$ reellwertige Koordinaten in 2D, 3D und nD

Vektorraum $\Re^2$

- V ist die Menge aller geordneter Paare (x, y) mit $x, y \in \Re$
- K ist der Körper der reellen Zahlen
- Nullelement in V: (0, 0)
- besondere Elemente in V: (1, 0) und (0, 1)
  - Basis des Vektorraumes $\Re^2$
  - = maximale Menge linear unabhängiger Elemente aus V
- Verknüpfung „+“ entspricht komponentenweiser Addition $(x, y) + (a, b) = (x + a, y + b)$
- Skalarmultiplikation entspricht komponentenweiser Multiplikation mit dem Skalar $\alpha(x, y) = (\alpha x, \alpha y)$

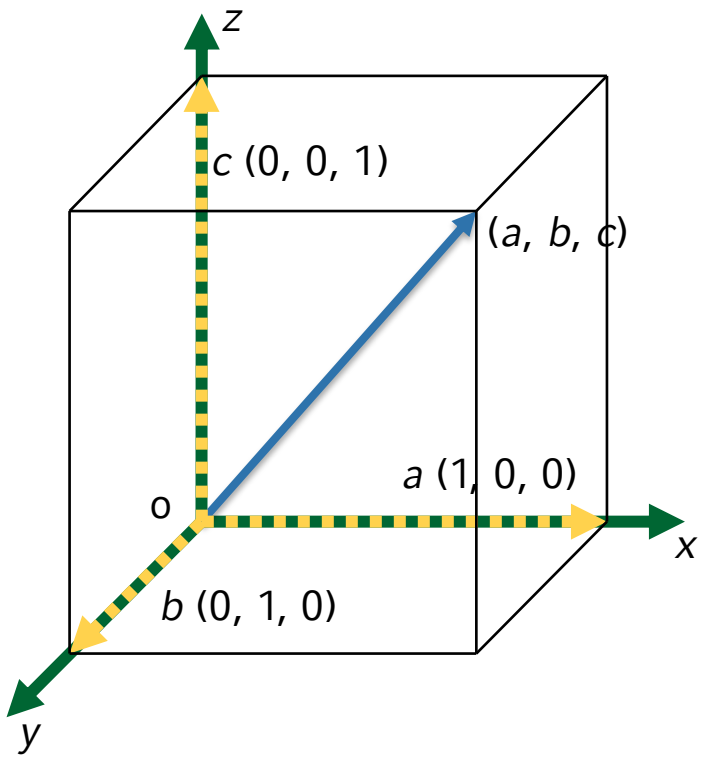
Vektorraum $\mathfrak{R}^3$

- V ist die Menge aller geordneter Tripel (x, y, z) mit $x, y, z \in \mathfrak{R}$
- K ist der Körper der reellen Zahlen
- Nullelement in V: (0, 0, 0)
- besondere Elemente in V: (1, 0, 0), (0, 1, 0) und (0, 0, 1)
  - Basis des Vektorraumes $\mathfrak{R}^3$
  - = maximale Menge linear unabhängiger Elemente aus V
- Verknüpfung „+“ entspricht komponentenweiser Addition $(x, y, z) + (a, b, c) = (x + a, y + b, z + c)$
- Skalarmultiplikation entspricht komponentenweiser Multiplikation mit dem Skalar $\alpha(x, y, z) = (\alpha x, \alpha y, \alpha z)$

Punkte und „Vektoren“

- Geometrische Interpretation der beiden letzten Beispiele führt zu Punkten und Vektoren, wie sie aus der Geometrie bekannt sind.
- Jeder Vektor (a, b, c) kann eindeutig in eine Linearkombination der Elemente der Basis des Vektorraumes zerlegt werden:
  - $(a, b, c) = a\,(1, 0, 0) + b\,(0, 1, 0) + c\,(0, 0, 1)$
  - Die Skalare a, b und c sind die kartesischen Koordinaten des Vektors im System der Einheitsvektoren des Koordinatensystems.
  - Die kartesischen Koordinaten eines Vektors sind die Projektionen dieses Vektors auf die einzelnen Koordinatenachsen.

Koordinaten



- Interpretation als Vektor:
  - Ein Vektor selbst hat keine Position.
  - Ausgehend von einem festen Punkt (z. B. o) definiert ein Vektor einen Punkt.
  - Vektor (a, b, c) kann als Punkt im Raum dargestellt werden, der dem Endpunkt eines Vektors (a, b, c) ausgehend vom Koordinatenursprung (0, 0, 0) entspricht.

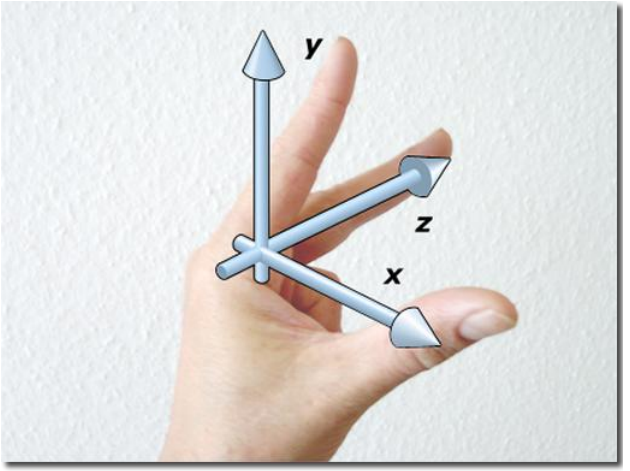
Koordinatensystem

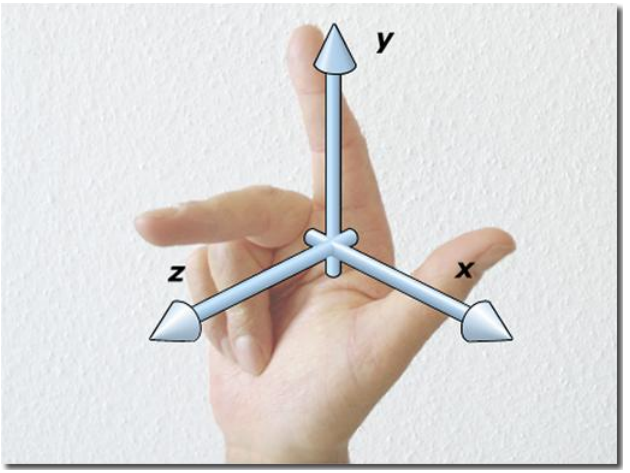
- Eine Menge (o, e_1, e_2, ..., e_n) bestehend aus einem Punkt o $\in A^n$ und der Basis (e_1, e_2, ..., e_n) von A^n heißt Koordinatensystem
- Punkt o heißt Koordinatenursprung
- Für jeden Punkt $p \in A^n$ ist $v = (op)$ Ortsvektor von p
- Komponenten von v heißen Koordinaten bezüglich (e_1, e_2, ..., e_n) d. h. p besitzt die Koordinaten (x_1, x_2, ..., x_n):

$$\left(\overrightarrow{op}\right) = v = x_1 e_1 + x_2 e_2 + \ldots + x_n e_n$$

Koordinatensysteme

- X – Richtung des Daumens
- Y – Zeigefinger
- Z – Mittelfinger
- Die beiden Koordinatensysteme sind spiegelbildlich und nicht durch Drehung ineinander zu überführen.





LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Spezifikationen in 2D und 3D

Transformationen

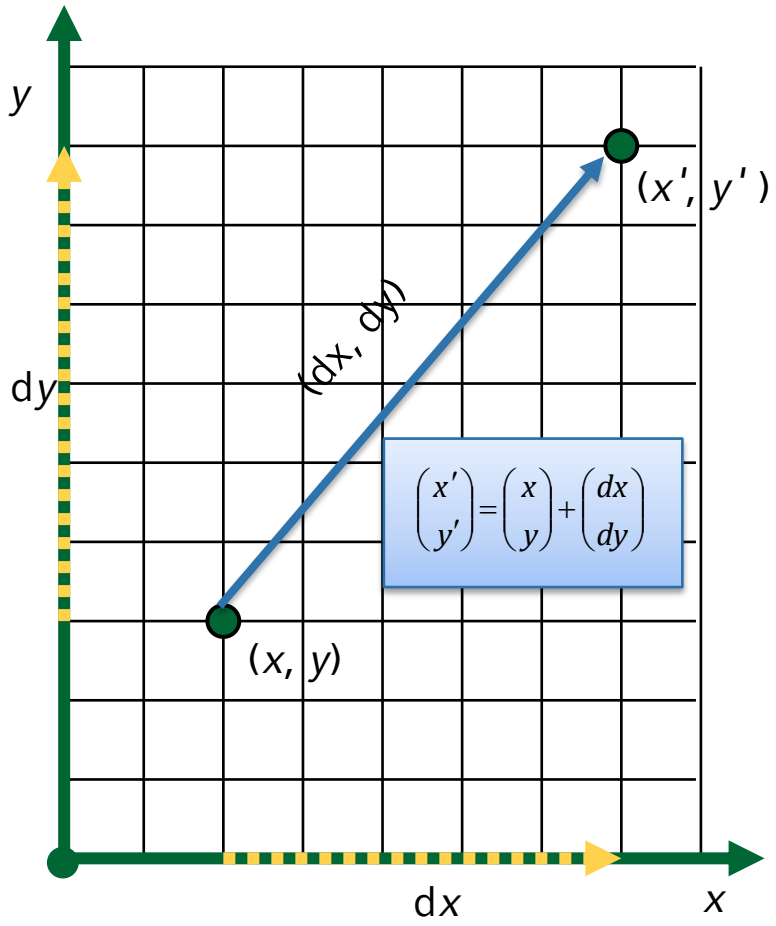
Koordinatensysteme

- Wie werden Bewegungen beschrieben?
- Wie wird die Position von Objekten nach Bewegungen berechnet?
- zunächst Betrachtung von Transformationen in 2D

Koordinatensysteme

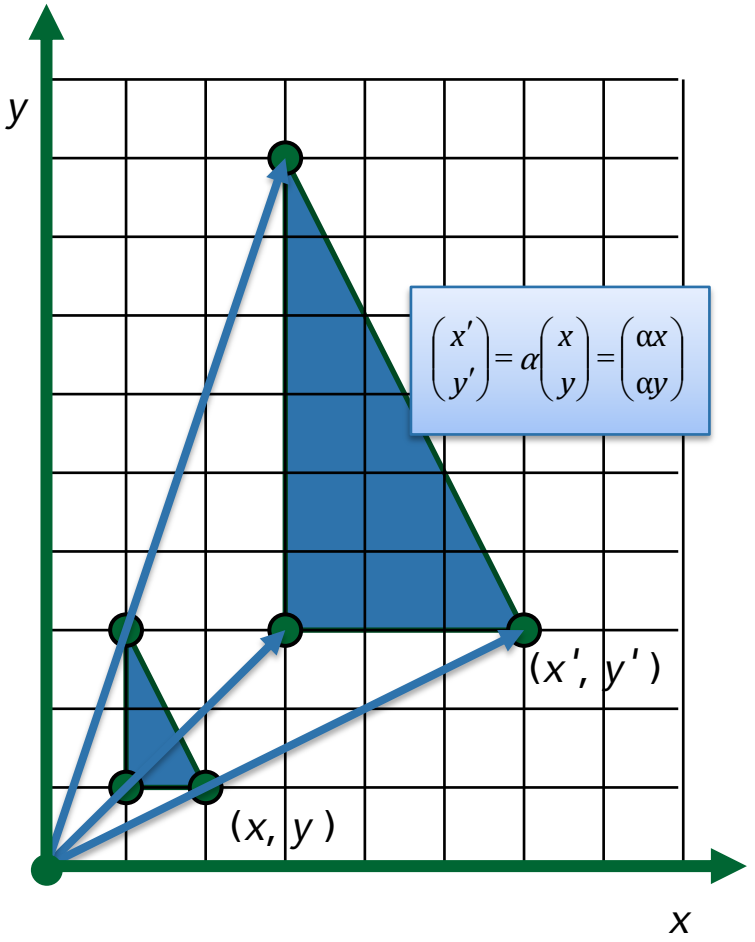
- Bewegungen = Transformationen
  - Veränderung der Position von Punkten
  - Verschiebung = Translation
  - Größenveränderungen = Skalierung
  - Drehung = Rotation
  - Weitere affine Transformationen:
    - Spiegelung
    - Scherung

Translation



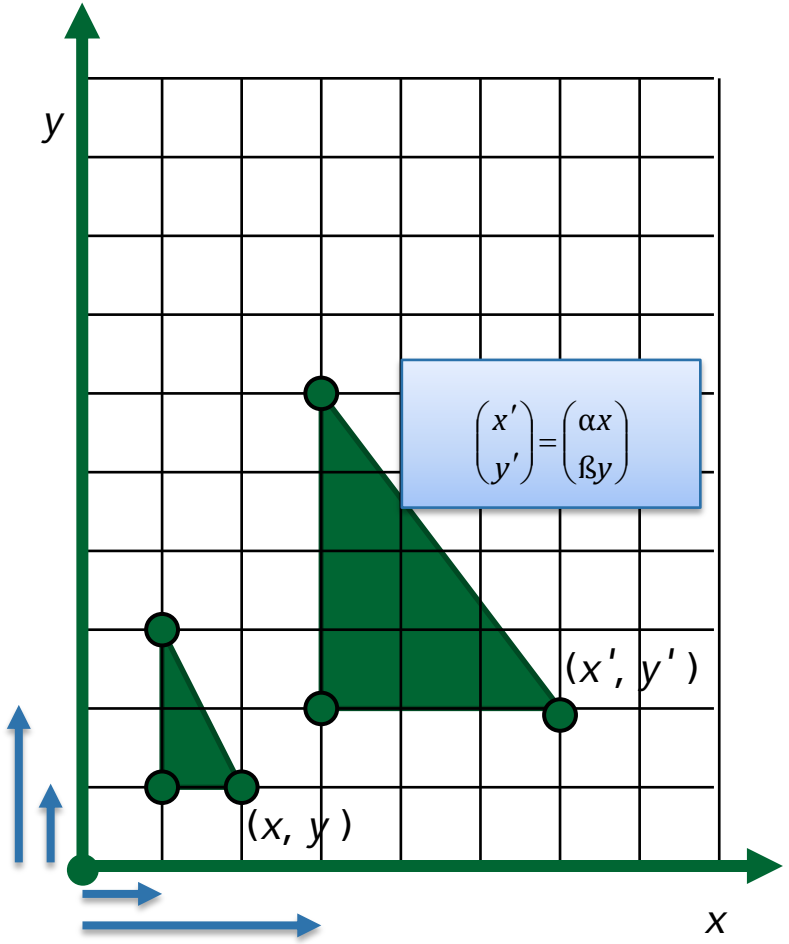
- Punkt (x, y) wird auf gerade Linie nach (x', y') verschoben.
- Addition des Verschiebungsvektors
- Beschreibung der Translation durch einen Vektor (dx, dy), der die Verschiebungsweite in x- und y-Richtung angibt

Uniforme Skalierung



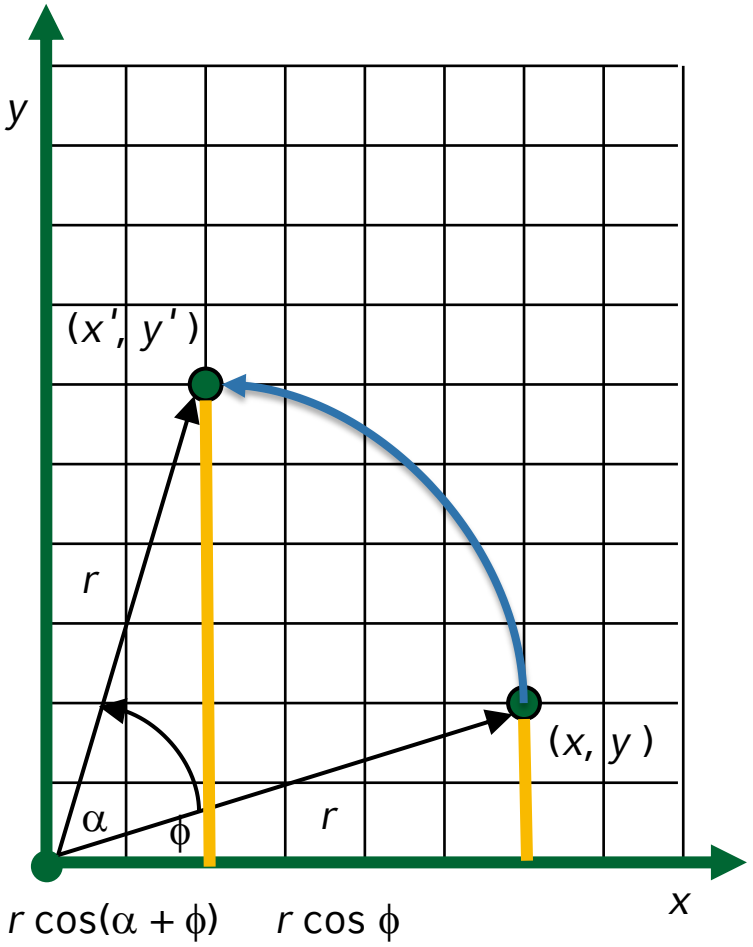
- Zentrum der Skalierung ist o, Skalierung erfolgt in alle Richtungen uniform mit dem skalaren Faktor α
- Multiplikation mit dem Skalierungsfaktor
- Ortsvektor zu (x, y) wird auf das α-fache verlängert, um (x', y') zu erhalten

Nicht-uniforme Skalierung



- Zentrum der Skalierung ist o, Skalierung erfolgt in *x*-Richtung mit dem Faktor α, in *y*-Richtung mit β (Skalierungsvektor $(\alpha, \beta)^T$)
- Multiplikation mit entsprechenden Skalierungsfaktoren
- Ortsvektor zu (*x*, *y*) wird auf das α-fache in *x*-Richtung und das β-fache in *y*-Richtung verlängert.

Rotation



- Rotationszentrum ist o.
- Positive Werte von α ergeben eine Drehung entgegen dem Uhrzeigersinn.
- Punkt (x, y) wird um den Winkel α um o gedreht, so dass sich der Punkt (x', y') ergibt.

Rotation: Herleitung der Berechnungs-Vorschrift

I. $x = r \cos \phi$

II. $y = r \sin \phi$

III. $x' = r \cos (\alpha + \phi) = r \cos \phi \cos \alpha - r \sin \phi \sin \alpha$

IV. $y' = r \sin (\alpha + \phi) = r \cos \phi \sin \alpha - r \sin \phi \cos \alpha$

- anschließend (I) in (IV) einsetzen:

$x' = x \cos \alpha - y \sin \alpha$

$y' = x \sin \alpha + y \cos \alpha$

Rotation: Herleitung der Berechnungs-Vorschrift

- Rotationen um negative Winkel erfolgen mit dem Uhrzeigersinn; ausnutzen:
- $\cos(-\alpha) = \cos(\alpha)$ und $\sin(-\alpha) = -\sin(\alpha)$
- die Berechnungs-Vorschrift

$$x' = x \cos \alpha - y \sin \alpha$$

$$y' = x \sin \alpha + y \cos \alpha$$

- kann als Matrix-Vektormultiplikation ausgedrückt werden:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Kritische Rückschau

- Bisherige Betrachtung:
  - Translation: Addition des Verschiebungsvektors
  - Skalierung: Multiplikation des Skalierungsfaktors
  - Rotation: Matrixmultiplikation
- **Fazit:**
  - Hierbei erfolgt keine einheitliche Behandlung
  - Es ergeben sich Schwierigkeiten bei zusammengesetzten Transformationen
- **Gesucht:** Einheitliche Repräsentation von Transformationen

Homogene Koordinaten

- Ein Koordinatensystem wird in ein homogenes Koordinatensystem überführt, indem eine zusätzliche Dimension eingeführt wird: $n \rightarrow n + 1$ Dimensionen.
- Ein Punkt (x, y, z) wird in homogenen Koordinaten durch das Tripel $(x \cdot w, y \cdot w, w)$ repräsentiert, mit $w \neq 0$.
- Normalisierte Darstellung: $w = 1 \Rightarrow (x, y, 1)$
- Jeder Punkt hat unendlich viele äquivalente Repräsentationen in homogenen Koordinaten.

Vorteile

- Repräsentation aller Punkte in homogenen Koordinaten ermöglicht einheitliche Behandlung der Transformationen

Vorteile

- Fragen:
  - Was steht für das Fragezeichen?
  - Welcher Art ist die Operation ist „*“?
- Antwort:
  - Transformationen werden als Matrizen repräsentiert
  - Verknüpfung durch Multiplikation

$$\begin{pmatrix} wx' \\ wy' \\ w \end{pmatrix} = ?* \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Homogene Koordinaten

- **Translation**
  - Vorher: Addition eines Vektors
  - Jetzt: Translationsmatrix

$$\begin{pmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- **Skalierung**
  - Vorher: Multiplikation mit Skalierungsfaktoren
  - Jetzt: Skalierungsmatrix

$$\begin{pmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Homogene Koordinaten

- **Rotation**
  - Vorher: komplexe Gleichung oder Matrixmultiplikation
  - Jetzt: Rotationsmatrix

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \Rightarrow$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Homogene Koordinaten

- Allgemeine 2D-Transformations-Matrix

$$T = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix}$$

Skalierung

Rotation

Translation

Inverse Transformationen

- Frage: Wie macht man Transformationen wieder rückgängig, also was sind die entsprechenden inversen Transformationen?
- für die elementaren Transformationen relativ einfach:
  - Translation: Verschiebung um den negativen Verschiebungsvektor $T^{-1}(dx, dy) = T(-dx, -dy)$
  - Skalierung: Skalierung mit dem reziproken Skalierungsfaktor
    - $S^{-1}(\alpha) = S(1/\alpha)$
  - Rotation: Rotation um den negativen Rotationswinkel. Da aber Rotationsmatrizen speziell orthogonal sind, gilt hier $R^{-1} = R^T$ und keine Neubesetzung der Matrix ist notwendig.

Zusammengesetzte Transformationen

- Nacheinander-Ausführung zweier Translationen
- Translation ist additiv, d. h. Ergebnis ist eine Verschiebung um die Summe der beiden Vektoren

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & dx_2 \\ 0 & 1 & dy_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & dx_1 \\ 0 & 1 & dy_1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & dx_1 + dx_2 \\ 0 & 1 & dy_1 + dx_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Zusammengesetzte Transformationen

- Nacheinander-Ausführung zweier Skalierungen
- Skalierung ist multiplikativ, d. h. Ergebnis ist eine Skalierung um das Produkt der beiden Faktoren

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} sx_2 & 0 & 0 \\ 0 & sy_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} sx_1 & 0 & 0 \\ 0 & sy_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} sx_1 \cdot sx_2 & 0 & 0 \\ 0 & sy_1 \cdot sy_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Zusammengesetzte Transformationen

- Nacheinander-Ausführung zweier Rotationen
- Rotation ist additiv

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha_2 & -\sin\alpha_2 & 0 \\ \sin\alpha_2 & \cos\alpha_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha_1 & -\sin\alpha_1 & 0 \\ \sin\alpha_1 & \cos\alpha_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

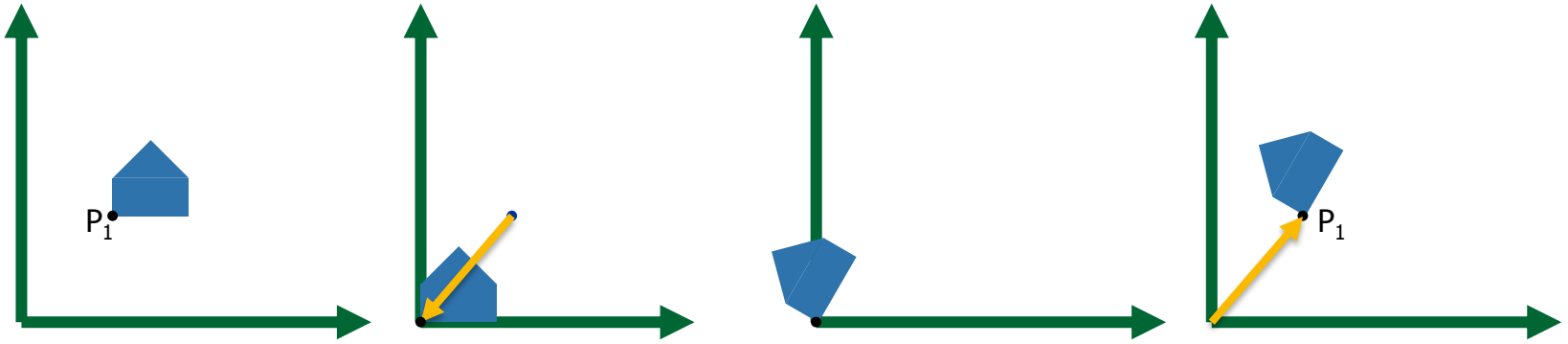
Zusammengesetzte Transformationen

- Allgemein: Arbeit mit Transformationsmatrizen
  - vereinheitlicht alle Transformationen
  - einfache Möglichkeit der Kombination von Transformationen
- hier genutzte Schreibweise
  - Transformationen werden in der Reihenfolge T_1, T_2, ..., T_n ausgeführt $\Rightarrow P' = T_n \cdot \ldots \cdot T_2 \cdot T_1 \cdot P$

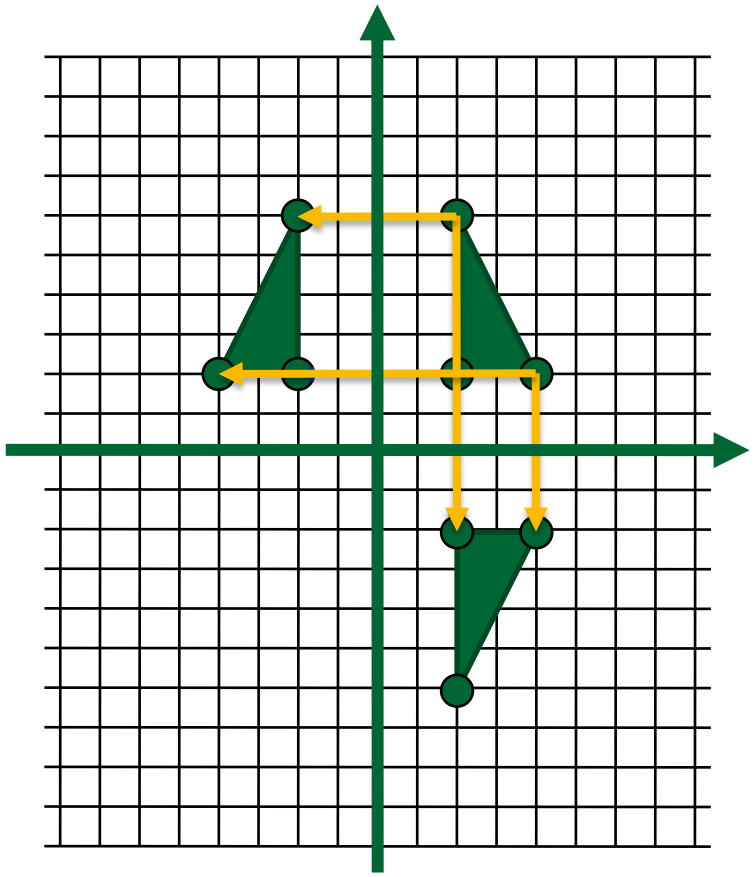
Zusammensetzen beliebiger Transformationen

- Rotation eines Punktes um einen *beliebigen* Punkt P_1 in der Ebene, Ausführung in drei Schritten:

  1. Translation, so dass P_1 im Ursprung liegt
  2. Rotation um den Ursprung
  3. Rück-Translation des Ursprungs nach P_1



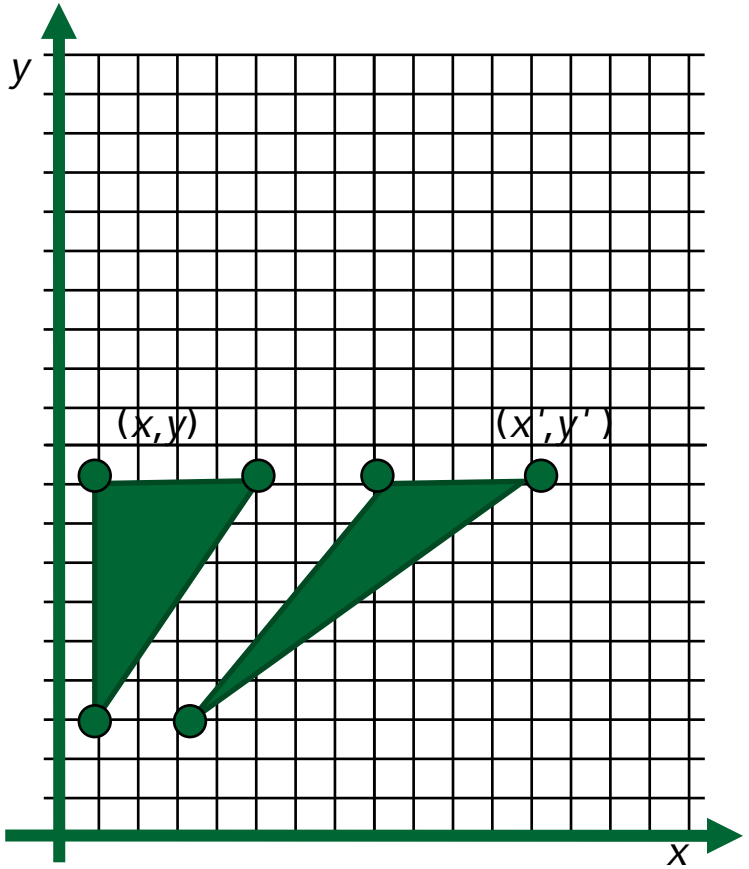
Weitere Transformationen: Spiegelung



- zwei Standard-Möglichkeiten:
  - Spiegelung an der x-Achse
  - Spiegelung an der y-Achse
- Spiegelung wird implementiert als Skalierung mit dem Skalierungs-Faktor −1

$$T = \begin{pmatrix} (-1) & 0 & 0 \\ 0 & (-1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Scherung



- Versatz parallel zur *x*-Achse, proportional zur *y*-Position (bzw. umgekehrt)
- zwei Standardmöglichkeiten:
  - Scherung entlang der *x*-Richtung
  - Scherung entlang der *y*-Richtung

$$T = \begin{pmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad T = \begin{pmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Spezifikationen in 3D

Transformationen in 3D

Dreidimensionale Transformationen

- Vorgehensweise gleich zu 2D
- Arbeit mit homogenen Koordinaten
- homogene Koordinaten sind jetzt vierdimensional
- Transformationsmatrizen demzufolge 4×4-Matrizen
- Anwendung wie in 2D

Transformationsmatrizen: Translation

- Addition eines Translationsvektors bzw.
- Multiplikation mit einer Translationsmatrix

$$T = \begin{pmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Skalierung

- Multiplikation mit Skalierungsfaktoren bzw. Multiplikation mit einer Skalierungsmatrix
- uniforme Skalierung, wenn $sx = sy = sz$, sonst
- nichtuniforme Skalierung

$$S = \begin{pmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation

- nicht mehr ganz trivial, da jetzt Rotationen um die verschiedenen Koordinatenachsen betrachtet werden müssen
- drei verschiedene Rotationsmatrizen
- Achse, um die gedreht wird, bleibt „Einheitsvektor“ in der Matrix

$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_y = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_z = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Konvertierung

- Überführung des rechtshändigen in das linkshändige Koordinatensystem
- Konvertierung:

$$T_{r\to l} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Homogene Koordinaten

- Zusammensetzen von Transformationen
- auch über Multiplikation der Matrizen
- generelle Transformationsmatrix in 3D

$$T = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix}$$

Skalierung

Rotation

Translation

Zusammenfassung Koordinaten

- Einheitliche Behandlung der Transformationen durch Übergang zu homogenen Koordinaten und zur Darstellung der Transformationen durch Matrizen
- Zusammengesetzte Transformationen durch Hintereinanderausführen von elementaren Transformationen, entspricht Multiplikation der Matrizen
- Transformation der Objekte oder des Koordinaten-Systems

Unterschiedliche Koordinaten

- Bisher sind Objekte in **lokalen Koordinaten** gegeben, d. h. jedes Objekt hat sein eigenes Koordinatensystem
- Berechnungen, die mehrere Objekte einbeziehen, sind schwierig
- Transformation in ein gemeinsames Koordinatensystem notwendig
- **Weltkoordinaten**
- Platzieren der Objekte zu einem globalen Koordinatensystem

Zusammenfassung Transformationen

- Geometrische Transformationen sind Abbildungen vom $\mathfrak{R}n$ in den $\mathfrak{R}n$
  - von besonderem Interesse $\mathfrak{R}2 \rightarrow \mathfrak{R}2$ und $\mathfrak{R}3 \rightarrow \mathfrak{R}3$
- für Computergraphik relevant:
  - Translation
  - Skalierung
  - Rotation
  - (Scherung, Spiegelung)
- einheitliche Behandlung der Transformationen durch Übergang zu homogenen Koordinaten und zur Darstellung der Transformationen durch Matrizen

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

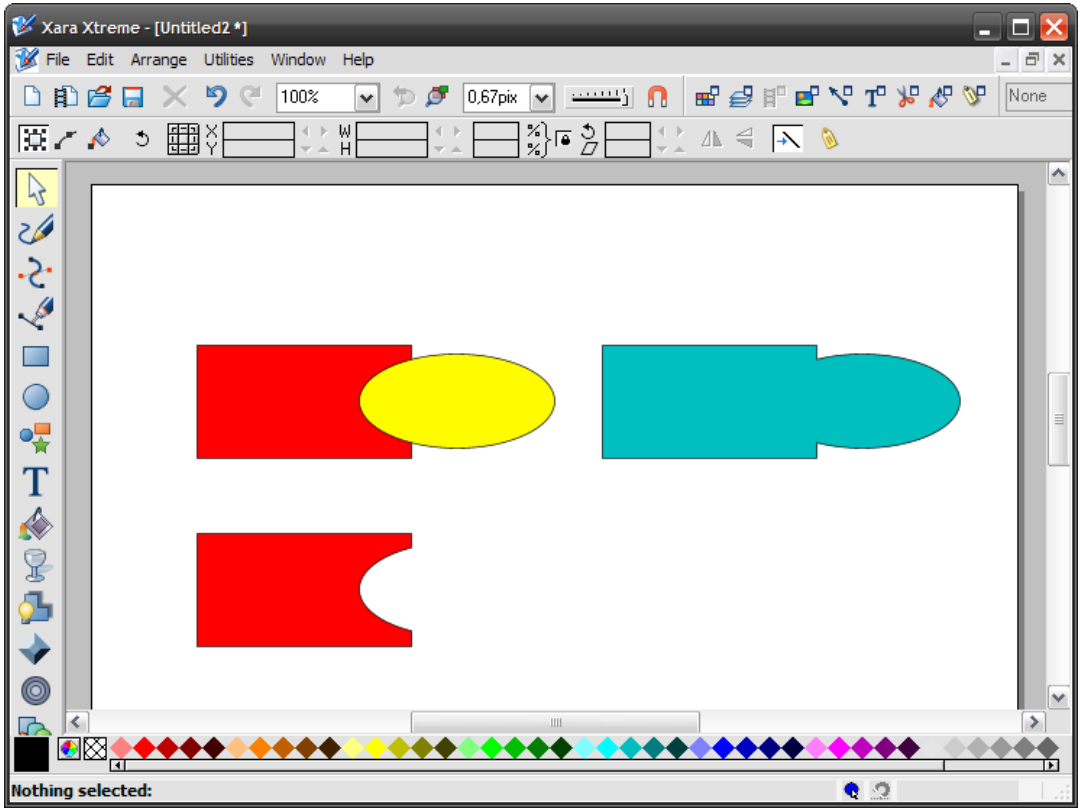
Spezifikationen in 2D und 3D

Transformationen für den Anwender

Arbeiten mit 2D-Primitiven

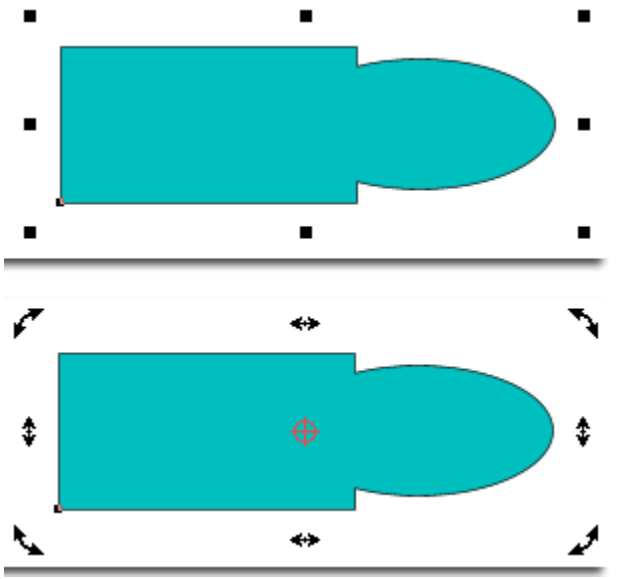
2D-Vektorgrafik: Szene wird u. a. erstellt, indem 2D-Primitive über boolesche Operationen zu neuen Formen erstellt werden

Formen können anschließend verlustfrei transformiert werden, um die 2D-Szene zu komponieren (= gemäß der Vision alle Objekte zueinander anzuordnen)



Arbeiten mit 2D-Primitiven

- Werkzeug: Xara
- Für das interaktive Ausführen von Transformationen existieren unterschiedliche Manipulatoren
- Translation per Drag & Drop
- Skalieren 1 × Klicken
- Rotieren, Scheren 2 × Klicken

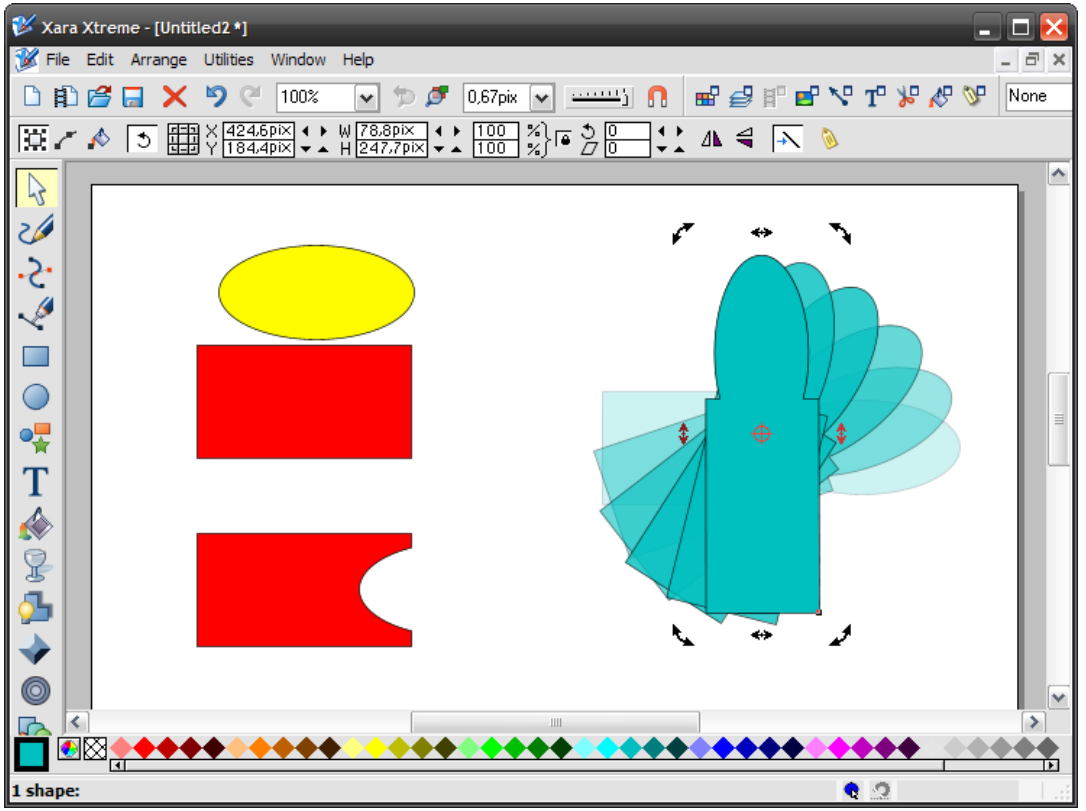


Arbeiten mit 2D-Primitiven

Rotation bezieht sich auf Bezugspunkt

dieser ist Ursprung eines lokalen Koordinatensystems pro Objekt

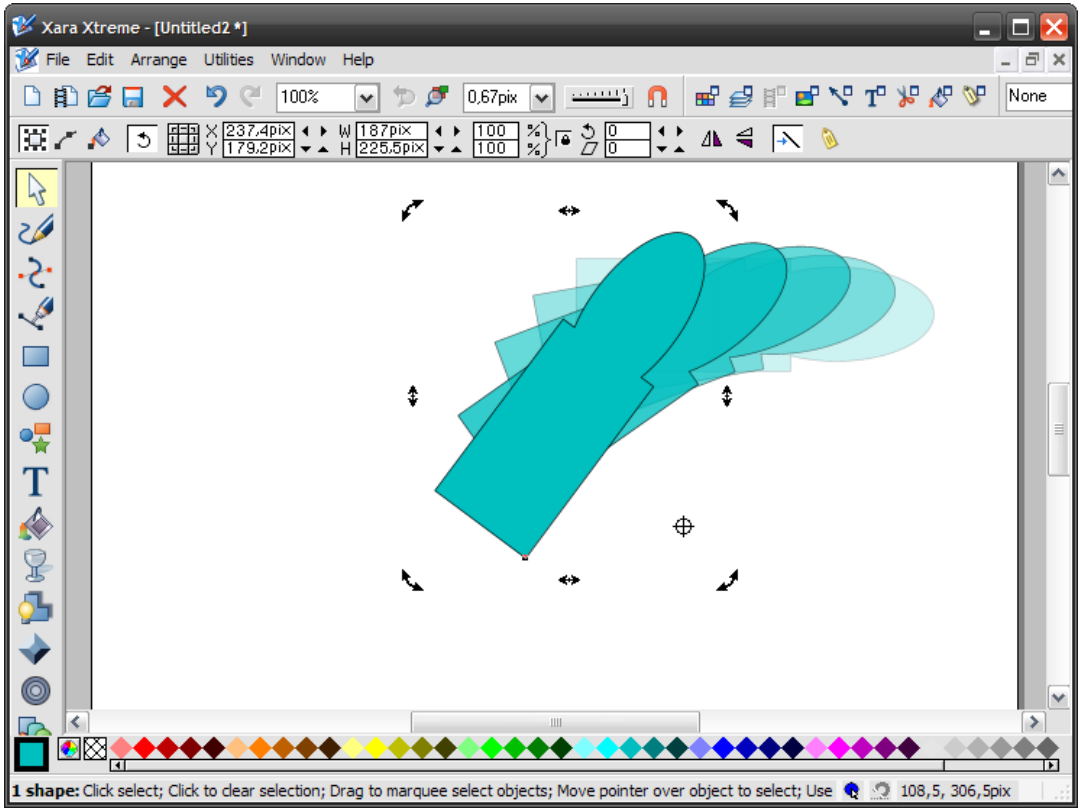
standardmäßig im Zentrum des Objekts



Arbeiten mit 2D-Primitiven

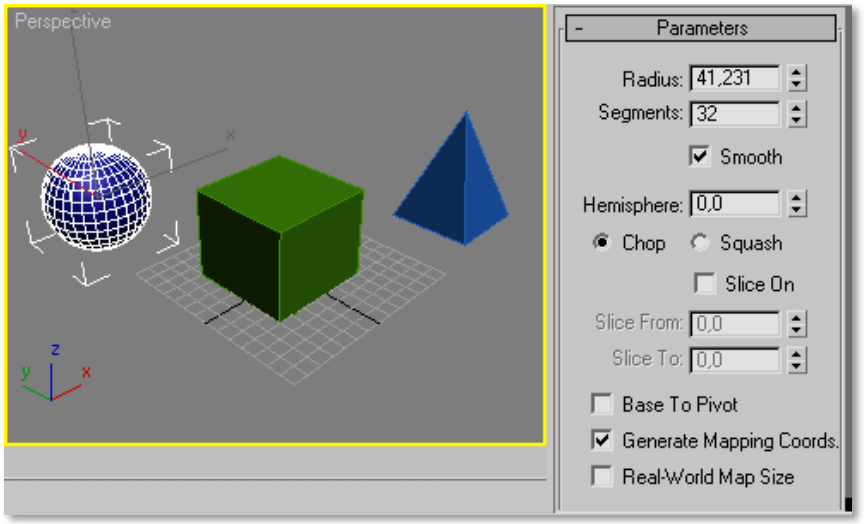
Bezugspunkt kann interaktiv geändert werden

Das Ergebnis der Rotation ist entsprechend



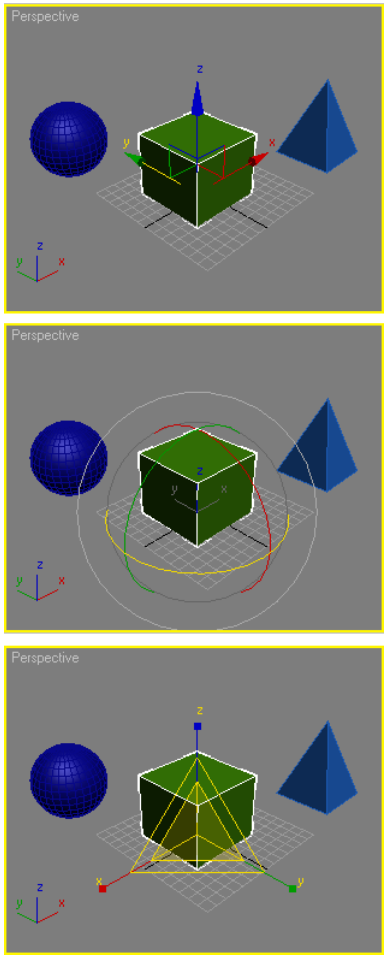
Arbeiten mit 3D-Primitiven

- Werkzeug: 3D-Studio MAX
- 3D-Primitive werden auf Parameter abgebildet
- Objektorientierung: Jedes Primitiv „kennt“ seine Parameter



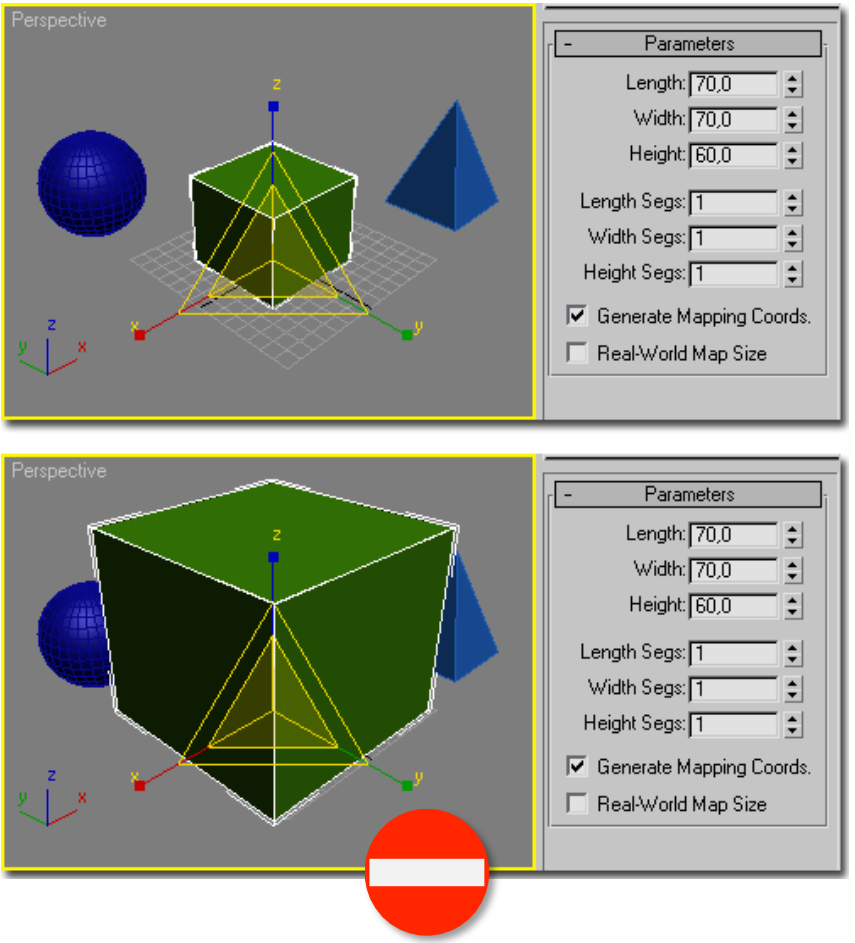
Arbeiten mit 3D-Primitiven

- Transformationen sind interaktiv möglich über so genannte „Gizmos“
- das lokale Koordinatensystem wird repräsentiert durch den „Pivot-Point“



Problem Skalierung

- Zur Erinnerung: 3D-Primitive werden über Parameter eingestellt
- nach Skalieren:
- weiß das Objekt nichts davon, weil sein lokales Koordinatensystem mitskaliert wurde
- Mit Blick auf (mögliche) Programmierung: Niemals skalieren!



Quellen

- Foley, van Dam, Feiner, Hughes. *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim. *Computergraphik 1*. Universität Magdeburg, Vorlesungsskript, Juli 2005.

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik und -animation

Shading und globale Beleuchtungsverfahren, Animationstechniken

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Von 2D nach 3D

Weiter: Modell für eine Sichtbeschreibung

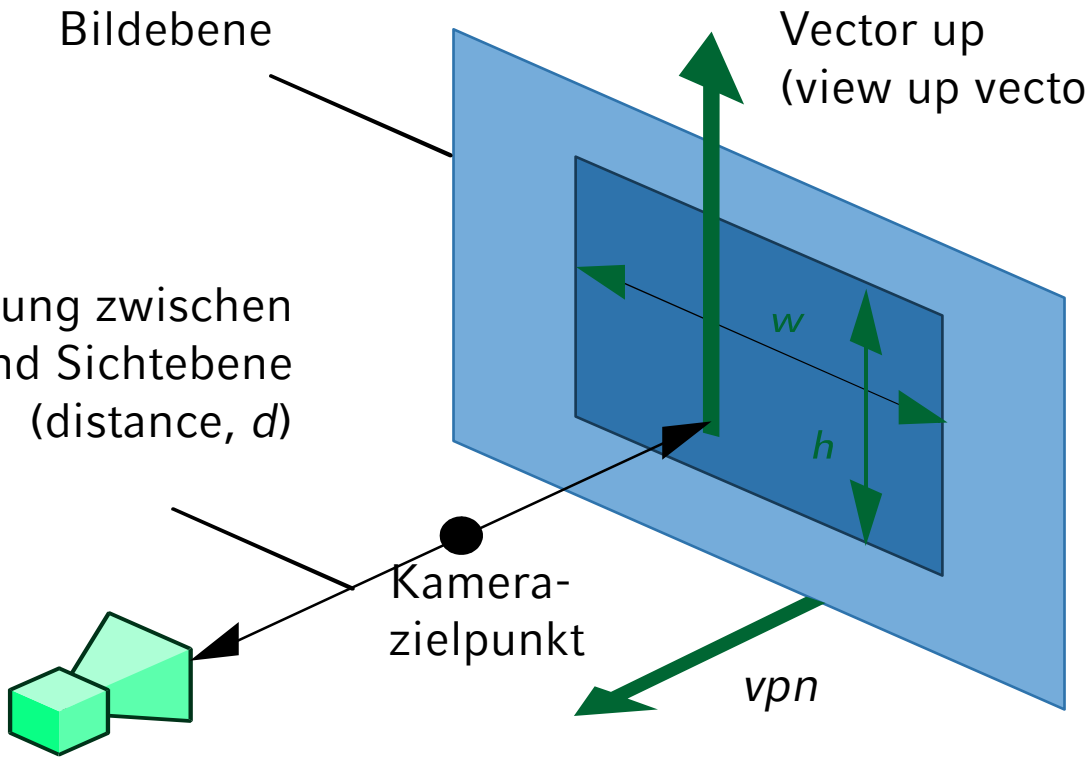


Kameramodell

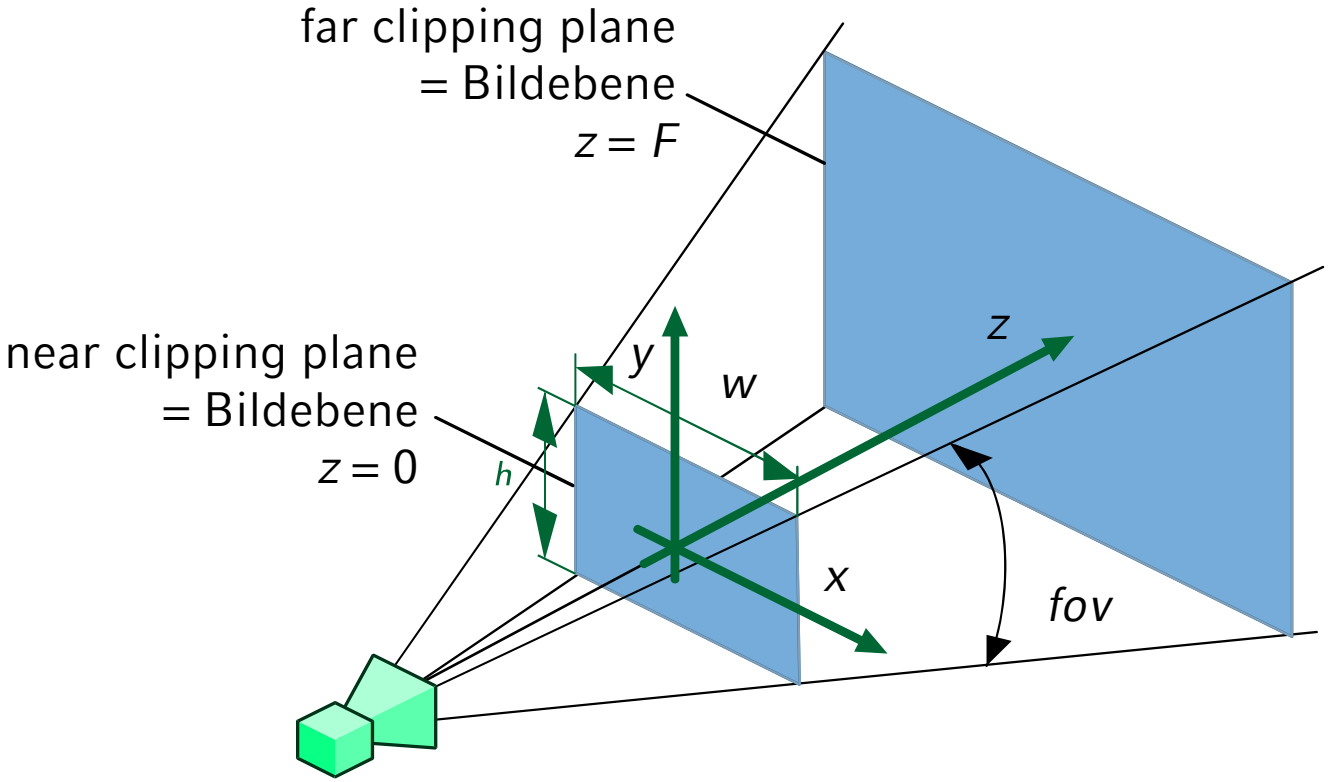
- Reale Kamera als Orientierung und Metapher für den Viewing-Prozess
- Parameter realer Kameras:
  - Position
  - Blickrichtung
  - Brennweite
  - Tiefenschärfe
  - Art des Objektivs (Weitwinkel, Tele) bestimmt den Öffnungswinkel → Projektionen sind teilweise nicht planar
- Parameter der entstehenden Bilder:
  - Größe (Breite, Höhe, bzw. Breite und Seitenverhältnis)
  - Typische Werte für Seitenverhältnis: 4:3 (Fernsehen), 16:9
  - Auflösung



Synthetische Kamera



Kameramodell: Sichtkörper



Kameramodell

- Blickrichtung der Kamera: Entlang der *z*-Achse.
- Sichtebene: Die *xy*-Ebene.
- Warum die Schnittebenen?
  - *Near*-Ebene verhindert, dass Objekte, die sehr nahe an der Kamera sind, alles verdecken und das Objekte hinter der Kamera berücksichtigt werden.
  - *Far*-Plane verhindert, dass weit entfernte und damit im Bild sehr kleine Objekte Rechenaufwand beanspruchen.



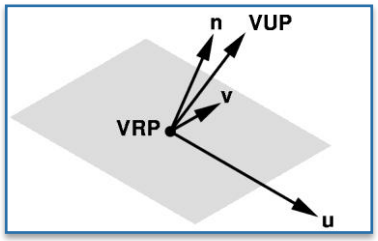
Kameramodell

- Model-View-Transformation beschreibt die Transformation zwischen Modell und der Kamera.
- Repräsentiert durch eine Matrix in homogenen Koordinaten.
- **Standardwert**: Einheitsmatrix
- **Praktisch**:
  - In der Regel ist zumindest eine Verschiebung der Kamera nötig, damit diese außerhalb der Szene platziert ist. Das Modell ist normalerweise um den Ursprung angeordnet.
  - Richtlinien für einheitliche Szenen sind wichtig!



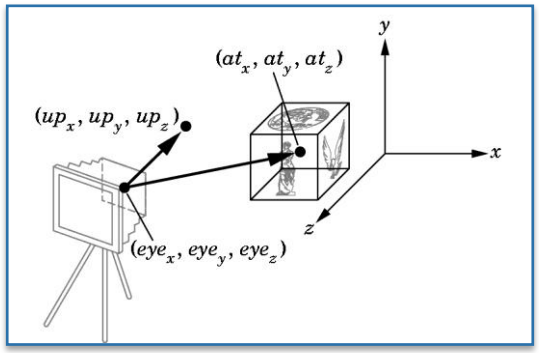
Kameramodell – Formen der Kameraspezifikation

- Angabe der Kameraposition: view_reference_point (*vrp*)
- Angabe der Orientierung: view_plane_normal (*vpn*)
- Angabe der Richtung „nach oben“ view_up-Vektor (*vup*)
- *v* ist Projektion des *vup* in die Sichtebene
- *u* steht senkrecht zu *v* und *n*
- Art der Perspektive wird in der Projektionsmatrix festgelegt (später).



Kameramodell

- Look-At Funktion:
- Kameraposition/Augpunkt: *eye*;
  - Ziel: *at*
  - *vup* kann gewählt werden.
  - $vpn = eye - at$
  - $vrp = eye$
- Wahl des „*At*“-Vectors: z. B. Schwerpunkt des Modelles oder Mittelpunkt des umschließenden Quaders (für eine zentrierte Darstellung)



Mathematische Beschreibung von Projektionen

- Parallelprojektion:
- Projektionsebene sei die *xy*-Ebene
- Projektionsrichtung parallel zur *z*-Achse
- Projektionsrichtung parallel zum Vektor $(x_p; y_p; z_p)^T$ $zp = 0$
- Projektionsgerade für Punkt $(x_1; y_1; z_1)$ erfüllt
  - $x = x_1 + x_p\ u$
  - $y = y_1 + y_p\ u$
  - $z = z_1 + z_p\ u$

Mathematische Beschreibung von Projektionen

- Parallelprojektion
- Darstellung in homogenen Koordinaten:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \\ 0 \\ 1 \end{pmatrix}$$

Mathematische Beschreibung von Projektionen

- Schiefwinklige/schräge Parallelprojektionen:

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \frac{-\cos(\alpha)}{\tan(\beta)} & 0 \\ 0 & 1 & \frac{\sin(\alpha)}{\tan(\beta)} & 0 \\ 0 & 0 & \frac{-1}{\sin(\beta)} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & \frac{-\cos(\alpha)}{\tan(\beta)} & 0 \\ 0 & 1 & \frac{\sin(\alpha)}{\tan(\beta)} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Mathematische Beschreibung von Projektionen

- Bei Kavalierprojektion: $\beta = 45°$
- Geraden in z-Richtung werden nicht verkürzt, wegen $\cos^2\alpha + \sin^2\alpha = 1$

$$P = \begin{pmatrix} 1 & 0 & -\cos(\alpha) & 0 \\ 0 & 1 & \sin(\alpha) & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

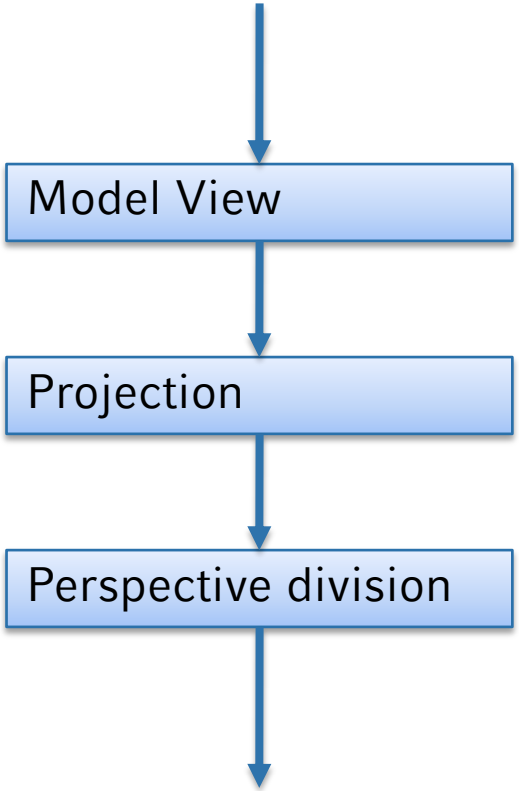
Mathematische Beschreibung von Projektionen

- Bei Kabinettprojektion: β =63,4°
- Geraden in z-Richtung werden halbiert, wegen $\cos^2(\alpha/2) + \sin^2(\alpha/2) = 1/2$

$$P = \begin{pmatrix} 1 & 0 & \frac{-\cos(\alpha)}{2} & 0 \\ 0 & 1 & \frac{\sin(\alpha)}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Zusammenfassung: Projektionen

- Übergang zwischen verschiedenen Koordinatensystemen (3D → 2D)
- Projektionen sind keine affinen Transformationen; sie sind nicht umkehrbar (Parallelität von Geraden wird nicht erhalten).
- Ausnutzen der Eigenschaften homogener Koordinaten bei der Berechnung
- Mathematische Beschreibung war auf die einfachsten Projektionen (Projektionsebene = *xy*-Ebene) konzentriert.
- Transformation der Kamera in Weltkoordinaten (Model-View) und Projektion werden nacheinander durchgeführt.



Ausblick

- In einigen Gebieten der Visualisierung, z. B. bei der Darstellung geographischer Daten und abstrakter Daten werden nichtlineare Projektionen genutzt
- **Beispiele**: Fisheye-Projektionen mit einem oder mehreren Fokuspunkten zur Darstellung geographischer Daten
- **Vorbild**: Projektionen aus der Kartographie, die das Stadtzentrum (als Fokus) vergrößert darstellen



Quellen

- Foley, van Dam, Feiner, Hughes. *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim. *Computergraphik 1*. Universität Magdeburg, Vorlesungsskript, Juli 2005.

Fahrplan für heute

- Beleuchtung und Shading
- Animation



LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

von 3D nach 2D

Rastern und Shading

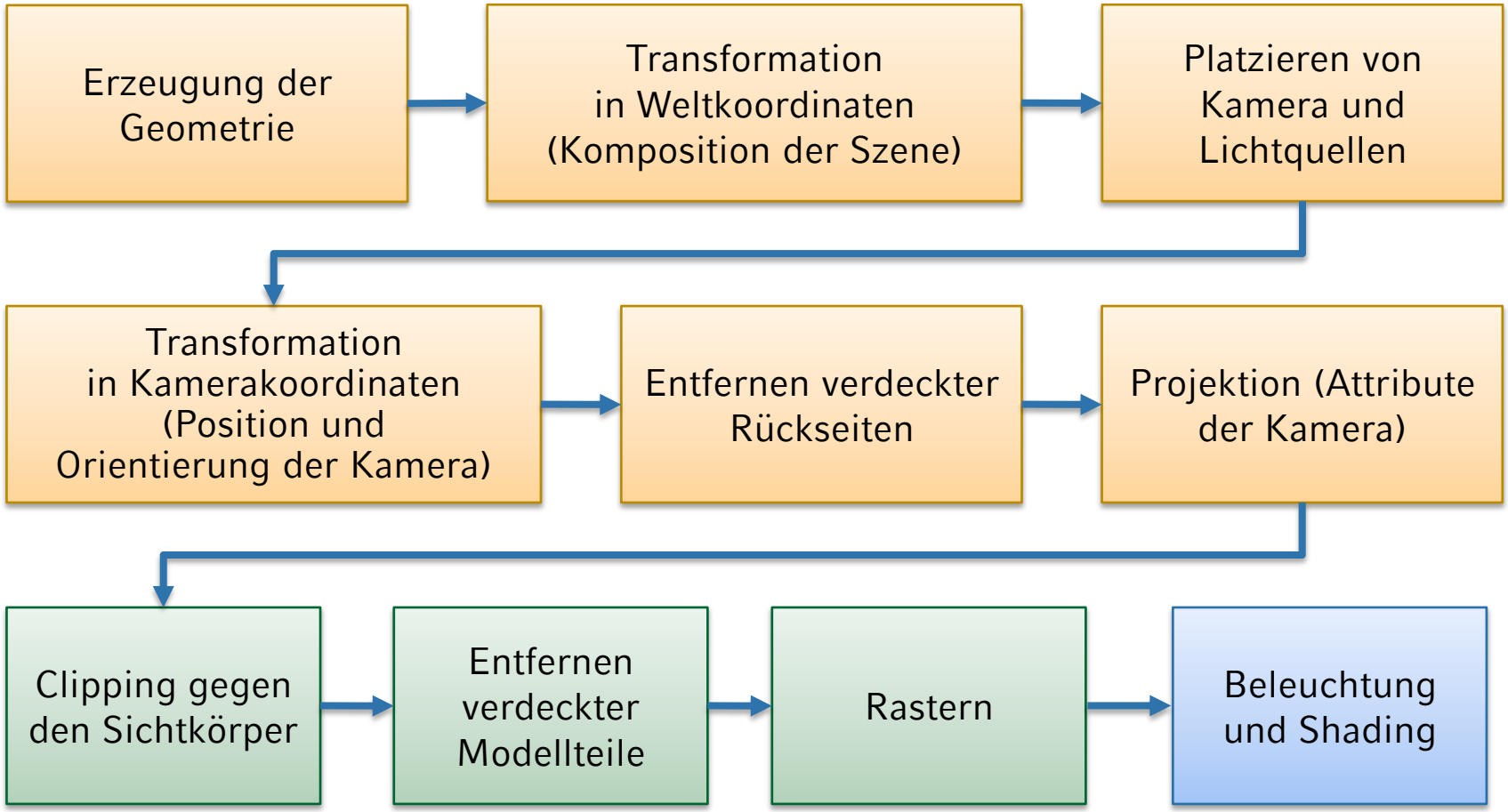


Gliederung

- Beleuchtungsmodelle
- Lichtquellen
- Interaktion von Licht mit Oberflächen
- Beleuchtungsmodell nach Phong
- Shading polygonaler 3D-Modelle
- Globale Beleuchtungsberechnung



Rendering-Pipeline



Beleuchtungsmodelle – Physikalische Effekte

- Oberflächen senden Licht aus oder reflektieren es (oder beides). Teile des Lichtes werden absorbiert; andere Teile (ungerichtet) gestreut
- Lichtintensität und Farbe an einem Punkt der Szene hängt von seiner Lage zu den Lichtquellen ab
- Licht wird oft mehrfach reflektiert/gebrochen und an einem Punkt „addiert“ sich die Lichtintensität, die auf verschiedenen (langen) Wegen an diesem Punkt angekommen ist

Beleuchtungsmodelle – Physikalische Effekte

- Mathematische Beschreibung
  - führt zu einer Integralgleichung – der Renderinggleichung.
  - Diese ist im Allgemeinen nicht lösbar
  - Computergrafik: Annäherungen, die nur bestimmte Arten von Lichtquellen und Oberflächentypen berücksichtigt



Beleuchtungsmodelle

- Beleuchtungsmodelle beschreiben die Faktoren, die die Farbe eines Objektes an einem bestimmten Punkt bestimmen
  Unterscheiden in:
- Globale Beleuchtungsberechnung
  - Einbeziehung aller Objekte einer Szene, um die Farbe an einem Punkt zu bestimmen,
  - Interaktion Lichtquelle-Objekt und auch Interaktion Objekt-Objekt,
  - Oft physikalisch basiert (Erhaltungssätze)
  - Methoden: Raytracing, Radiosity



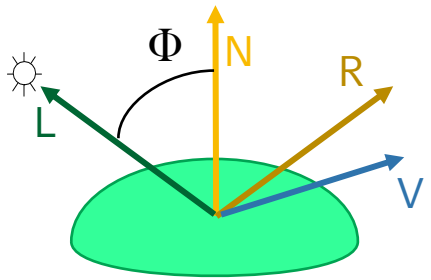
Einschub: Normalenberechnung

- Normalen sind wichtig für die Sichtbarkeitsbestimmung (*occlusion culling*) und die Beleuchtungsberechnung.
  - Die Normale n ist ein Vektor, der senkrecht zu einer Ebene steht. Eine Ebene wird durch drei Punkte definiert (p_0, p_1, p_2)
- Normale wird unter Nutzung des Kreuzproduktes folgendermaßen berechnet:
- $n = (p_2 - p_0) \times (p_1 - p_0)$.
  - Reihenfolge der Punkte ist wichtig, weil $a \times b = -b \times a$
  - Typische Konvention in der CG: Eckpunkte im Uhrzeigersinn angeordnet
  - Bei Polygonen mit > 3 Eckpunkten: Nutze die ersten drei Eckpunkte für die Normalenberechnung



Interaktion von Licht mit Oberflächen

- Einfallswinkel θ: Winkel zwischen L und N (bestimmt die diffuse Reflexion)
- Reflexionswinkel r: Winkel zwischen R und N
- Winkel Φ zwischen V und R bestimmt Intensität des einfallenden Lichts
- Wenn V = R (bzw. Φ = 0) wird das Licht maximal zum Betrachter reflektiert



L-Vektor zur Lichtquelle
N-Oberflächennormale
R-Reflektierter Lichtstrahl
V-Vektor zum Betrachter

von 3D nach 2D

Shading



Beleuchtungsmodell nach Phong

- In der CG am häufigsten verwendetes Modell
- Modelliert Beleuchtung aus drei Anteilen:
- $I_{phong} = I_a + I_d + I_s$



Beleuchtungsmodell nach Phong

- Ambienter Anteil
  - Grundhelligkeit in der Szene
  - Simuliert Streuung des Lichts durch Oberflächen
  - Unabhängig von Betrachterstandpunkt und vom einfallendem Licht
- Diffuser Anteil
  - Reflexion an matten Oberflächen
  - Gleichmäßig in alle Richtungen
  - Unabhängig vom Betrachterstandpunkt
- Spekularer Anteil
  - Gerichtete Reflexion an spiegelnden Oberflächen
  - Abhängig vom Betrachterstandpunkt
  - Erzeugt Glanzpunkte



Beleuchtungsmodell nach Phong

Ambiente Reflexion:

- In der Realität kommt durch vielfältige Licht-Oberflächen-Interaktionen in einem beleuchteten Raum praktisch überall ein gewisses Maß an Licht an.
- Oft ist eine gleichmäßige Beleuchtung auch das Ziel bei der Gestaltung von Räume
- In der CG würde es durch die stark vereinfachte Nachbildung der Beleuchtungsvorgänge dazu kommen, dass Teile der Szene komplett dunkel werden.
- **Daher**: Annahme einer Grundhelligkeit (etwa 10–15% der maximalen Helligkeit). Bewirkt mit geringem Aufwand das, was sonst nur durch Einfügen vieler Lichtquellen erreichbar wäre.
- **Problem**: Farbe des ambienten Lichtes (müsste prinzipiell von allen Objekten in der Szene abhängen)

Lichtquellen

Art	Beschreibung	Spezifikation
Punkt-Lichtquelle	keine Ausdehnung, sendet Licht in alle Richtungen	3D-Koordinaten (Position)
Gerichtetes Licht	parallele Lichtstrahlen (~ einer unendlich weit entfernten Lichtquelle). Effekt wie z. B. Sonnenstrahlen auf der Erde	3D-Vektor (Richtung)
Spotlicht	Punktlichtquelle, die nur in einem bestimmten Öffnungswinkel Licht sendet. Lichtintensität sinkt zu den Rändern des Sichtkegels	3D-Koordinate (Position), 3D-Vektor (Richtung) Öffnungs-Winkel, Maß für das Abfallen der Intensität



Lichtquellen: Weitere Aspekte:

- Abschwächung (attenuation)
  - Intensität des Lichtes wird mit wachsender Entfernung zur Lichtquelle kleiner (physikalisch: $I \sim 1/d^2$)
  - In der CG hat sich eine geringere Abschwächung

    ($I \sim 1/(a + bd + cd^2)$) als günstig erwiesen. a, b und c müssen geeignet gewählt werden. Oft: $a = 0$, $b = 1$, $c = 0$)
- Farbe
  - Lichtquellen haben eine Farbe. Diese wird als RGB-Tripel angegeben. Verhalten der Lichtstrahlen hängt von der Wellenlänge ab. Annäherung: betrachten nur R-, G- und B-Anteile.

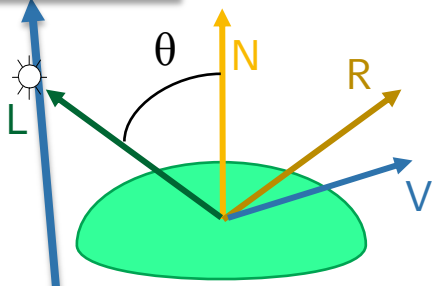
Beleuchtungs-Modell nach Phong

$k_a + k_d + k_s = 1$

$I_{Phong} = L_a k_a + 1/(a + bd + cd^2)(L_d k_d(\mathbf{L} \cdot \mathbf{N}) + L_s k_s(\mathbf{R} \cdot \mathbf{V})^e)$

Ambienter Term modelliert Licht, dass von anderen Oberflächen gestreut wird

Diffuser Term modelliert Reflexion an matten Oberflächen



Entfernungsabhängige Abschwächung für gerichtete und diffuse Reflexion

Spekularer Term modelliert Reflexion an spiegelnden Oberflächen.

Beleuchtungsgleichung muss für jede Lichtquelle (ggf. modifiziert bei Spotlichtern) und jede der 3 Grundfarben ausgewertet werden.

Beleuchtungsmodell nach Phong

- Warum ist die Berechnung bei einer gerichteten Lichtquelle schneller als bei einer Punktlichtquelle?
- Gerichtetes Licht approximiert sehr weit entfernte Lichtquelle → Distanzabhängigkeit der Berechnung entfällt.
- Aufgrund der Parallelität der Lichtstrahlen muss die Normalenberechnung nur einmal pro Polygon durchgeführt werden.



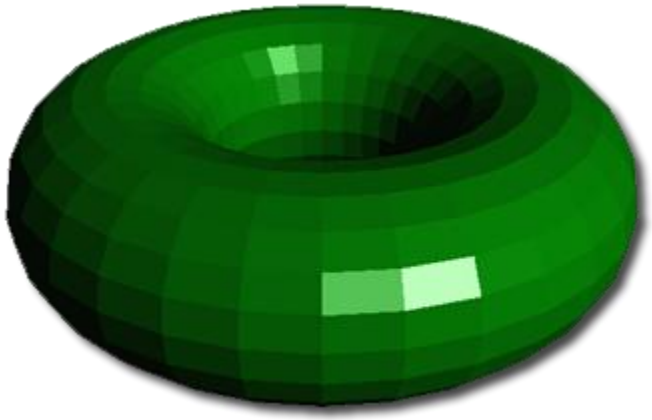
Shading polygonaler 3D-Modelle

- **Ziel**: Zuordnung von Farben zu den Pixeln,
- Betrachten polygonale 3D-Modelle.
- Verfahren unterscheiden sich darin, an welchen Stellen das Beleuchtungsmodell ausgewertet wird.
- Einfache Techniken: interpolative Shading-Verfahren
  - Über das Beleuchtungsmodell werden Farbwerte an den Eckpunkten berechnet.
  - Farbwerte aller Punkte eines Polygons ergeben sich durch lineare Interpolation der Farbwerte an den Eckpunkten.
  - Drei Verfahren
    - Flat-Shading
    - Gouraud-Shading
    - Phong-Shading (Unterschied zu Phong-Modell)



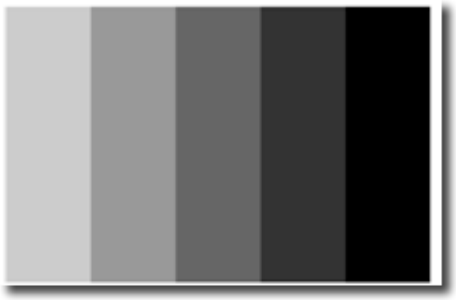
Shading: Flat-Shading

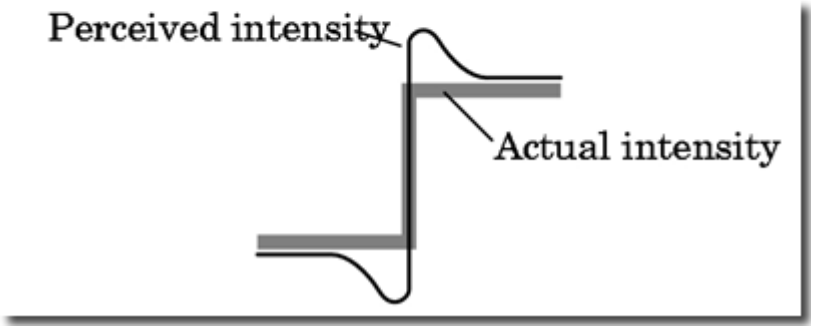
- Keine Interpolation
- Bestimmung eines Farbwerts pro Polygon
  - Variante 1: Mittelwert aus den Farbwerten an den Eckpunkten
  - Variante 2: durch Auswahl eines Punktes und Berechnen der Farbe dort
- Setze alle Pixel des Polygons auf diese Farbe
- Einzelne Polygone deutlich sichtbar
- Einfache Implementierung, sehr schnell, aber geringe Qualität



Shading: Flat-Shading

- Warum sind die Diskontinuitäten an den Kanten so auffällig und störend?
- Visuelle Wahrnehmung: Kontraste werden verstärkt wahrgenommen. Wahrgenommene Helligkeits-unterschiede sind stärker als die (physikalischen) Intensitätsunterschiede.





Shading: Gouraud-Shading

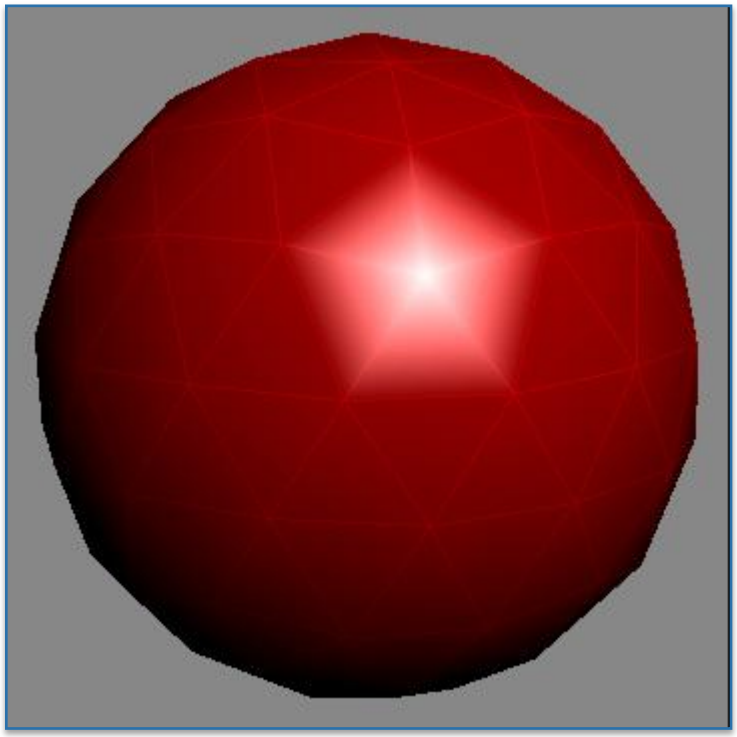
- Berechnung der Helligkeitswerte an den Eckpunkten des Polygons
- lineare Interpolation der Helligkeitswerte über die Fläche des Polygons
- Aufwändiger als Flat-Shading, aber qualitativ besser
- Kanten zwischen Polygonen sind nicht mehr sichtbar



Shading: Gouraud-Shading

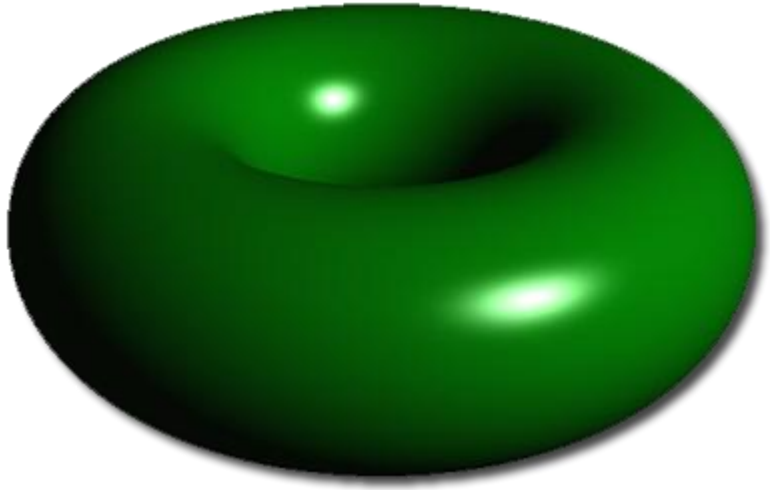
Vorteile	Nachteile

- schnell
- keine Facettierung der Oberflächen mehr
- implementiert in Hardware



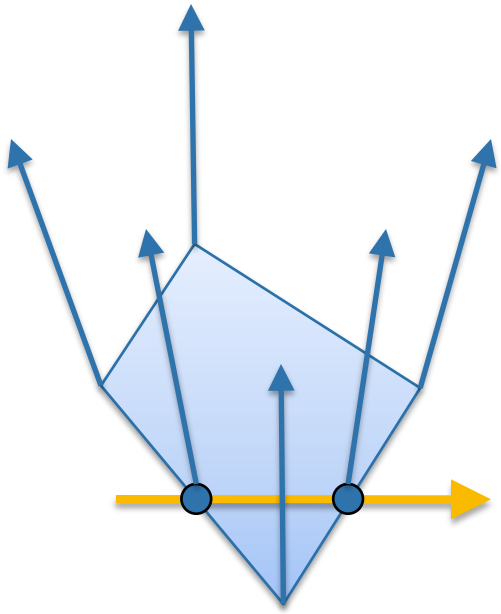
Shading: Phong-Shading

- Interpoliere die Normalenvektoren der Eckpunkte des Polygons über die Fläche des Polygons und berechne an jedem Punkt (Pixel) im Inneren des Polygons einen Farbwert aus der interpolierten Normalen
- Wesentlich aufwändiger als Gouraud-Shading, liefert aber die besten Ergebnisse
- Glanzpunkte im Inneren eines Polygons werden korrekt dargestellt
- Aber: auch einige Fälle, die nicht dargestellt werden können

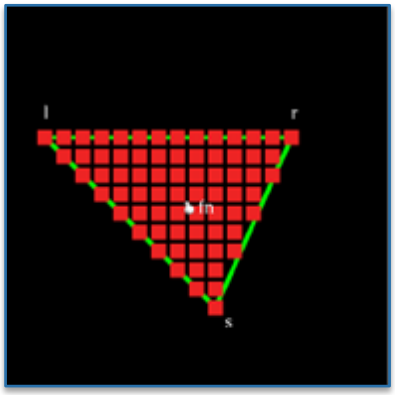


Shading: Phong-Shading

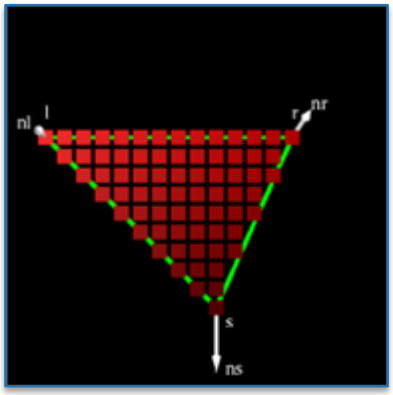
- Normalen-Berechnung entlang von Polygonkanten und Scanlinien durch Interpolation



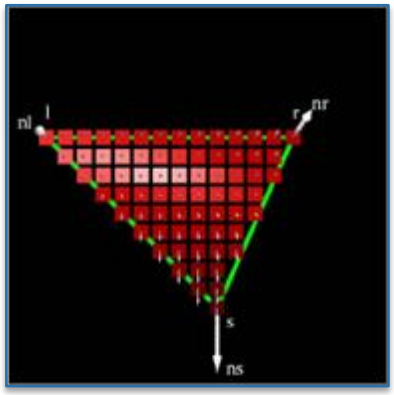
Shading polygonaler 3D-Modelle



Flat-Shading



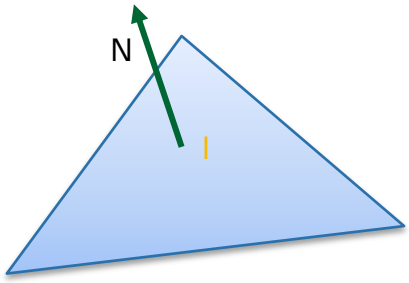
Gouraud-Shading



Phong-Shading

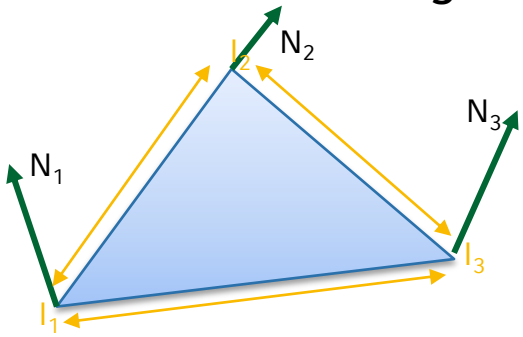
Shading polygonaler 3D-Modelle – Zusammenfassung

Flat-Shading



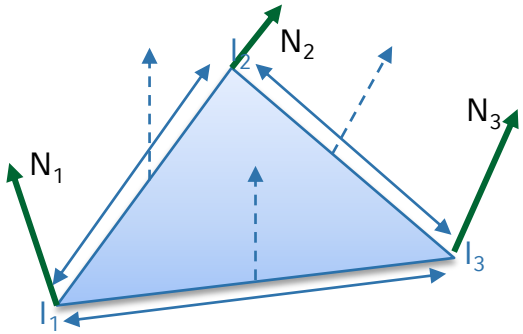
Ein Helligkeitswert für das gesamte Polygon

Gouraud-Shading

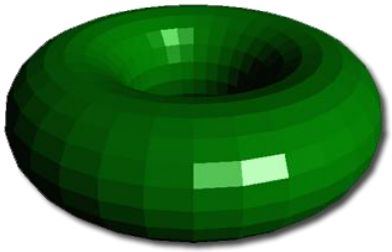


Bestimmung der Helligkeit an den Eckpunkten; Interpolation im Inneren

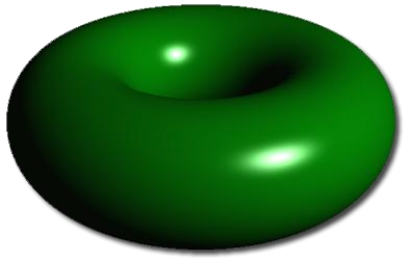
Phong-Shading



Bestimmung der Normalen an den Eckpunkten und Interpolation im Inneren; Bestimmung der Helligkeit mit Hilfe der interpolierten Normalen







Kombination von z-Buffer und Shading-Verfahren

1. Fülle den Framebuffer mit der Hintergrundfarbe
2. Fülle den z-Buffer mit der maximal möglichen Tiefe
3. Zeichne alle Polygone nacheinander:
   a) Für alle Eckpunkte: Bestimme z-Wert und Normale.
   b) Nutze den Scanlinien-Füllalgorithmus zum Zeichnen der Polygone, dabei
      - Interpoliere die entsprechenden Werte (z, N, I) entlang der Kanten und entlang der Spannen.
      - Teste für jedes Pixel, ob der Wert im z-Buffer ein Setzen des Pixels im Framebuffer notwendig macht
      - Aktualisiere z- und Framebuffer



Zusammenfassung

- Vorgestellte Shading-Verfahren sind nicht (korrekt) physikalisch basiert
- Erreichen akzeptable Ergebnisse in zumutbarer Zeit
- Qualität abhängig vom verwendeten Beleuchtungsmodell und vom Shading-Verfahren.
- Beleuchtungsberechnung muss in Weltkoordinaten erfolgen (ein Bezugssystem für alle Koordinaten)
- Cosinus-Funktionen werden für viele Aspekte der Beleuchtungsberechnung eingesetzt (Intensität von Spotlichtern, gerichtete Reflexion $f(\Phi)$, ungerichtete Reflexion $f(\theta)$)



Beispiele

„Virtuelle Kaiserpfalz“

Maic Masuch et al.,
Universität Magdeburg,
1999



Beispiele

„Sonnenblumen“

Oliver Deußen et al.,
Universität Magdeburg,
1997



Beispiele

Oliver Deußen et al.,
Universität Magdeburg,
1997



LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Realistischere Darstellungen in 3D

Globale Beleuchtung

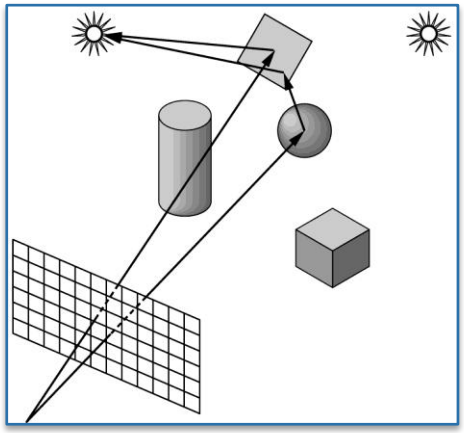


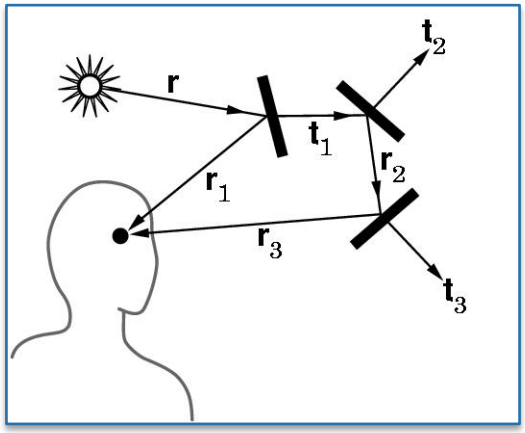
Globale Beleuchtung

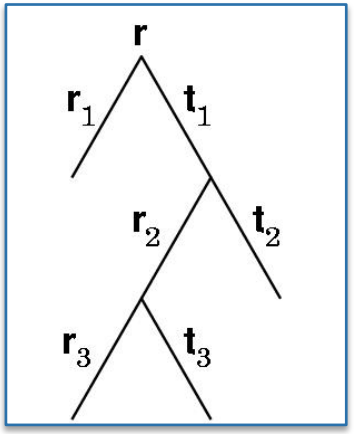
- Globale Beleuchtungsverfahren berücksichtigen nicht nur den direkten Lichteinfall auf die Oberflächen, sondern auch Lichteffekte höherer Ordnung wie Mehrfachreflexionen, Lichtbrechung und Schatten.
- Verbreitet sind zwei große Gruppen von Verfahren:
  - **Raytracing**
    - Verfolgung von Strahlen vom Bild zurück in die Szene bis zu den Lichtquellen. Mehrfache gerichtete Reflexion wird korrekt dargestellt.
  - **Radiosity**
    - Basiert auf dem Energieerhaltungssatz und betrachtet den Energieaustausch zwischen Flächen. Mehrfache diffuse Reflexion wird korrekt dargestellt.



Globale Beleuchtung: Raytracing







- Bei der rekursiven Verfolgung von Lichtstrahlen entsteht ein Baum, wobei jeder Strahl einen bestimmten Anteil an der resultierenden Intensität hat
- Strahlverfolgung/Rekursion wird abgebrochen, wenn der Anteil sehr klein ist bzw. die Rekursionstiefe einen Schwellwert erreicht hat

Globale Beleuchtung: Raytracing

- Aufwand hängt ab von der Zahl der Strahlen, die verfolgt werden und von den Schnittpunktberechnungen
- Test, ob Strahl ein Objekt trifft, kann durch Einsatz hierarchischer Datenstrukturen (z. B. Octtree) oder umschließende Hüllkörper (Kugeln, Bounding Box) verringert werden
- **Ziel:** In den meisten Fällen durch einen schnellen Test (z. B. Strahl – Kugel) entscheiden, ob aufwändige Schnittpunktberechnung nötig
- **Qualität:** Diffuse Mehrfachreflexionen (in alle Richtungen) können nicht nachgebildet werden



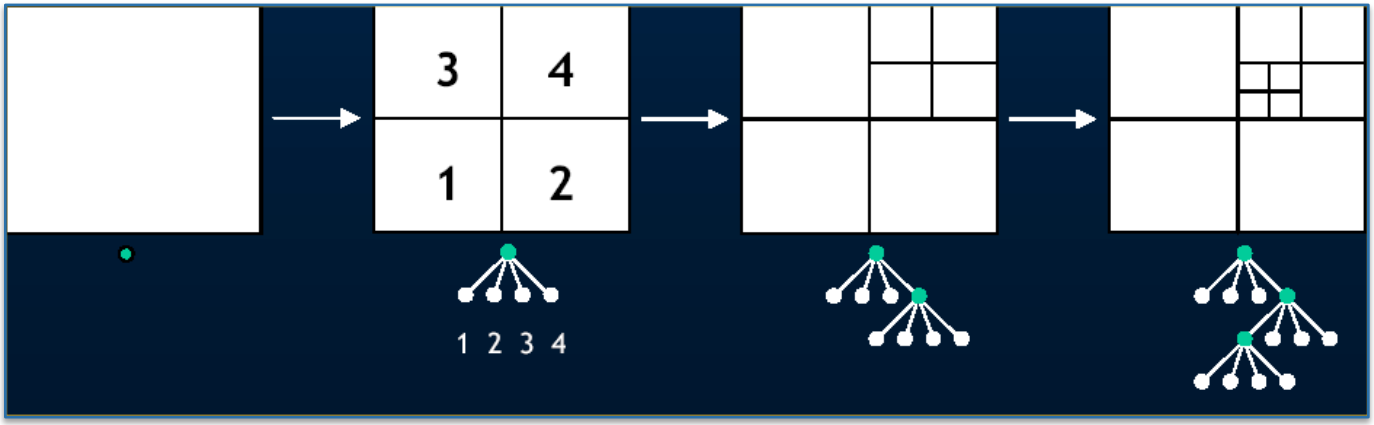
Globale Beleuchtung: Raytracing

- Beschleunigung der Schnittpunktberechnung durch umschließende Hüllkörper
- Kriterien für Hüllkörper:
  - Effiziente Berechnung
  - Gute Anpassung



Globale Beleuchtung: Raytracing

- Beschleunigung der Schnittpunktberechnung durch Octrees
- Hierarchische Unterteilung des Raumes in je 8 gleich große Kinder (Teilung entlang der *x*-, *y*- und *z*-Achse)
- Repräsentation in welchen Teilen der Szene sich Objekte befinden
- Adaptive Unterteilung („leere“ Knoten werden nicht unterteilt).



Globale Beleuchtung: Raytracing

© Cornell University Program of Computer Graphics. 1990



Globale Beleuchtung: Radiosity

- Szene wird angenommen als perfekt diffus reflektierend. Unter dieser Annahme kann die Renderinggleichung numerisch gelöst werden.
- Polygone werden unterteilt in kleine Patches. Gesucht ist der Einfluss/der Anteil des diffus reflektierten Lichtes von einem Patch auf einen anderen ($O(n^2)$).
- Ausgehend von Betrachtungen zur Verteilung von Wärmeenergie wird der Einfluss in sogenannte Formfaktoren berechnet und gespeichert.
- Szene wird daraufhin mit Flatshading dargestellt, wobei Formfaktoren berücksichtigt werden.

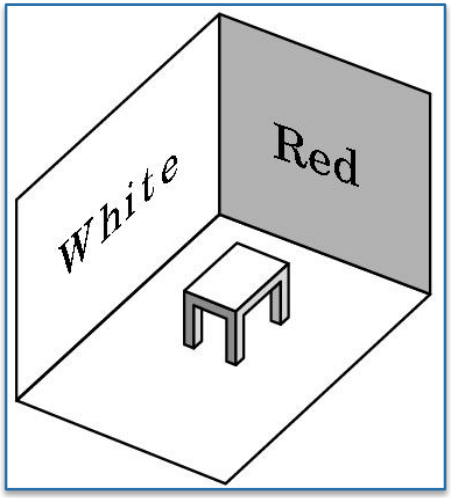


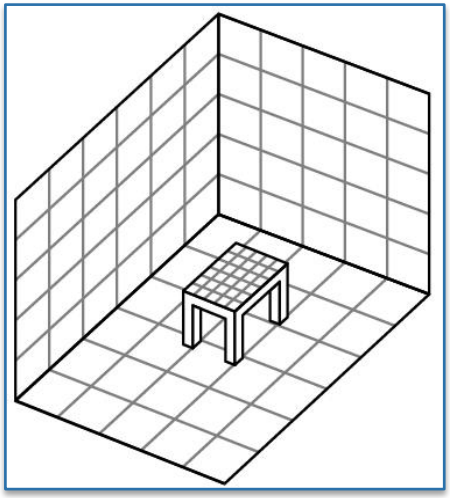
Globale Beleuchtung: Radiosity

- Wesentlicher Aufwand ist die Berechnung der Form-Faktoren (FF). Aufwand und Qualität hängen von der Diskretisierung (Zahl der FF) ab.
- **Ziel**: Adaptive Diskretisierung (an den Stellen feiner unterteilen, wo es besonders auffällt)
- FFs sind unabhängig von der Betrachterposition und müssen damit bei Rotation/Zoom/Kamerafahrt nicht neu berechnet werden.
- Qualität wird deutlich bei indirektem Licht. Eine beleuchtete große Wand reflektier Licht diffus auf Gegenstände im Raum oder andere Wände.

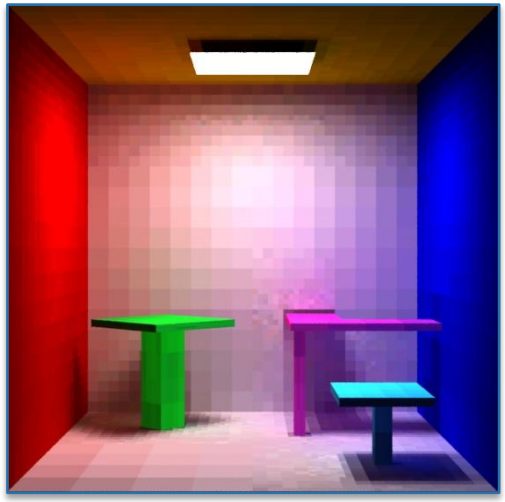


Globale Beleuchtung: Radiosity





Eine einfache Szene für Radiosity-Darstellungen und die Unterteilung in Patches



Patches für die Radiosity-Berechnung

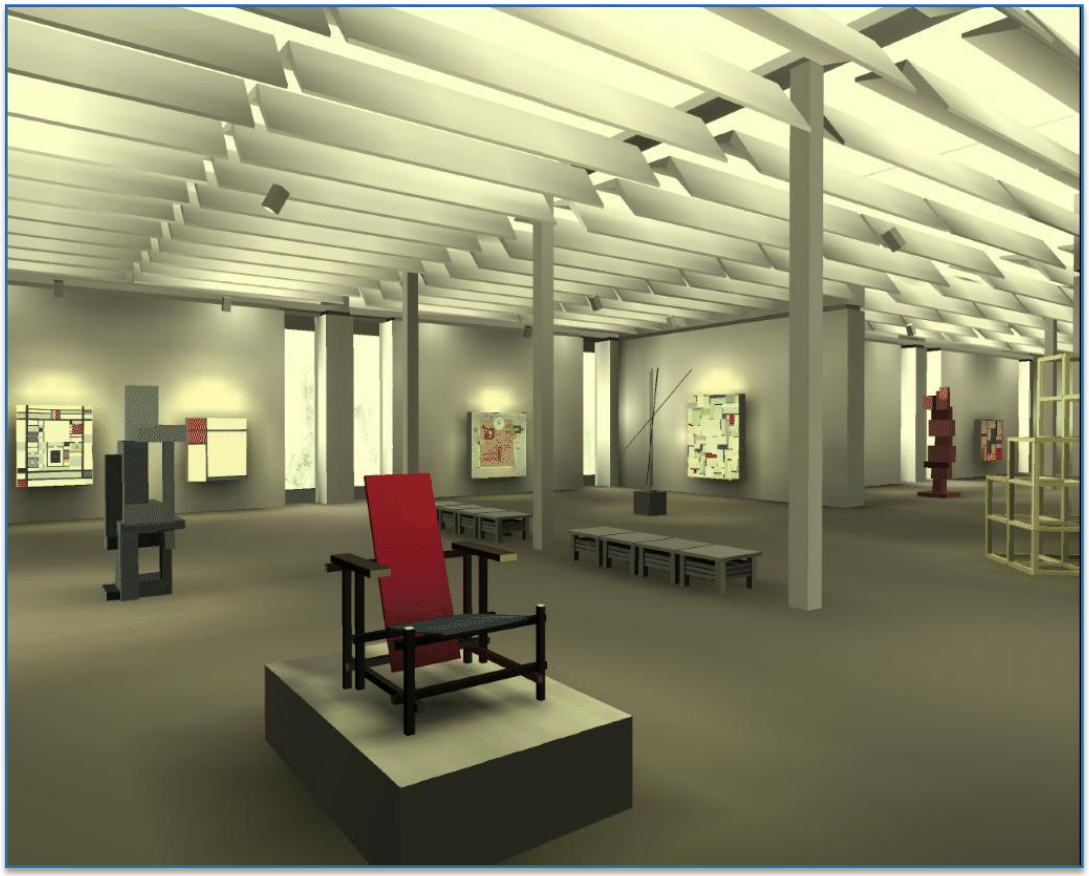
Globale Beleuchtung: Radiosity

© Michael Cohen et al., Cornell University, Program of Computer Graphics.1988 gerendert; 30.000 Patches, 190 Stunden Rechenzeit auf einer VAX

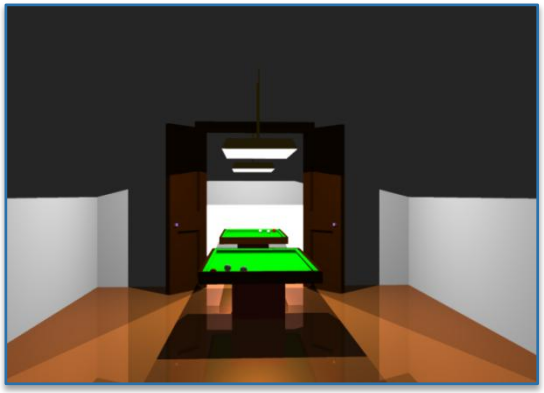
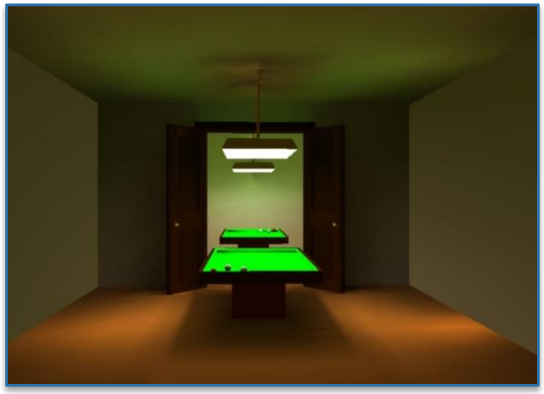


Globale Beleuchtung: Radiosity

© Michael Cohen et al. Cornell University, Program of Computer Graphics. 1988 gerendert



Globale Beleuchtung: Raytracing und Radiosity

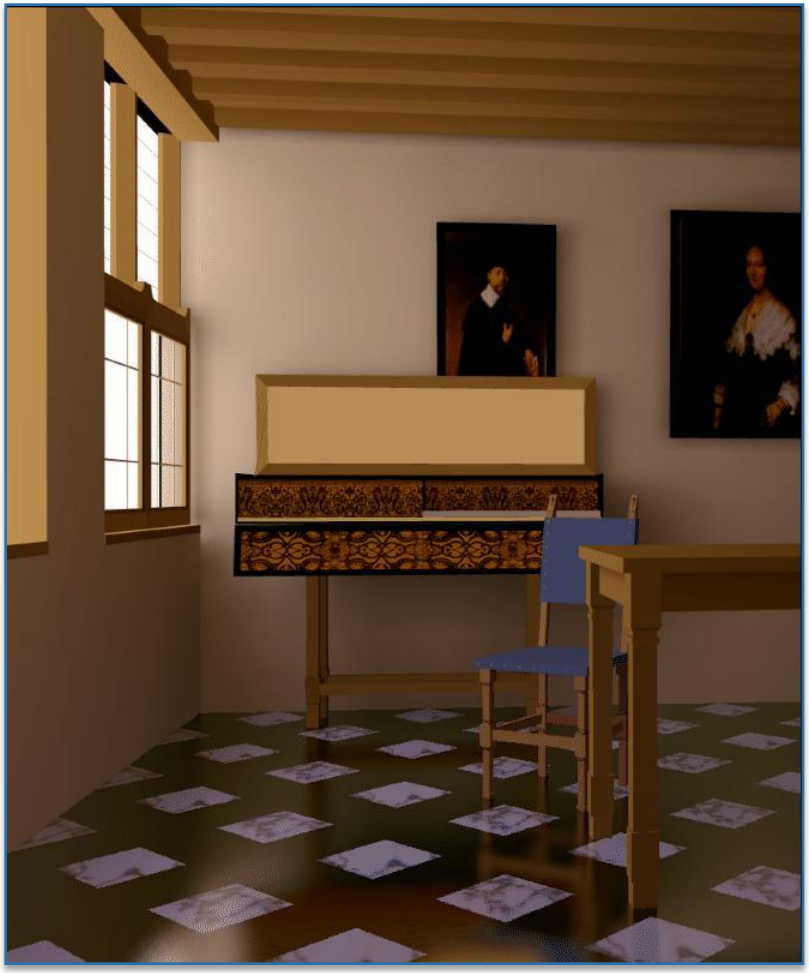




- Raytracing im 1. Schritt,
- Radiosity im 2. Schritt, Kombination rechts

Globale Beleuchtung

Kombination von
Raytracing und Radiosity
in einem 2-Pass-Verfahren

© Michael Cohen et al.
Cornell University Program
of Computer Graphics,
1987



Zusätzliche Techniken

- Modellierung von Oberflächendetail über Texturen: Texture Mapping, Bump Mapping,
- Atmosphärische Effekte wie Nebel, Tiefenschärfe, etc.
- Simulation realer Objekte und Materialien (Rauch, Feuer, Wasser, etc.)
- Animation



Quellen

- Axel Hoppe. *Validierung und Nachbearbeitung von gerenderten Bildern.* Dissertation an der Otto-von Guericke-Universität Magdeburg, Shaker-Verlag, Aachen, 1999. ISBN 3-8265-4610-5
- Foley, van Dam, Feiner, Hughes. *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim. *Computergraphik 1*. Universität Magdeburg, Vorlesungsskript, Juli 2005.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Computergrafik I

Sichtbarkeitsbestimmung



Letztes Mal

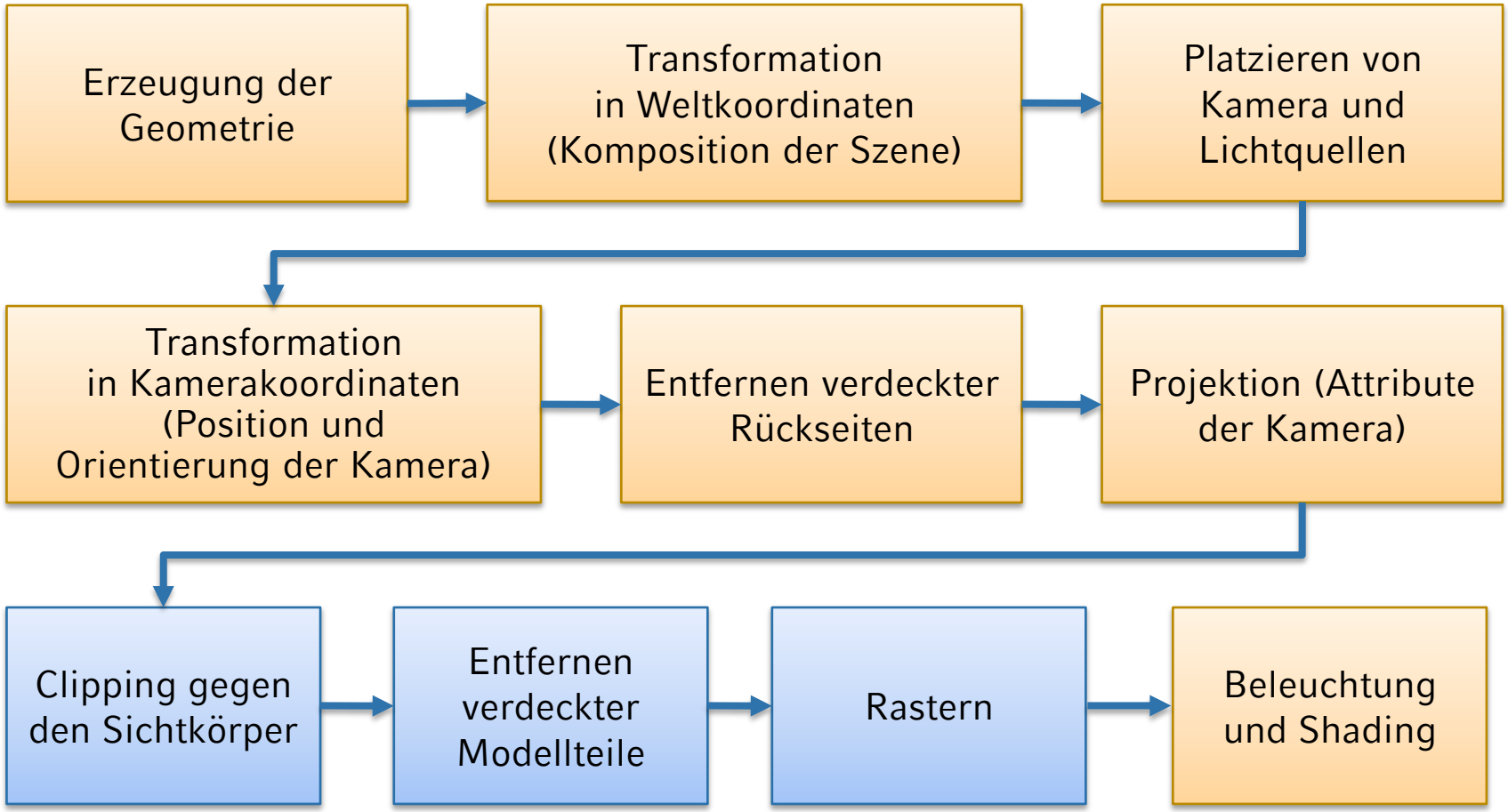
- Shading
- Lokales Beleuchtungsmodell
- Globale Beleuchtungsmodelle



Fahrplan für heute

- Nachsatz: Prozedurale Modellierung
- Animationstechniken
- Sichtbarkeitsbestimmung
- Rastern (und Shading)

Rendering-Pipeline



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

von 3D nach 2D

Prozedurale Modellierung



L-Systeme

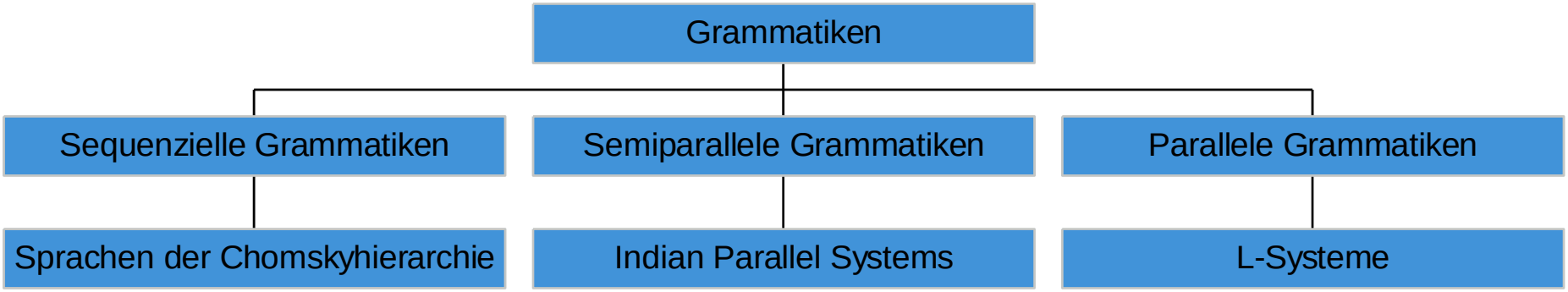
- Entwickelt von Aristid Lindenmayer
- Ende der 60iger Jahre
- Entwickelt als Automat und später als Teil der Formalen Sprachen definiert
- Mathematisches Modell zur Beschreibung von biologischen Prozessen
  - Zelltrennung
  - Pflanzenwachstum

L-Systeme

- Bereich der Theoretische Informatik
- Formale Sprachen
- Parallele Ersetzungsgrammatik



L-Systeme


Grammatiken
Sequenzielle Grammatiken
Semiparallele Grammatiken
Parallele Grammatiken
Sprachen der Chomskyhierarchie
Indian Parallel Systems
L-Systeme

L-Systeme

- Mittels eines L-Systems wird ein Wort generiert
- Wortgenerierung erfolgt über Regeln
- Beispiel
  - Startwort: F
  - Regel: F → FF
  - Wortgenerierung
    - 1. Ableitung: F
    - 2. Ableitung: FF
    - 3. Ableitung: FFFF
    - 4. Ableitung: FFFFFFFF
    - 5. Ableitung: FFFFFFFFFFFFFFFF

L-Systeme

- Bestimmte Zeichen können als Kommandos interpretiert werden
- Beispiele
  - F0
    - Gehe eine bestimmte Wegstrecke und hinterlasse eine Linie (im 2D-Raum)
  - M0
    - Gehe eine bestimmte Wegstrecke und hinterlasse keine Linie (im 2D-Raum)
  - L(x)
    - Änder die Länge der Wegstrecke
  - LMul(x)
    - Multipliziere die aktuelle Länge der Wegstrecke mit x
  - RU(x)
    - Rotiere um x-Grad auf der y-Achse



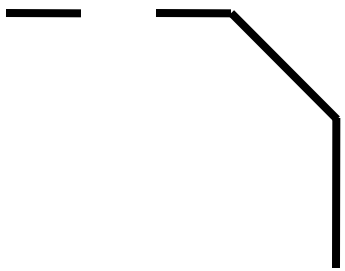
L-SYSTEME

- Was ist das graphische Ergebnis aus dem folgenden Wort?
- `L(10) F0 RU(45) F0 RU(45) LMul(0.5) F0 M0 F0 ?`



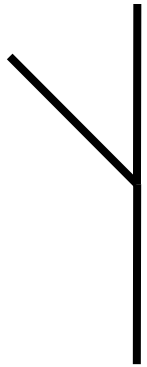
L-Systeme

- `L(10) F0 RU(45) F0 RU(45) LMul(0.5) F0 M0 F0`



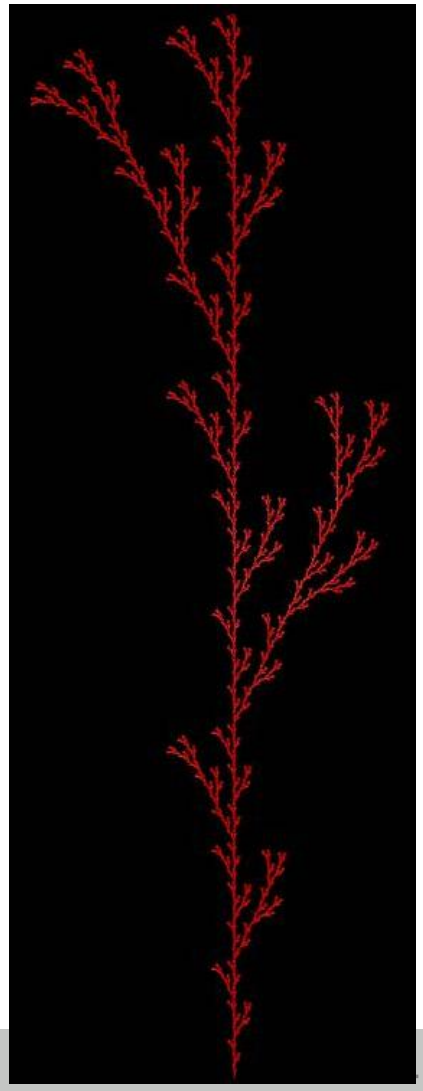
L-Systeme

- Realisierung von Verzweigungen mittels []
- Beispiel: F [RU(45) F] F



L-Systeme

- Ableitungsschritte = 5
- Basisstärke = 60
- Startaxiom = F
- Regel: F → F[RU(25.7)F]F[RU(-25.7)F]F

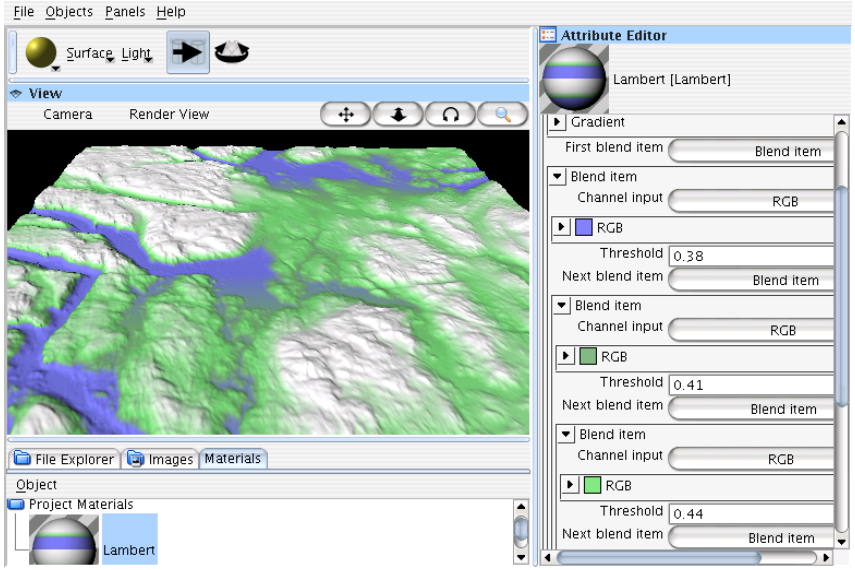


L-Systeme



Die Plattform GroIMP

- Freie Entwicklungsplattform (Open Source) für 3D Modellierung



Wie kann GroIMP eingesetzt werden?

- Büsche, Bäume, Blumen, …, jegliche Art von Pflanzenstrukturen
- Steine, Felsen und Gebirge
- Verschiedene Arten von Landschaften
  - Mondlandschaft, Vulkanlandschaft,etc.

Warum prozedurale Modellierung?

- Biologische Korrektheit
  - Integration des Öko-Systems
  - Wettkampf um Ressourcen
  - Realistischere Modelle
- Genaues Modellieren
  - Genaue Dimensionierung von 3D-Teilen
  - Die 3D-Primitiven sind alle miteinander verbunden
  - Keine fehlenden 3D-Teile



Beispiele

Oliver Deußen et al.,
Universität Magdeburg,
1997



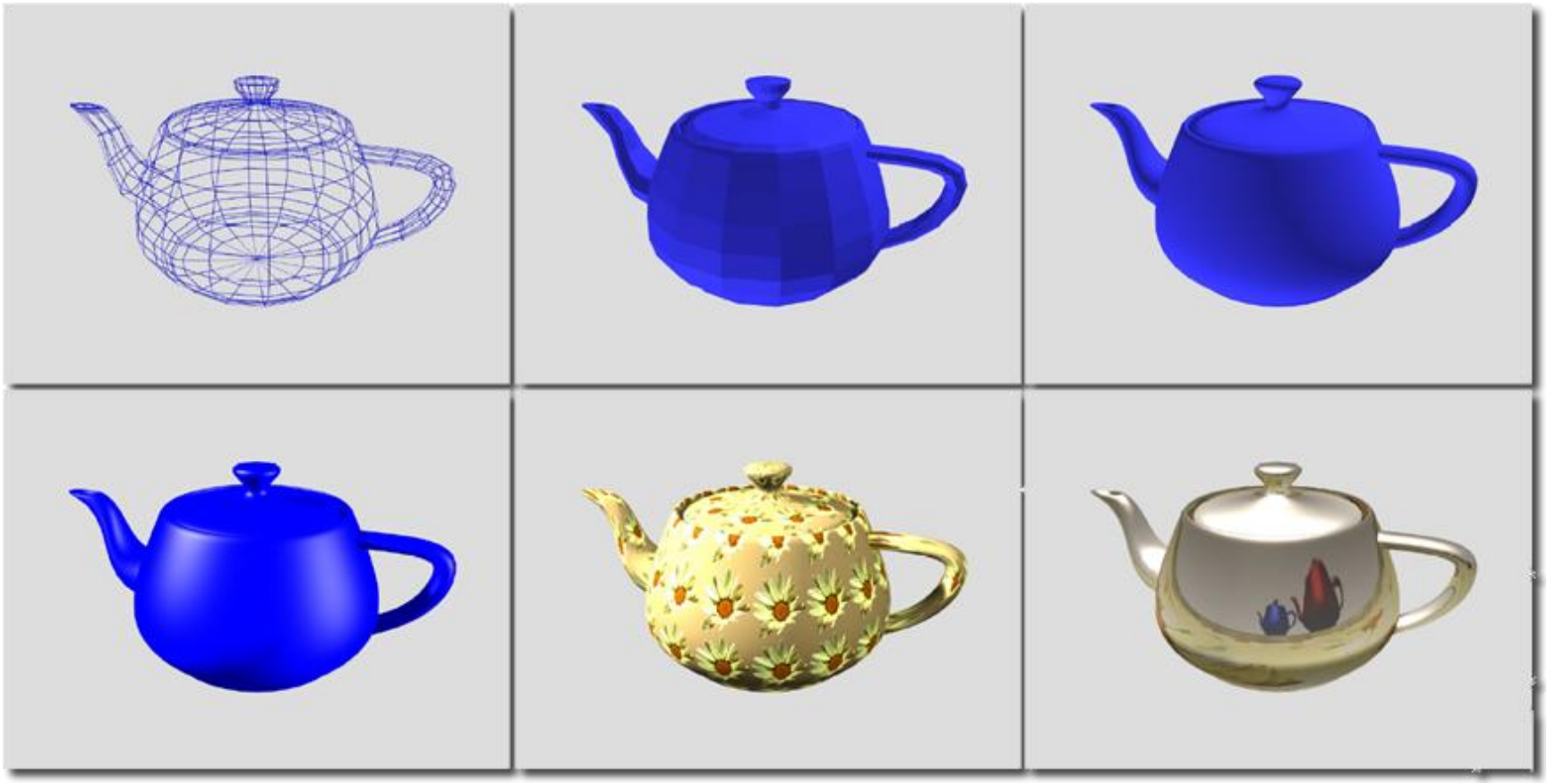
LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

von 3D nach 2D

Materialien

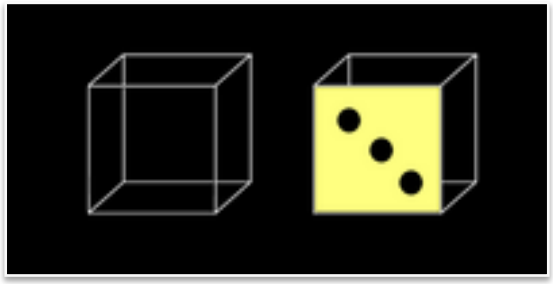


Weitere Techniken



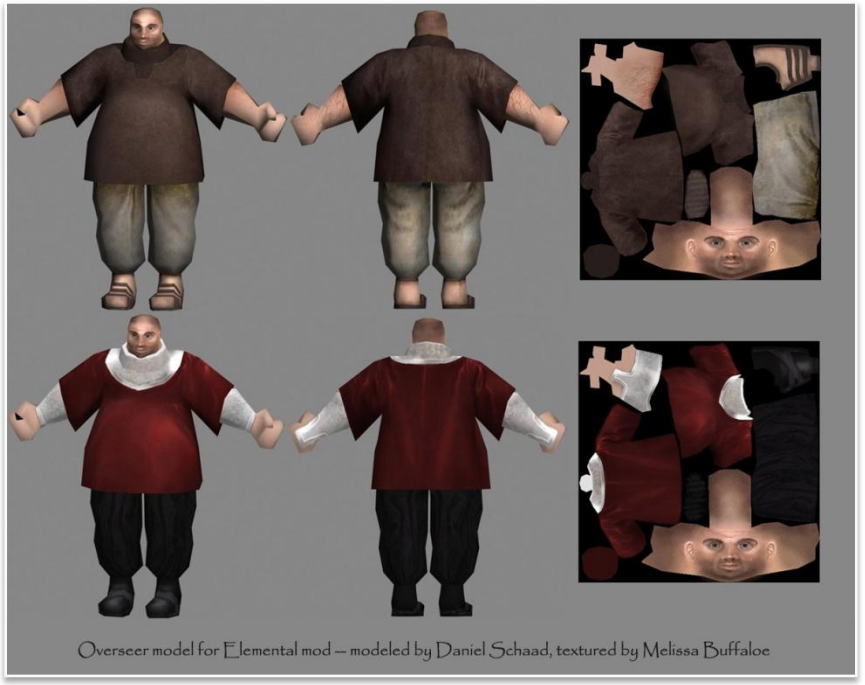
Mapping

- Auftragen einer 2D Map auf 3D Polygon(e) (Drahtgitter)
- Map/Textur: Bild/Muster oder Oberflächeneigenschaften
- erhöht Realismus ohne Anzahl der Polygone (Rechenaufwand) zu erhöhen
- Jeder 3D Vertex(x,y,z) bekommt Texel zugeordnet
- Texel:
  - u-,v-Koordinaten
  - Position auf der Textur (0,0 links unten, 1,1 rechts oben)
- Interpolation von Eckpunkt zu Eckpunkt eines Polygons



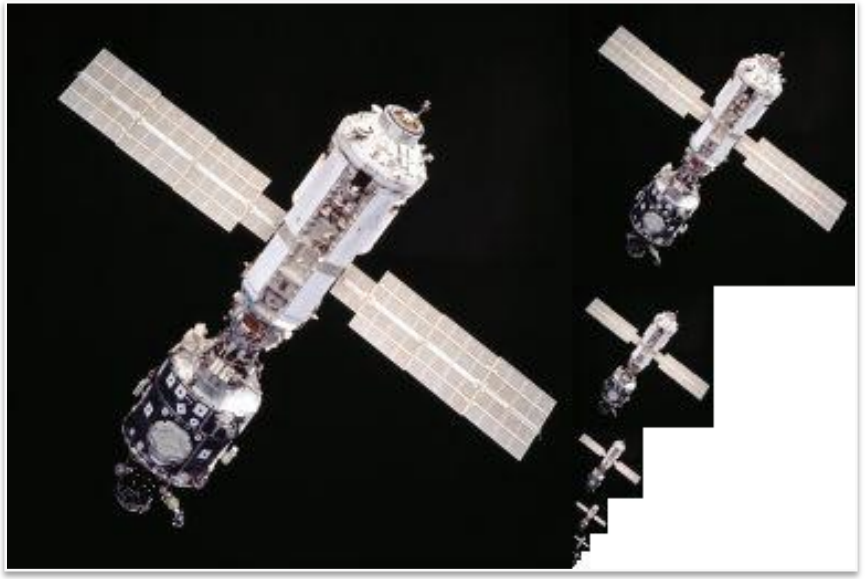
Texture Mapping (Diffuse Color Map)

- Textur: Bild oder Muster
- Problem: Ansicht auf weit entferntes oder nahe Objekte
  - Pixel können wesentlich größer/kleiner als Texel sein
  - Mittelwertberechnung meist zu aufwendig
  - Lösung: Mip Mapping



MIP-Mapping

- Ursprüngliche Textur wird abgespeichert und verkleinert
  - Verhältnis 4:1 (halbe Höhe, halbe Breite)
  - mit entstandenen Texturen rekursiv fortgeführt bis Auflösung 1:1
  - häufig von Grafikhardware beim Texturladen angewandt
- Textur mit passender Größe wird verwendet
- 1/3 höherer Speicherbedarf



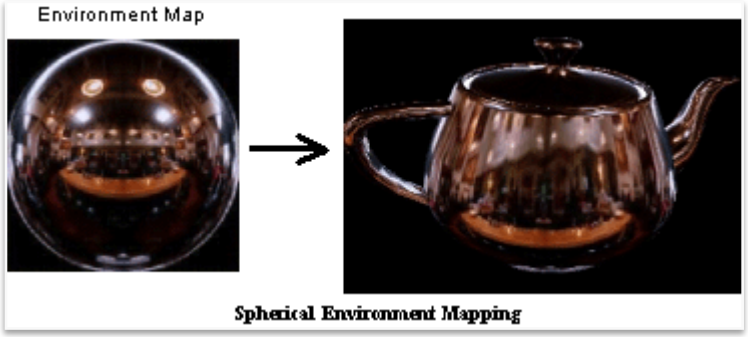
Environment Mapping

Spiegeleffekte der Umgebung simuliert, ohne eine exakte Lichtverteilungsberechnung

durchzuführen

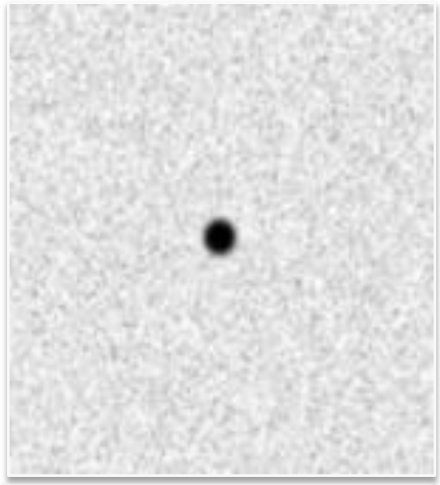
Cubic EM: 6 Maps (vorn, hinten, oben,...)

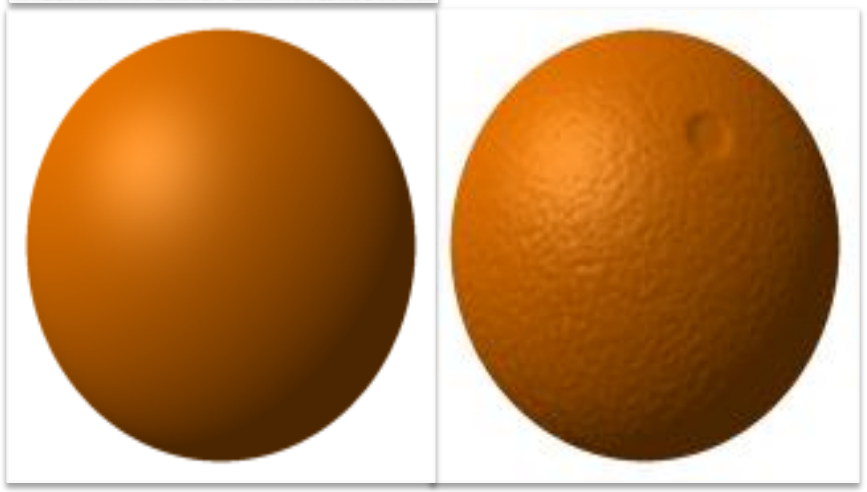
Spherical EM: eine Map für alle Seiten



Bump Mapping

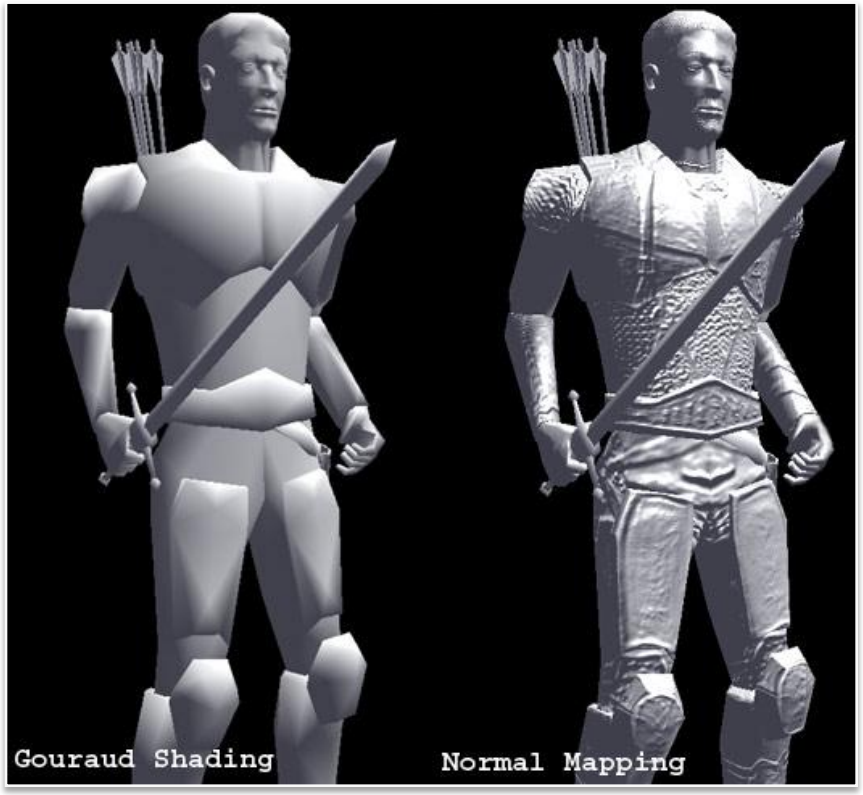
- Höhenunterschiede simuliert, die in der Geometrie des Modells gar nicht vorhanden sind
- Bump-Map:
  - besteht aus Grauwerten
  - Schwarz (Farbwert 0) die „tiefste“ Stelle
  - Weiß (Farbwert: 255) die „höchste“ Stelle





Normal Mapping

- Oberflächenunebenheiten simuliert, die in der Geometrie des Modells gar nicht vorhanden sind
- Normal-Map:
  - RGB-Werte als XYZ Werte interpretiert
  - normalisierter Vektor ist Normale für Helligkeitsberechnung
  - von Hi-Res-Modell errechnet und für Low-Res-Modell verwendet



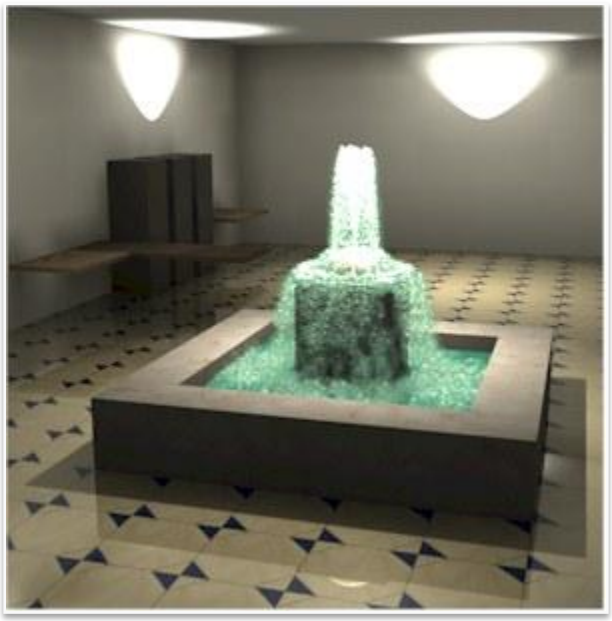
Displacement Mapping

- Höhenwerte aus Bump-Map werden zum erzeugen neuer und verschieben vorhandener Vertices benutzt
- Vorteil: Geometrie wird verändert
- Nachteil: deutlich mehr Rechenaufwand



Partikel-Systeme

- einfache Weise eine große Anzahl von Objekten zu animieren
  - Effekte (Feuer, Wolken, Rauch, Wasser)
  - Bewegegung vieler Objekte (Schwärme, Asteroidenfelder)



Zusammenfassung

- Rendering besteht aus Viewing und Shading
- Wichtige Bestandteile:
  - Transformationen bzw. Kameramodell
  - Entfernen verdeckter Modellteile
  - Beleuchtungsmodell
  - Shading (Anwenden des Beleuchtungsmodells)
- Unterschiedliche Verfahren in jedem Schritt möglich, daher große Breite von Effekten
- Effiziente Algorithmen und Datenstrukturen (effizienter Zugriff) für die jeweiligen Schritte nötig.



Zusammenfassung: Gestalterische Relevanz

- keine unendlich scharfen Kanten
- diffuse Reflexionen
- Schattenwurf
- Varianz
  - in Gestalt und
  - Oberfläche
- Ambient Occlusion (Dirt Maps)



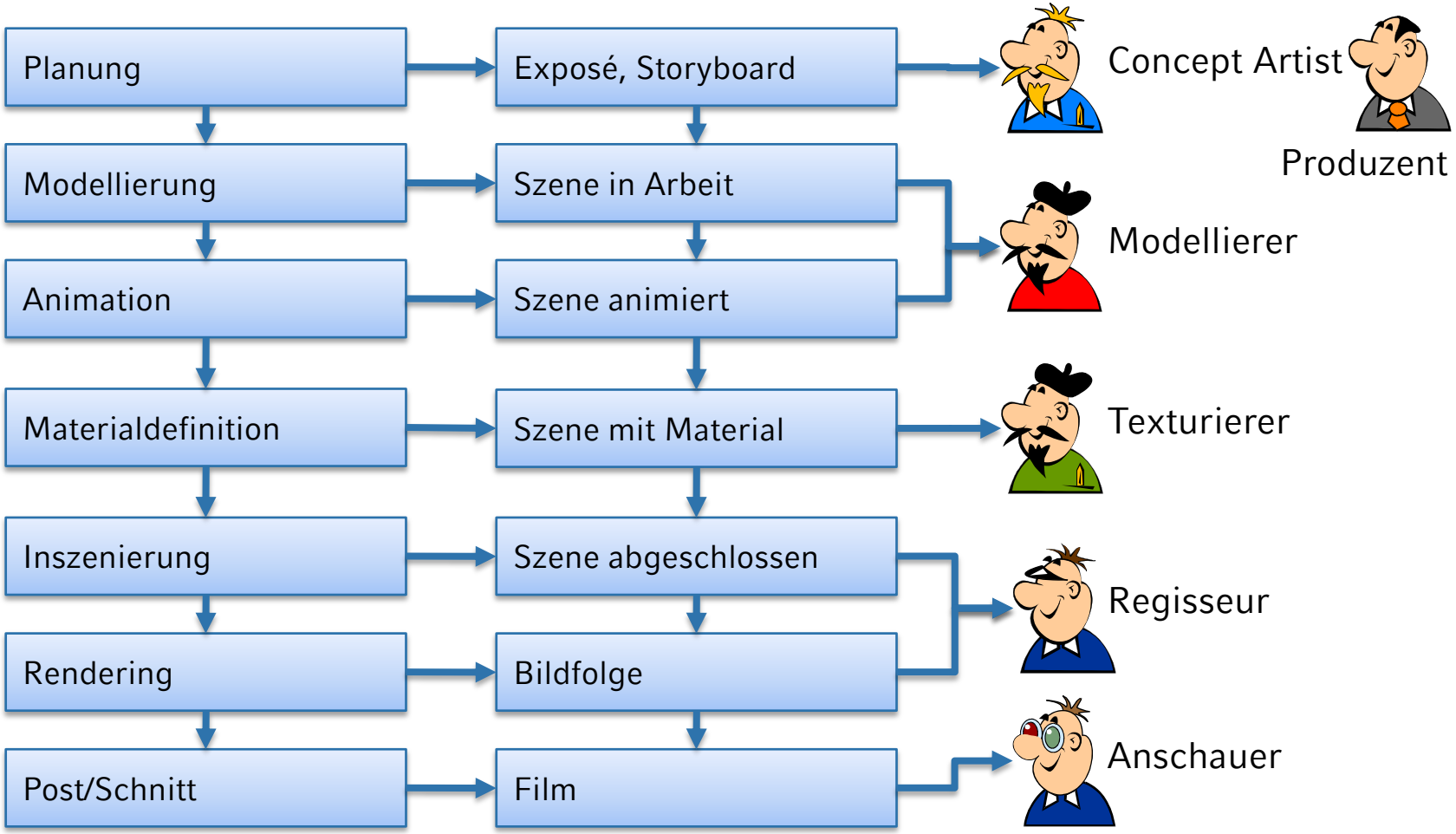
LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Von der Vision zum Bild

Bildproduktions-Prozess



Bildproduktionsprozess



Vor der Animation: Erstellung der Geometrien

- Konstruktion
  - Konstruieren mit 3D-Primitiven
  - Konstruktion mit Freiformflächen
  - Extrusions- und Rotationskörper
  - Zusammensetzen aus 2D-Formen
  - zusammengesetzte 3D-Körper
- Modellieren auf polygonaler Ebene



LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

„Belebtes“ 3D

Animations-techniken

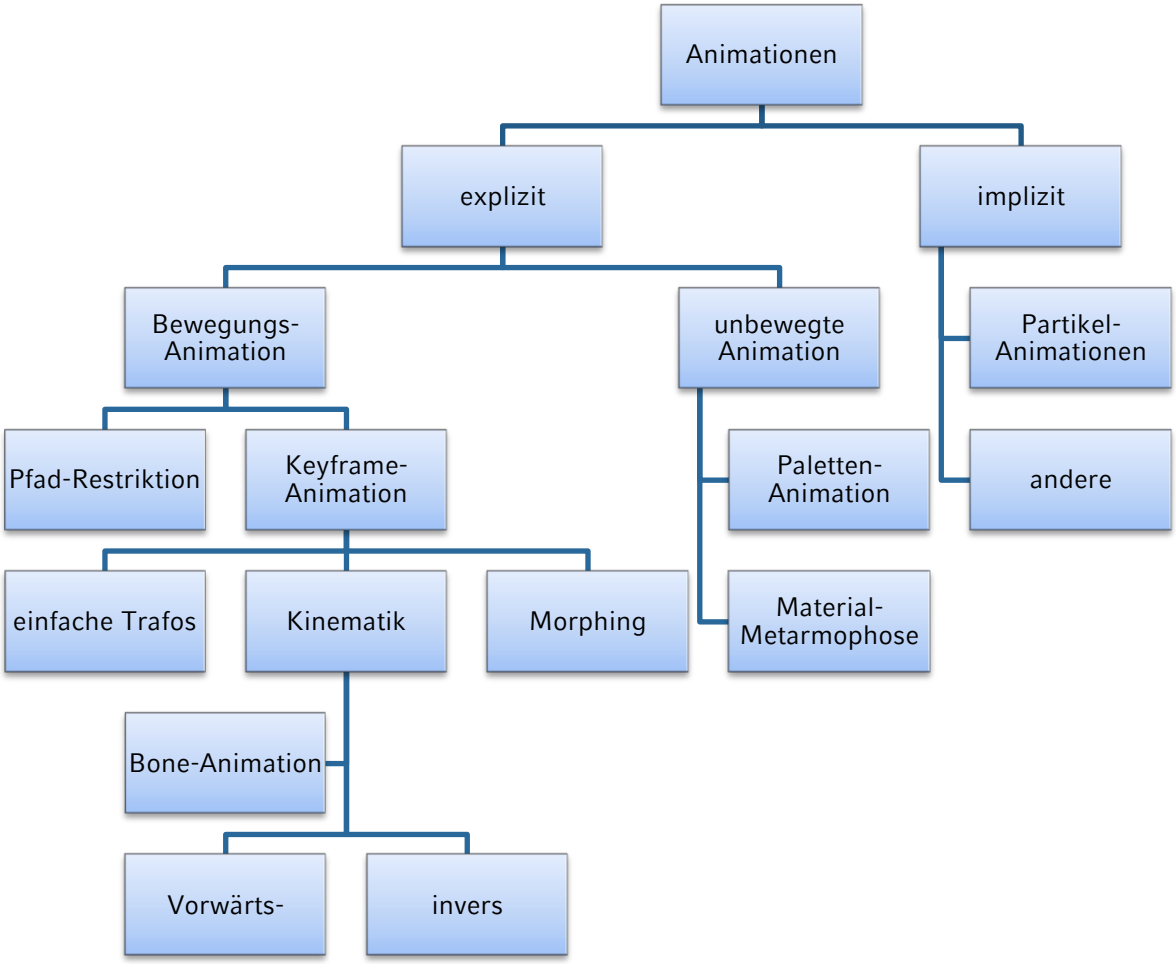


Animationen – Systematik

- Unterscheidung nach den Dimensionen:
  - 2D-Animation (Objekte und Kulisse)
  - 2½-Animationen (Eindruck von Räumlichkeit dank mehrerer Ebenen für u. a. Parallaxe-Scrolling)
  - 3D-Animationen



Animationstechniken



von 3D nach 2D

Sichtbarkeits-Bestimmung

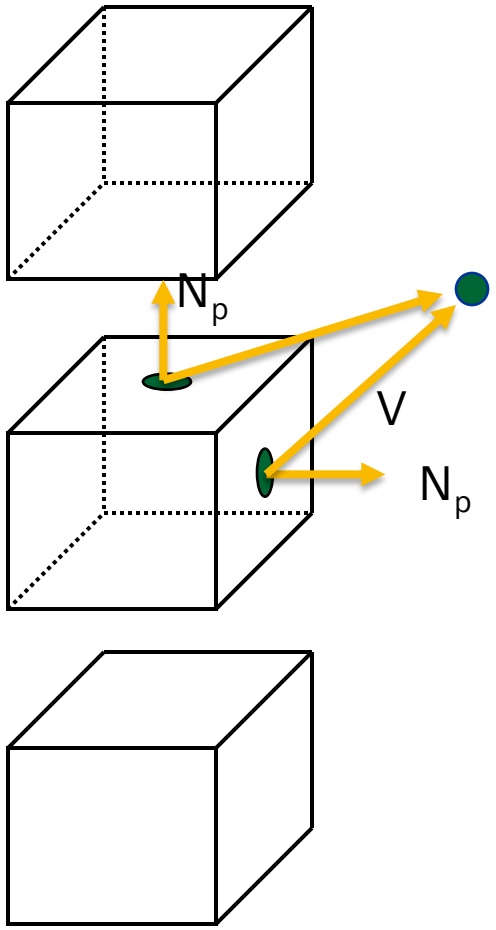


Sichtbarkeitsbestimmung: Entfernen verdeckter Rückseiten

- Polygone entfernen, die vom Betrachter wegzeigen (Rückseiten), da sie mit Sicherheit nicht sichtbar sind (engl.: *back face culling*)
- Generelles CG-Prinzip: Versuche mit einem schnellen (einfachen) Test eine aufwändigere Berechnung in vielen Fällen einzusparen.

Sichtbarkeitsbestimmung: Entfernen verdeckter Rückseiten

- Einfacher Test auf Sichtbarkeit eines Polygons mit Hilfe von:
  - Polygonnormale N_p (nach außen zeigend)
  - Vektor v(iew) vom Polygon zum Betrachter-Standpunkt
- Ist der Winkel > 90°, dann zeigt die Normale (und damit die Außenseite des Polygons) vom Betrachter weg und das Polygon ist nicht sichtbar.

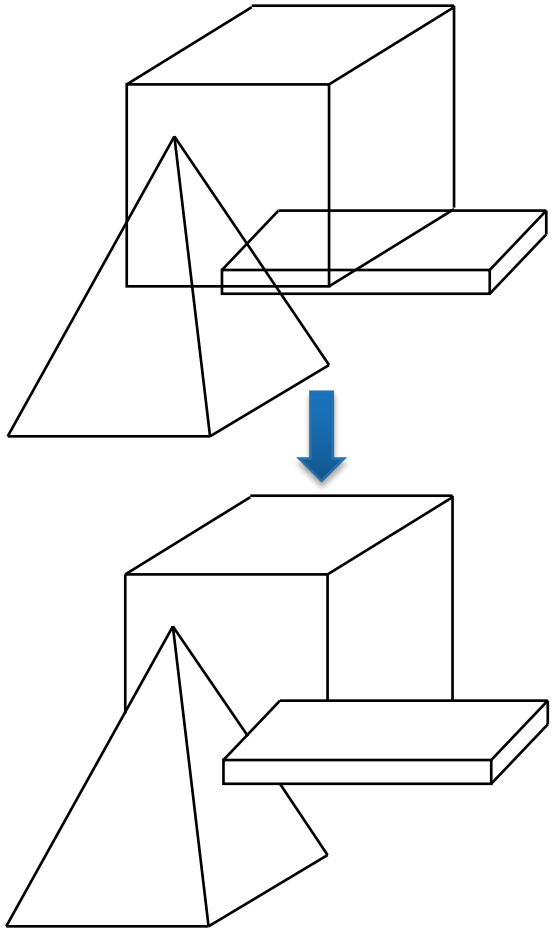


Sichtbarkeitsbestimmung: Entfernen verdeckter Rückseiten

- Backface Culling (Eliminieren von Polygonen mit einer vom Betrachter abgewandten Normale)
- Erster Schritt beim Entfernen verdeckter Polygone
- Vorgehen:
  - Bilde Skalarprodukt aus $N_p \cdot V$
  - Polygon ist prinzipiell sichtbar, wenn $N_p \cdot V > 0$ (und damit cos zwischen beiden Vektoren < 90°. Vorzeichentest ist schneller als Winkelberechnung)
  - wichtig: Vektoren vorher normalisieren
- Einfacher Test, weil Betrachterstandpunkt im Ursprung des Kamerakoordinatensystems
- Back face culling entfernt ~ 50% aller Polygone.

Sichtbarkeitsbestimmung: Entfernen verdeckter Rückseiten

- Bestimmen der Teile des Modells, die nicht von anderen Teilen des Modells verdeckt werden (oder der Teile, die verdeckt werden)
- Hidden Line bzw. Hidden Surface Removal



Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen

- Viele Verfahren zur Lösung der Aufgabe:
  - Unterschiedliche Herangehensweise
    - mit Objektgenauigkeit (d. h. analytische Verfahren in 3D)
    - mit Pixelgenauigkeit (d. h. Verfahren in „2D“)
  - Unterschiedliche Komplexität
  - Unterschiedliche Algorithmenklassen
  - Keine detaillierte Behandlung hier, nur depth-sort und z-Buffer-Algorithmus (später)

Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen

- Aus dem Kamara-Koordinatensystem werden die Koordinaten in der (2D-)Bildebene berechnet
- Projektionstransformation
- Änderung zu den bisher behandelten Transformationen: *z*-Werte bleiben erhalten, so dass jeder Objektpunkt einen Tiefenwert besitzt.
- Tiefenwerte werden für das Entfernen verdeckter Teile des Modelles beim *z*-Buffer-Algorithmus benötigt.

Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen

- Mögliche Konstellationen

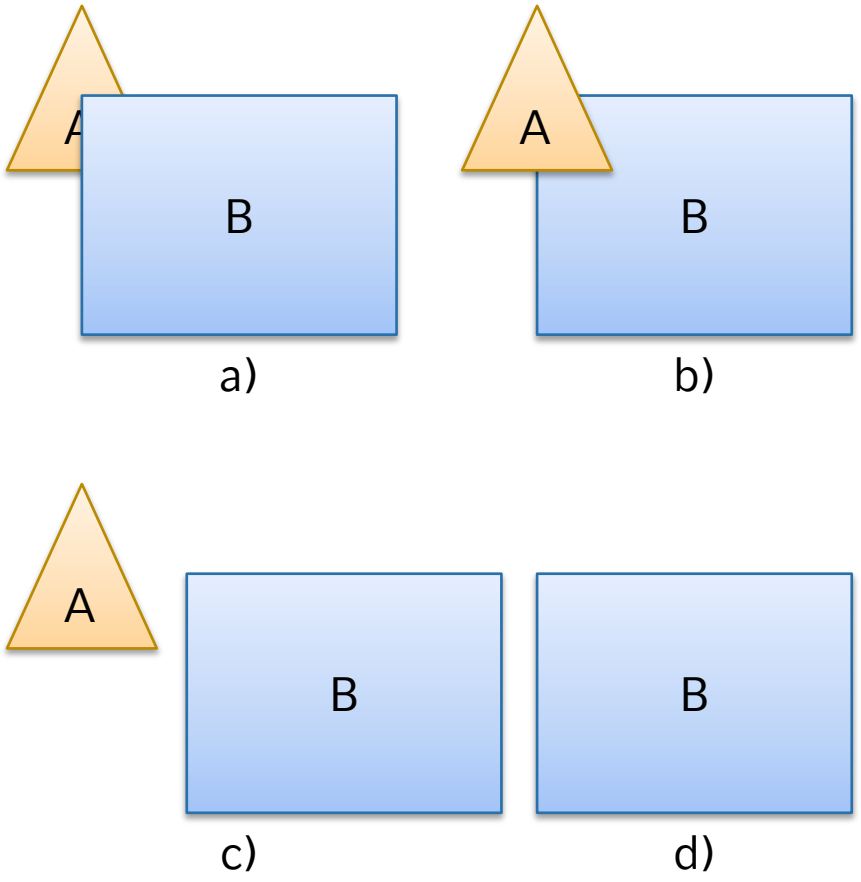
a) B vor A und Projektion B und A überlappt (B verdeckt A teilweise)

b) A vor B und Projektion A und B überlappt (A verdeckt B teilweise)

c) Projektion von A und B überlappt nicht (beide komplett sichtbar)

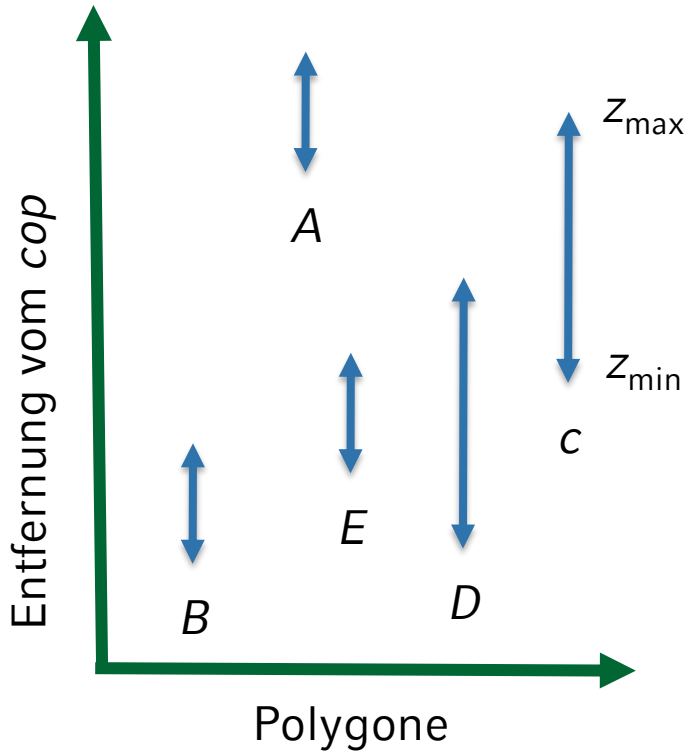
d) Ein Polygon vor dem anderen und Projektion des vorderen umschließt Projektion des hinteren (hinteres Polygon komplett verdeckt)

e) Projektion des vorderen Polygones komplett in der Projektion des hinteren enthalten (vorderes Polygon wird „über“ das hintere gezeichnet.)



Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen

- Depth-Sort (Objektbasierter Algorithmus)
- Sortieren der Polygone nach ihrer Entfernung zum *cop*
- Zeichnen von hinten nach vorn
- **Problem**: Bei Polygonen, bei denen sich die Abstände zum *cop* und die Projektion überlappen, ist die Zeichenreihenfolge unklar.



Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen

- **Lösung**:
  - Teile solche Polygone solange bis Polygonteile entstehen, die sich eindeutig sortieren lassen.
  - Dreiecke: Schwerpunkt bilden; 3 kleinere Dreiecke konstruieren

Sichtbarkeitsbestimmung: Z-Buffer-Algorithmus

- Effizienter Algorithmus zum Erkennen der sichtbaren Teile einer Szene, der in Bildschirmkoordinaten arbeitet (besser: im Bildraum)
- Ist in Hardware implementiert (NVidia, ATI, etc.).
- Voraussetzung: Alle Polygone liegen transformiert vor, wobei die Bildschirmkoordinaten eine transformierte z-Koordinate besitzen, die die Tiefe angibt.
- Technische Voraussetzung: zwei „Bitmaps“
  - Framebuffer enthält das eigentliche Bild
  - Z-Buffer enthält Tiefenwerte

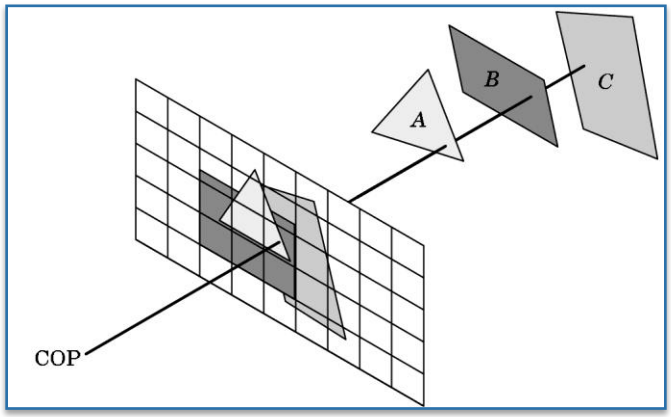
Sichtbarkeitsbestimmung: Z-Buffer-Algorithmus

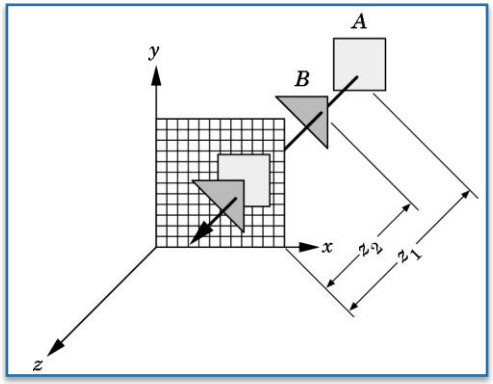
1. Fülle den Framebuffer mit der Hintergrundfarbe
2. Fülle den z-Buffer mit der maximal möglichen Tiefe
3. Zeichne alle Polygone nacheinander:
   - Bestimme für jeden Punkt des Polygons die Position im Bild, die Farbe und den Tiefenwert
   - Vergleiche den Tiefenwert an der Position mit dem, der dort im z-Buffer gespeichert ist
   - Wenn der Tiefenwert des Polygonpunktes kleiner ist (weiter vorne), dann zeichne die Farbe in den Framebuffer und setze den Tiefenwert im z-Buffer auf den neuen Tiefenwert, sonst wird nichts verändert

Sichtbarkeitsbestimmung: Z-Buffer-Algorithmus

- Was heißt „jeder Punkt des Polygons“?
  - Polygon wird abgetastet;
  - Berechnung wird für diskrete Punkte des Polygons durchgeführt.
  - Abtastrate hängt von der Größe des projizierten Polygones ab.

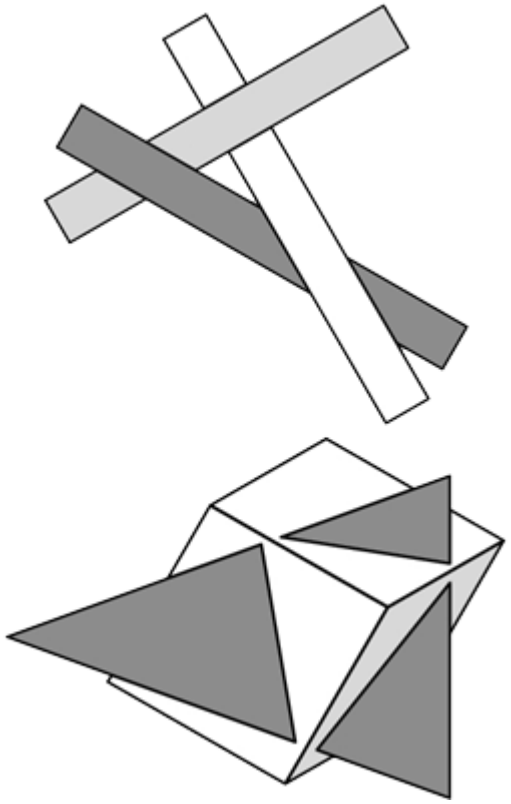
Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen





Sichtbarkeitsbestimmung: Entfernen verdeckter Kanten/Flächen

- Probleme bei der Sichtbarkeitsbestimmung:
  - Zyklisch überlappende Polygone
  - Ein Polygon „zersticht“ ein anderes (engl. *piercing*).
- Lösung: Teilen in solche Polygone, die sich eindeutig in Tiefenrichtung sortieren lassen (schwierig)



Zusammenfassung

- Entfernen von Rückseiten der Polygone (Normale und Vektor zur Kamera betrachten)
- Entfernen verdeckter Flächen(teile) durch bildbasierte oder objektbasierte Algorithmen
- Beispiele:
  - Objektbasiert: Sortierung der Polygone nach Abstand zur Kamera (Sortierreihenfolge nicht immer eindeutig)
  - Bildbasiert: z-Buffer

Weiter: Rasterung und Shading

- Nach der Projektion sind die Eckpunkte der Polygone in Bildschirmkoordinaten gegeben.
- Noch zu tun:
  - Umsetzung dieser Koordinaten auf Pixelpositionen
  - Bestimmen der entsprechenden Farbe des Pixels
  - „Zeichnen“ der Objekte = Rasterung
- Farbe des Pixels ist abhängig von:
  - Einfallendem Licht in der 3D-Szene
  - Material und Oberflächeneigenschaft der Objekte
  - Betrachterstandpunkt
- Bestimmen der Farbe an einem Punkt im 3D-Modell über Beleuchtungsmodelle, Umsetzen auf Pixelfarbe: Shading

von 3D nach 2D

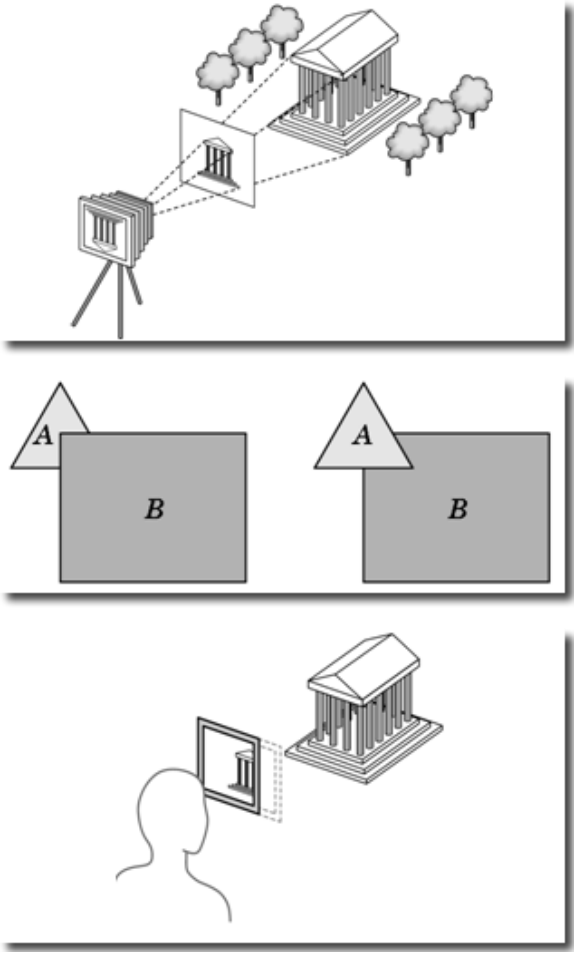
Clipping

Einleitung

- Motivation und Anwendung:
  - Clippen: Einleitung „Abschneiden“ einer geometrischen Beschreibung an einem Clipkörper.
- Anwendungen:
  - Fenstersysteme: Abschneiden der dargestellten Geometrie an den Fenstergrenzen (Bestandteil des Graphiksystems von Fenstersystemen)
  - Abschneiden von 3D-Geometrien am Sichtkörper (Pyramidenstumpf bzw. Quader)
  - Abschneiden eines verdeckten Polygons an einem davor liegenden Polygon (HSR)
  - Exploration von 3D-Daten mit Clipebenen

Einleitung: Anwendungen für Clipping

- *Oben:* Clippen am Sichtkörper
- *Mitte:* Clippen bei der Bestimmung verdeckter Kanten
- *Unten:* Clippen am Viewport



Einleitung

- Was heißt Clippen?

1. Entscheiden, welche Graphikprimitive komplett gezeichnet werden
2. Entscheiden, welche Teile von Graphikprimitiven gezeichnet werden

- → Schnittpunktberechnungen zwischen Graphikprimitiven und Clipgeometrie
- → Geschlossene Polygone bzw. Polyeder sind nach dem Abschneiden nicht mehr geschlossen. Berechnung von verbindenden Kanten/Flächen
- Beim Rendering muss das Clippen für jedes Polygon in 3d und für die sichtbaren Polygone nach Projektion in 2d durchgeführt werden. → Zahl der Schnittpunktberechnungen minimieren

Clippen von Liniensegmenten an Rechtecken

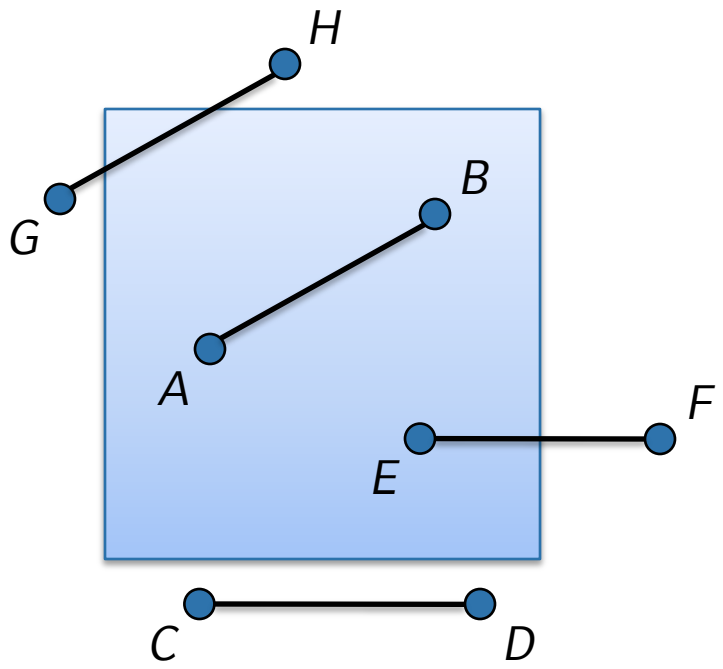
1. Punkte p_1 und p_2 innerhalb (*AB*)
   Linie komplett innerhalb (*accept*)

2. Punkte p_1 und p_2 außerhalb und jenseits der gleichen Kante (*CD*)
   Linie komplett außerhalb (*reject*)
   Bei 1. und 2. nutzt man Konvexität der Linie und des Clipkörpers

3. Punkte p_1 und p_2 außerhalb aber in Bezug auf verschiedene Kanten
   → testen, ob es zwei Schnittpunkte gibt (*GH*)

4. Punkte p_1 innerhalb p_2 außerhalb (*EF*) Berechne Schnittpunkt zwischen p_1, p_2 und dem Rechteck

5. Punkte p_1 außerhalb und p_2 innerhalb Vertauschen von p_1 und p_2 und dann vorgehen wie Fall 4



Clippen von Liniensegmenten an Rechtecken

- Test, ob Punkt p im achsenparallelen Rechteck (x_{min}, x_{max}, y_{min}, y_{max}):

```
If (p(x)>=xmin && p(x)<=xmax && p(y)>=ymin &&
p(y)<=ymax)

inside = true;
```

- Schnittpunktberechnungen:
  1. Identifizieren der geschnittenen Rechteckkante Tests separat durchführen, ggf. kommen zwei Kanten in Frage (siehe EF)
  2. Schnittpunktberechnung mit dieser (diesen) Kante(n)

Clippen von Liniensegmenten an Rechtecken

- Cohen-Sutherland-Algorithmus: Bestimme für jeden Eckpunkt (x, y) ein Bitfeld (outcode), das angibt, ob Eckpunkt jenseits des entsprechenden Wertes ist.
- 1. Bit: $y > y_{max}$,
- 2. Bit: $y < y_{min}$,
- 3. Bit: $x > x_{max}$,
- 4. Bit: $x < x_{min}$

1001	1000	1010
0001	0000	0010
0101	0100	0110

Clippen von Liniensegmenten an Rechtecken

- Cohen-Sutherland-Algorithmus:
- outcode (p1) und outcode (p2) seien die Bitfelder der Eckpunkte:

```
if (outcode (p1) == 0000 && outcode (p2) == 0000)
   accept; //p1 und p2 innerhalb
if (outcode (p1) & outcode (p2) != 0000) // bitweises „und"
   reject;
if (outcode (p1) | outcode (p2) != 0000) // bitweises oder
   calculate_intersection (p1, p2, x_min, x_max, y_min,
   y_max);
```

- Schnittpunktberechnung hängt von der Linienrepräsentation ab. Parametrische Darstellung günstig, weil keine Sonderfälle (z. B. vertikale Linien) auftreten.

Clippen von Liniensegmenten an Rechtecken

Parametrische Liniendarstellung:

$$p(t) = (1 - t)\, p_1 + t\, p_2$$

Für $t \geq 0$ und $t \leq 1 \rightarrow$ Linie von p_1 zu p_2.

Wenn es einen Schnittpunkt *schn* zwischen (p_1, p_2) und einer Rechteckkante (r_1, r_2) gibt, dann gibt es Parameter s und t, mit s in (0, 1) und t in (0, 1) so dass

$$schn = (1 - s)\, r_1 + s\, r_2 = p(t).$$

Lösung: Gleichungen für x und y-Komponente der Punkte aufstellen und gleichsetzen (2 Gleichungen; 2 Unbekannte)

Clippen von Liniensegmenten an Rechtecken

- **Beispiel:** Schnittpunkt-Berechnung der Linie (x_1, y_1) (x_2, y_2) mit der unteren Rechteck-Kante (x_{min}, y_{min}) (x_{max}, y_{min})
- Schnittpunkt-Berechnung an horizontalen und vertikalen Kanten vereinfacht sich

$$\text{I} \quad (1-t)x_1 + tx_2 = (1-s)x_{min} + sx_{max}$$

$$\text{II} \quad (1-t)y_1 + ty_2 = (1-s)y_{min} + sy_{max}$$

$$(1-t)y_1 + ty_2 = y_{min}$$

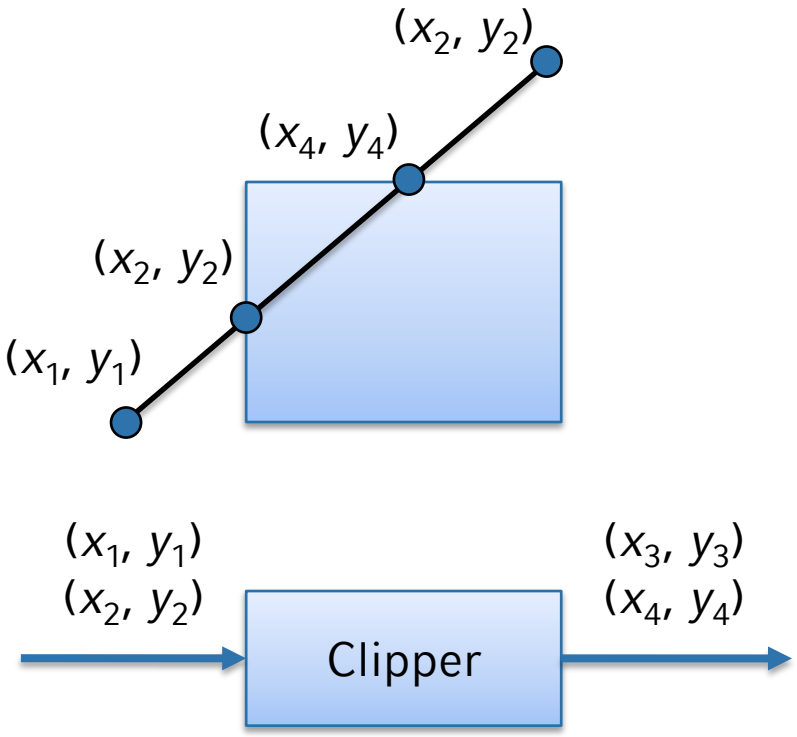
$$t - y_1 + ty_2 = y_{min} - 1$$

$$t = \frac{y_{min} - y_1}{y_2 - y_1}$$

Clippen von Liniensegmenten an Rechtecken

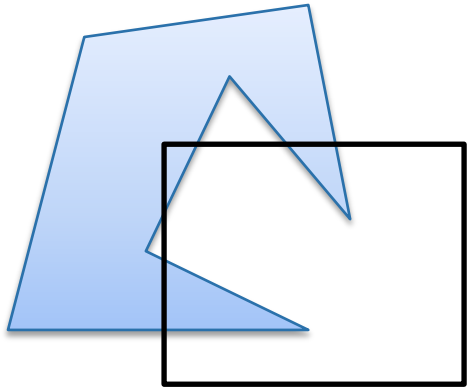
- Clipprozedur für eine Linie (x_1, y_1) (x_2, y_2) an einem Rechteck erzeugt neue Linie (x_3, y_3) (x_4, y_4), wobei
  - (x_3, y_3) und (x_1, y_1) sowie
  - (x_2, y_2) und (x_4, y_4)

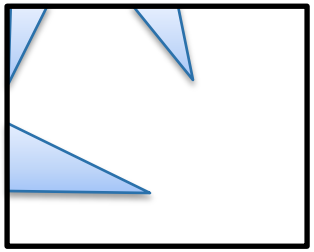
  gleich sein können



Clippen von Polygonen an Rechtecken

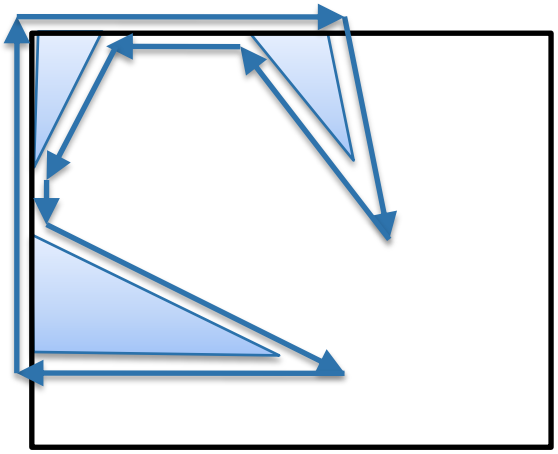
- Grundsätzlich kann das Clippen von Polygonen (p_1, p_2, …, p_n) an Rechtecken durch sukzessives Clippen der Linien (p_i, p_{i+1}) durchgeführt werden.
- **Problem:** Konkave Polygone → Durch Clippen kann ein konkaves Polygon in mehrere Polygone zerfallen.
- **Lösungsmöglichkeiten:**
  - Unterteilung des Polygons in konvexe Polygone (z. B. Triangulierung) oder
  - Einführen von zusätzlichen Kanten



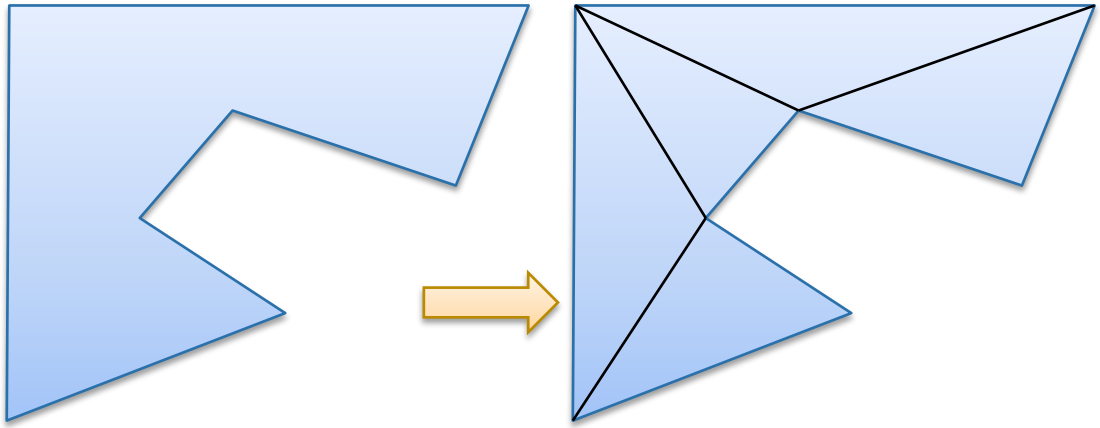


Clippen von Liniensegmenten an Rechtecken

Clippen von konkaven Polygonen



Einfügen von zusätzlichen Kanten, um geschlossenes Polygon zu repräsentieren



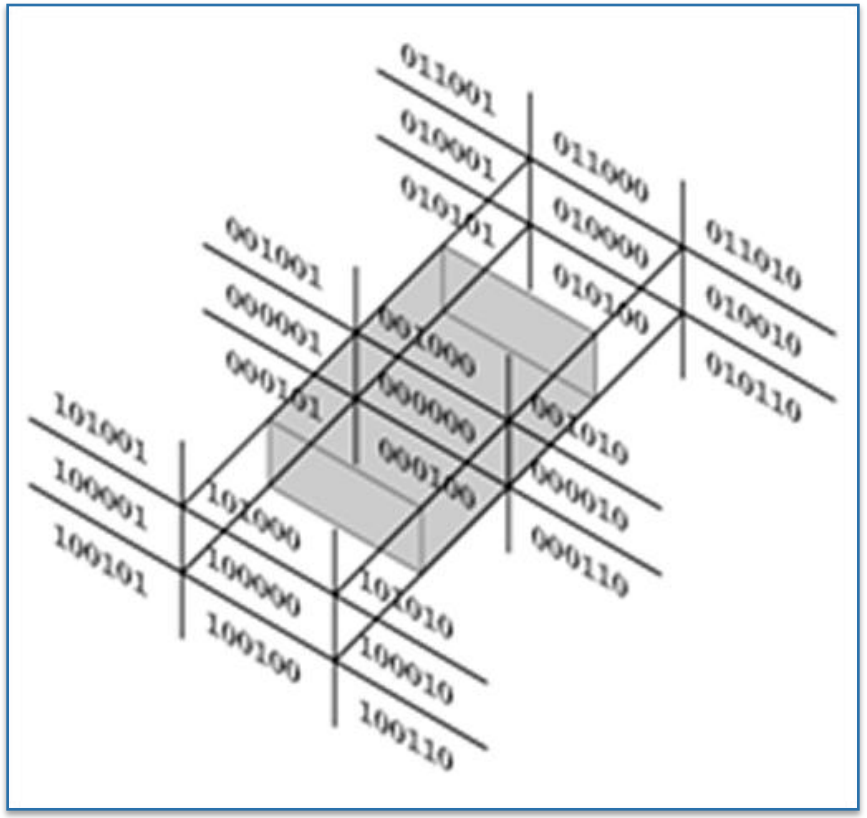
Triangulierung eines konkaven Polygons. Zusammenhang zwischen Dreiecken und Polygon muss repräsentiert werden

Clippen in 3D

- Wesentlicher Unterschied zwischen 2D und 3D:
- 2D: Schnittpunktberechnungen zwischen zwei Linien (Clipgeometrie durch Begrenzungslinien charakterisiert)
- 3D: Schnittpunkte liegen i. a. nicht auf den Kanten der Clipgeometrie. Schnittpunktberechnungen zwischen Linien und Flächen.

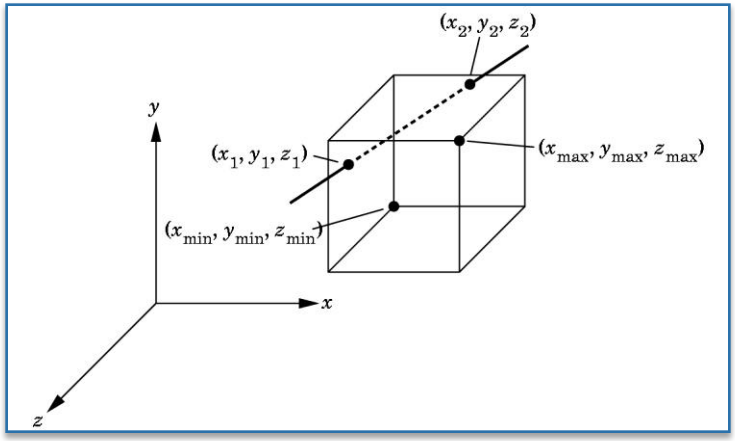
Clippen in 3D

- Erweiterung des Cohen-Sutherland-Algorithmus durch Ergänzen der Bitfelder pro Eckpunkt (zwei zusätzliche Bits repräsentieren Vergleich mit z_{min} und z_{max})



Clippen in 3D

- Für Schnittpunkt-Berechnung in parametrischer Form wird eine 3. Gleichung ergänzt, die sich auf die *z*-Koordinaten bezieht.



Clippen in 3D

- Clippen von Linien gegen achsenparallele Quader (Sichtbereich bei der Parallelprojektion). Vereinfachung wie im 2D-Fall.
- Clippen von Linien gegen Pyramidenstumpf (perspekt. Projektion): Scherung des Sichtkörpers → achsenparalleler Quader entsteht.
- Bei einer Pipeline-Architektur wird sukzessive gegen die sechs Begrenzungsflächen eines Quaders getestet.

Effizientes Clippen komplexer geometrischer Objekte

- Schnittpunktberechnungen sind sehr aufwändig für
  - Große Polygone
  - Parametrische Kurven
  - Text
- Intern wird Text häufig durch parametrische Kurven repräsentiert (Skalierbarkeit) und diese werden in Polygone umgewandelt.
- Schnittpunktberechnungen sind sehr effizient für einfache Formen (Kugeln, Quader).
- Effiziente Verfahren nutzen einfache (konvexe) Formen als Hüllkörper und testen, ob Hüllkörper geclippt werden muss.
- Nur wenn Hüllkörper teilweise innerhalb und teilweise außerhalb der Clipgeometrie ist, werden aufwändigere Tests durchgeführt (Raytracing).

Quellen

- Foley, van Dam, Feiner, Hughes. *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim. *Computergraphik 1*. Universität Magdeburg, Vorlesungsskript, Juli 2005.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

3D-Computergrafik

Rastergrafik

Notation von Polygonen

- Polygone werden nach 2D abgebildet
- Für weitere Betrachtungen reicht also zunächst:
  - Betrachtung 2D-Prmitiven
  - Erörtern der Aufgaben der Computergrafik im 2D-Fall

Vom Modell zum Bild: Vorüberlegungen

- Spezialfall: 2D-Computergraphik
- Bezug nehmend auf den allgemeinen Prozess der Bilderzeugung mit dem Rechner ergeben sich bei der 2D-Bilderzugung folgende Aufgaben:
  - Abbildung der „Vision eines Bildes“ eines Benutzers in einer geeigneten Modellbeschreibung im Rechner,
  - Überführung des Modells in eine Form, die am Bildschirm ausgegeben werden kann.

Vom Modell zum Bild: Vorüberlegungen

- Voraussetzung: Vision eines 2D-Bildes eines Benutzers durch Positionierung von 2D-Primitiven möglich
- Solche 2D-Primitive sind z. B.
  - Linie,
  - Kreis,
  - Ellipse,
  - Rechteck,
  - Vieleck (Polygon) etc.

Aufgabenstellung: Zielformulierung und Motivation

- Gesucht ist also grundsätzlich

1. eine Datenstruktur, die es erlaubt, die 2D-Primitive im Rechner abzubilden
2. spezielles Programm, dass die 2D-Primitive aus der Modellbeschreibung in die spezifische Beschreibung des Ausgabegeräts überführt

- Am weitesten verbreitet sind hierbei rasterorientiert arbeitende Ausgabegeräte.

Aufgabenstellung: Zielformulierung und Motivation

- Das bedeutet bei einer Ausgabe auf einem rasterorientierten Datensichtgerät (Monitor), Speicherung in einer Bitmap oder Ausgabe auf einen Drucker

1. die Überführung der „idealen“ 2D-Primitive aus der Modellbeschreibung in die entsprechenden Rasterprimitive des Ausgabegeräts und
2. die Transformation der der Koordinaten der 2D-Primitive aus dem Koordinatensystem des Modells in das gerätespezifische Koordinatensystem.

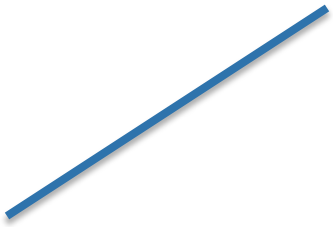
- Ausgegeben wird auf den rasterorientiert arbeitenden Ausgabegeräten ein Rasterbild.

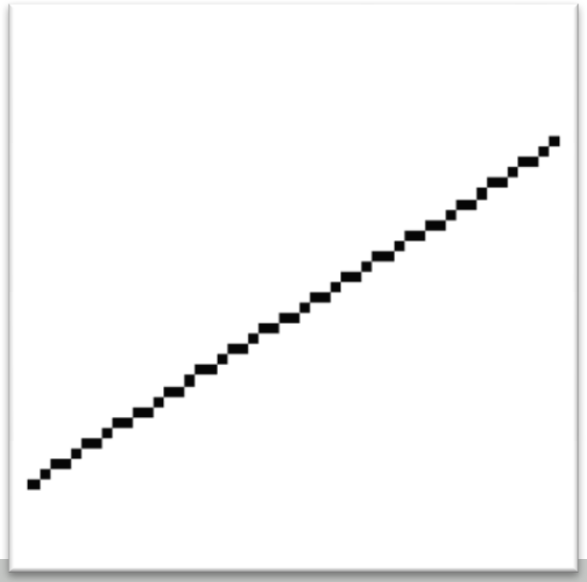
Rasterbilder – Vor- und Nachteile

- Vorteile:
  - Einfache Speicherung (Anordnung der Elemente)
  - Viele Nachbearbeitungsmöglichkeiten (vgl. Bildverarbeitung)

Rasterbilder – Vor- und Nachteile

- Nachteile:
  - Diskretisierung einer geometrischen Beschreibung erforderlich
  - Treppenstufen-Effekte (Aliasing) beim Transformieren (Vergrößern, Rotieren)
  - Hoher Speicherplatzbedarf





Rasterbilder – nähere Betrachtung

- Ein Rasterbild B als Matrix mit R Spalten und L Zeilen: $B = (b(x, y))$
- wobei x der Spaltenindex mit $0 \leq x \leq R - 1$ und
- y der Zeilenindex mit $0 \leq y \leq L - 1$ ist. Das Bildelement besitzt dabei den Wert $b(x, y)$.

Rasterbilder – nähere Betrachtung

- Der Farbwert c ist eine (irgendwie geartete) Kombination von Farbkanälen
- $c = K(v_0, \ldots, v_n)\ v \in V$
- wobei V das zu Grunde liegende Farbmodell repräsentiert.

Rasterbilder – nähere Betrachtung

- Ferner lässt sich ein Bild wie folgt betrachten:
- $B = (b(x, y, n))$
- wobei n der Kanalzähler für ein zu Grunde liegendes Farbmodell mit N Farbkanälen ist mit $0 \leq n \leq N$.

Anforderungen an Algorithmen

- Rasterungsalgorithmen werden oft gebraucht, sind daher zeitkritisch.
- Anforderungen:
  - keine aufwändigen Operationen (Multiplikation, Division sind aufwändiger als Addition),
  - möglichst mittels Integer-Arithmetik implementieren,
  - möglichst einfach strukturierte Algorithmen.

Vorausgehende Betrachtungen

- Wie sollen schließlich später die gerasterten Primitive vorliegen?
- Rasterbild ist aus Pixeln zusammengesetzt. „Pixel“ stammt vom englischen *picture element* und kennzeichnet die kleinste unteilbare Einheit eines Rasterbildes.
- Die Pixel im Rasterbild
  - sind gitterförmig angeordnet in Zeilen und Spalten,
  - sind alle gleich groß,
  - werden angesehen als Quadrate, die eine bestimmte Fläche überdecken,
  - alle Pixel eines Rasterbildes bedecken dieses vollständig (es gibt keine Lücken).

Zusammenfassung: Eigenschaften von Pixeln

- Position in der Bildmatrix,
  - angegeben in Koordinaten der Bildmatrix,
  - also als Wertepaar (Spalte, Zeile),
  - Grundlage hierfür bildet das Koordinatensystem der Bildmatrix
- beliebiger Farbwert,
- Größe (Ausdehnung) der Rechteckfläche des Pixels

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

2D-Rastergrafik

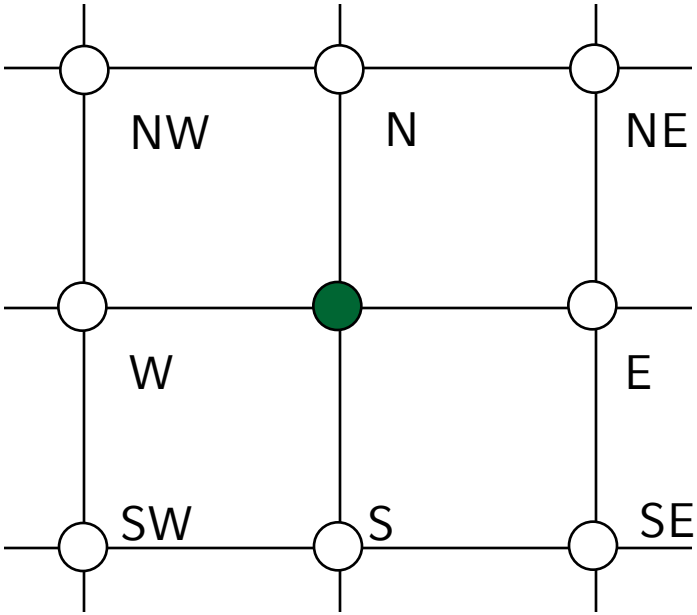
Einführung

Abstraktion: Pixelmodell für Entwurf von Algorithmen

- Konzentration auf das Wesentliche:
  - Bild besteht aus gleichmäßigem Gitter
  - An Gitterkreuzungspunkten befinden sich die Bildelemente
- Vereinfachung der Bildelemente:
  - keine Ausdehnung
  - sind ein- oder ausgeschaltet

Abstraktion: Pixelmodell für Entwurf von Algorithmen

- Bildelemente stehen in Relation zueinander:
  - Vierernachbarschaft
  - Achternachbarschaft
- Nachbarschafts-Beziehung wird gemäß der Himmelsrichtungen kodiert



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

2D Rastergrafik

Rastern von Linien

Anforderungen

- Linien sollten gerade aussehen
- Linien sollten gleichmäßig hell erscheinen
- Helligkeit sollte unabhängig von Richtung sein
- Endpunkte sollten exakt sein
- Linien sollen konstante Dicke haben
- Linienalgorithmus soll schnell sein

Linien mathematisch

- Gegeben: zwei Punkte $P_1(x_1, y_1)$ und $P_2(x_2, y_2)$
- Gesucht: Beschreibung der Linie zwischen P_1 und P_2
- setzen:
  - $\Delta x = x_2 - x_1$
  - $\Delta y = y_2 - y_1$
  - Anstieg $m = \Delta y / \Delta x$

Linie: Mathematische Beschreibung

- erste Möglichkeit: parametrische Beschreibung

$$y = x_1 + t(x_2 - x_1) = x_1 + t\Delta x$$

$$y = y_1 + t(y_2 - y_1) = y_1 + t\Delta y$$

- man erhält:

für $t = 0$ den Punkt (x_1, y_1)

für $t = 1$ den Punkt (x_2, y_2)

- Alle Punkte der Linie erhält man, wenn man für t alle Werte aus [0,1] einsetzt.

Linie: Mathematische Beschreibung

- zweite Möglichkeit: Beschreibung über Anstieg

$$y = mx + n$$

$$m = \Delta y / \Delta x$$

n: y-Koordinate des Schnittpunkts mit der y-Achse

- dritte Möglichkeit: implizite Beschreibung der Linie

$$F(x, y) = ax + by + c = 0$$

- Für alle Punkte auf der Gerade ist obige Gleichung erfüllt.

Naiver Algorithmus

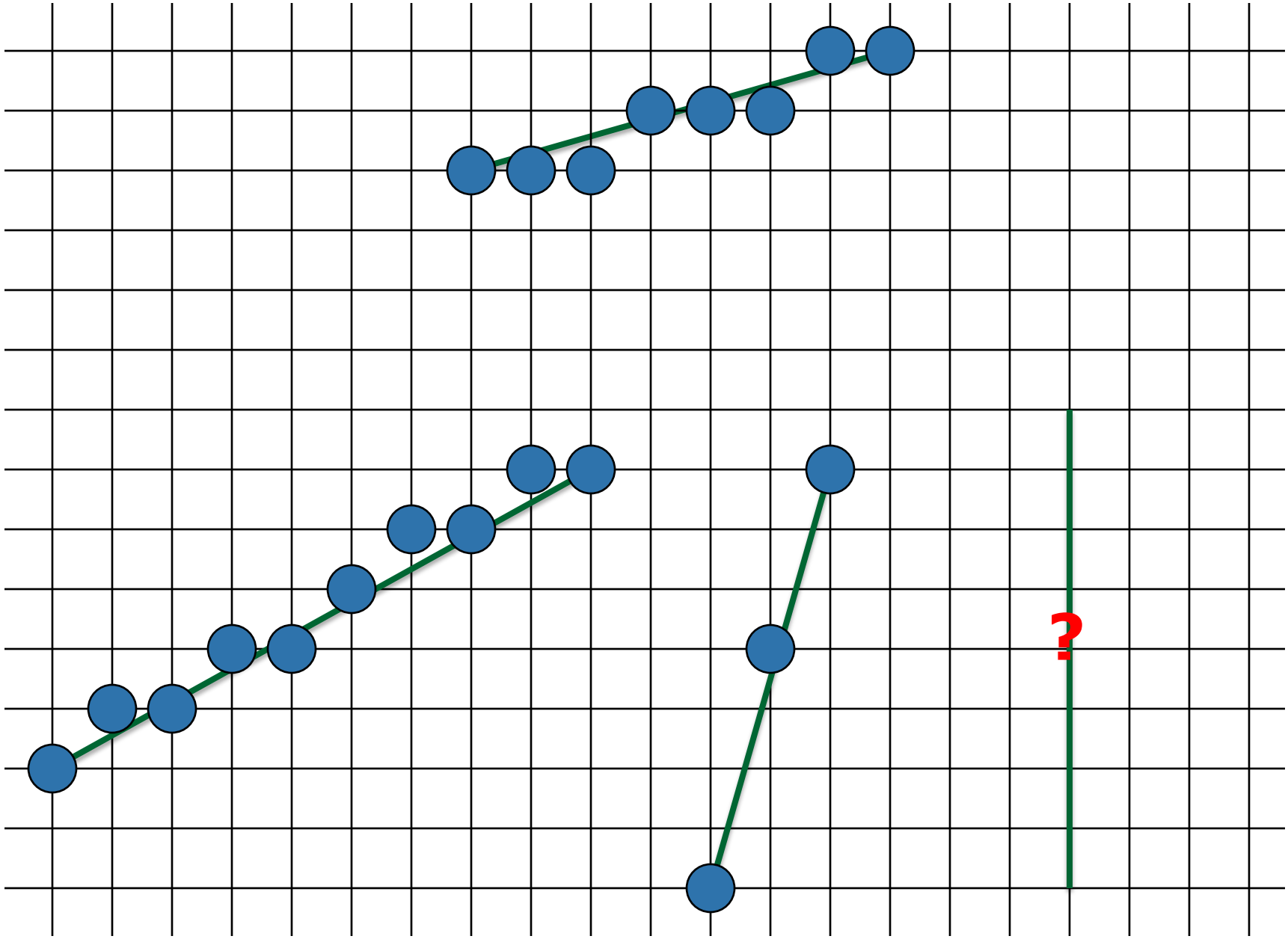
- Gegeben: zwei Punkte $P_1(x_1, y_1)$ und $P_2(x_2, y_2)$ jeweils Integer-Koordinaten
- Gesucht: alle Pixel, die auf der Linie liegen
- erster Schritt: Geradengleichung $y = mx + n$ aufstellen
- Einfachster Weg: Iteration von x_1 nach x_2 im Pixelabstand, entsprechende y-Werte berechnen, runden, zeichnen

```
double m = (y2-y1)/(x2-x1) ;
for (int x = x1; x <= x2; x++) {
    double y = m*x + n;
    setPixel( x, round( y ));
}
```

Naiver Algorithmus

```
double m = (y2-y1)/(x2-x1);
for (int x = x1; x <= x2; x++) {
 double y = m*x + n;
 setPixel( x, round( y ));
}
```

- Rechnen mit Gleitkommawerten bei *y* und *m*
- Division und Multiplikation verbrauchen Zeit
- Runden
- Was ist mit senkrechten Linien –*m* nicht definiert
- Aussehen der Linien bei verschiedenen Anstiegen



DDA (Digital Differential Analyzer)

- DDA = Digital Differential Analyzer
- Man wandert der Linie in kleinen Schritten entlang und zeichnet alle Pixel, in denen ein Schritt endet
- Schrittweite ≤ Pixelbreite, damit Keine Löcher
- Schrittweite erhält man z. B. durch mehrmaliges Halbieren der Gesamtlänge

DDA (Digital Differential Analyzer)

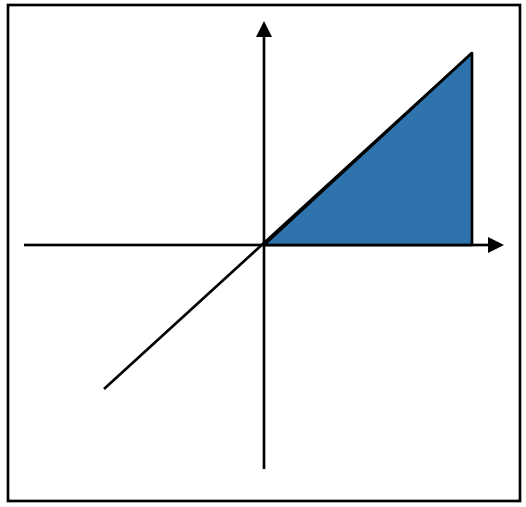
```
int Length, I;
double  X, Y, Xinc, Yinc;

Length = abs(X2 - X1);
if (abs(Y2 - Y1) > Length) Length = abs(Y2-Y1);
Xinc = (X2 - X1) / Length;
Yinc = (Y2 - Y1) / Length;
X = X1;
Y = Y1;
while(X < X2){
     setPixel(Round(X), Round(Y));
     X = X + Xinc;
     Y = Y + Yinc;
}
```

- Probleme:
  - Division
  - Runden

Bresenham-Algorithmus

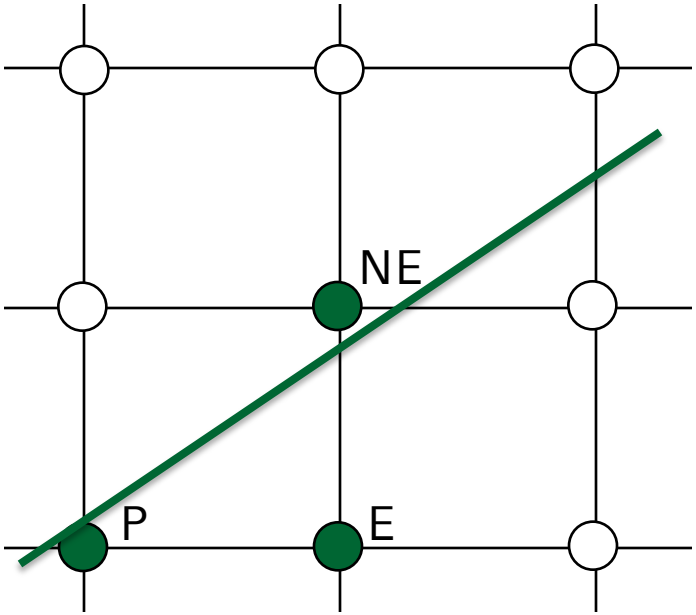
- auch Mittelpunktschema für Linien, Midpoint-Line-Algorithmus genannt
- Lösen der Probleme der anderen Ansätze, also
  - ausschließlich Integer-Arithmetik
  - keine Division, keine komplizierten Multiplikationen
- Einschränkungen:
  - Pixel liegen auf Integer-Gitter (keine wirkliche Einschränkung)
  - Anstieg der Linie kleiner 1, d. h. Linie befindet sich im 1. Oktanten ($0° \leq \alpha \leq 45°$) (auch keine Einschränkung, die eine Verallgemeinerung ausschließt)



Vorüberlegungen

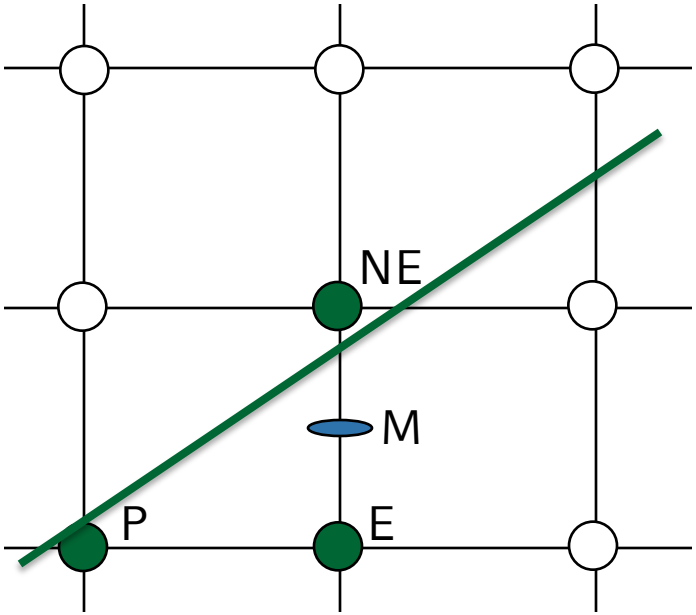
- implizite Geradengleichung $F(x, y) = ax + by + c = 0$
- liefert eine Antwort auf die Frage, auf welcher Seite der Geraden ein gegebener Punkt $P(x_1, y_1)$ liegt
  - ist $F(x_1, y_1) > 0 \rightarrow$ P liegt unterhalb der Geraden
  - ist $F(x_1, y_1) < 0 \rightarrow$ P liegt oberhalb der Geraden
  - ist $F(x_1, y_1) = 0 \rightarrow$ liegt auf der Geraden
- dabei ist:
  - $a = \Delta y$
  - $b = -\Delta x$

Mittelpunktschema für Linien



- Annahme: Steigung der Linie zwischen 0 und 1
- gerade gesetztes Pixel ist P
- Welches der beiden Pixel E oder NE muss als nächstes gesetzt werden?
- Entscheidung, welches der beiden Pixel näher am Schnittpunkt der Linie mit dem Pixelgitter liegt
- das für jedes Pixel

Mittelpunktschema für Linien

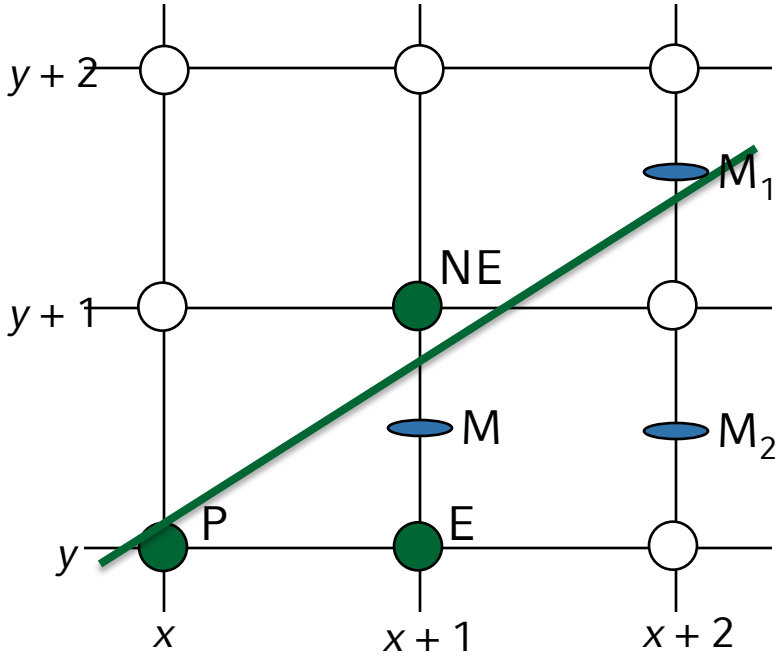


- leichter zu bestimmen: liegt der Mittelpunkt M zwischen E und NE oberhalb oder unterhalb der Linie
- momentane Position P(x, y)
- Koordinaten von M: M($x + 1$, $y + ½$)
- F(M) = F($x + 1$, $y + ½$)
- wenn F(M) > 0, dann geht Linie über M vorbei → NE nächstes Pixel
- wenn F(M) ≤ 0, dann geht Linie unter M vorbei → E nächstes Pixel
- wähle d = F(M)=F($x + 1$, $y + ½$) als *Entscheidungsvariable*

Mittelpunktschema für Linien

- Berechnen von F(M) für ein Pixel, Zeichnen des Pixels, Berechnen von F(M) für das neue Pixel
- Fragestellung: Kann F(M) für das neue Pixel aus dem F(M) für das gerade berechnete Pixel bestimmt werden?
  - möglicherweise einfachere Berechnungen
  - schrittweise Weitergehen von Pixel i zu Pixel $i + 1$
- Inkrementeller Algorithmus

Mittelpunktschema für Linien



- Fall 1: NE als nächstes Pixel ausgewählt
- $F(M) = F((x + 1), (y + ½))$
  $= a(x + 1) + b(y + ½) + c$
- dann ist M_1 der nächste Mittelpunkt
- $F(M_1) = F((x + 2), (y + 3/2))$
  $= a(x + 2) + b(y + 3/2) + c$
- Differenz zwischen beiden:
- $F(M_1) - F(M) = a + b$

→ $F(M_1) = F(M) + a + b$

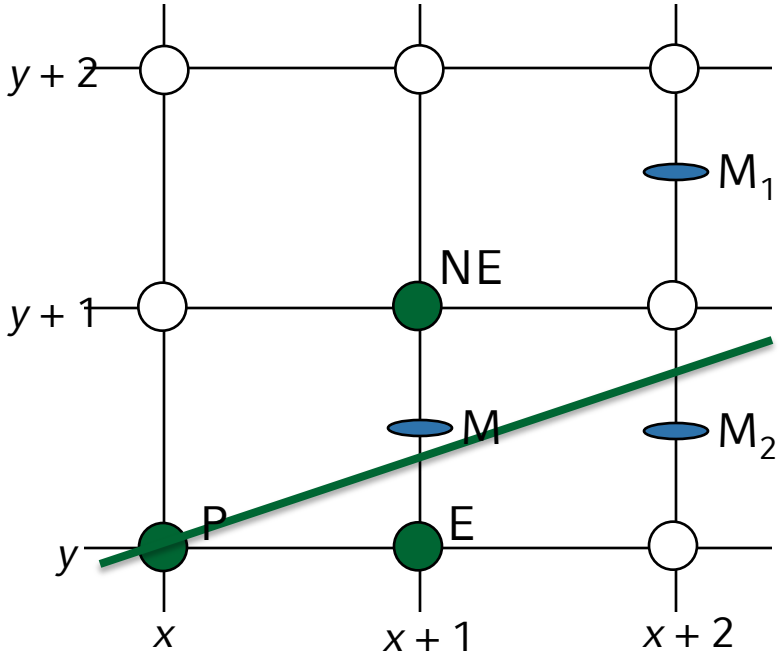
- mit $a = \Delta y$ und $b = -\Delta x$

→ $F(M_1) = F(M) + \Delta y - \Delta x$

→ $d = d + \Delta y - \Delta x$

→ $\Delta_{NE} = \Delta y - \Delta x$

Mittelpunktschema für Linien



- Fall 2: E als nächstes Pixel ausgewählt
- $F(M) = F((x + 1), (y + ½))$
  $= a(x + 1) + b(y + ½) + c$
- dann ist M_2 der nächste Mittelpunkt
- $F(M_2) = F((x + 2), (y + ½))$
  $= a(x + 2) + b(y + ½) + c$
- Differenz zwischen beiden:
- $F(M_2) - F(M) = a$

→ $F(M_2) = F(M) + a$

- mit $a = \Delta y$:

→ $F(M_1) = F(M) + \Delta y$

→ $d = d + \Delta y$

→ $\Delta_E = \Delta y$

Mittelpunktschema für Linien

- Idee für den Algorithmus:
  - beginne beim ersten Pixel
  - berechne die Entscheidungsvariable d=F(M) für dieses Pixel
  - entscheide, ob E oder NE das nächste Pixel ist
  - zeichne das Pixel
  - aktualisiere d entsprechend der Wahl von E oder NE
  - wiederhole, solange der Endpunkt nicht erreicht ist
- Offene Frage: Initialer Wert von d?

Mittelpunktschema für Linien

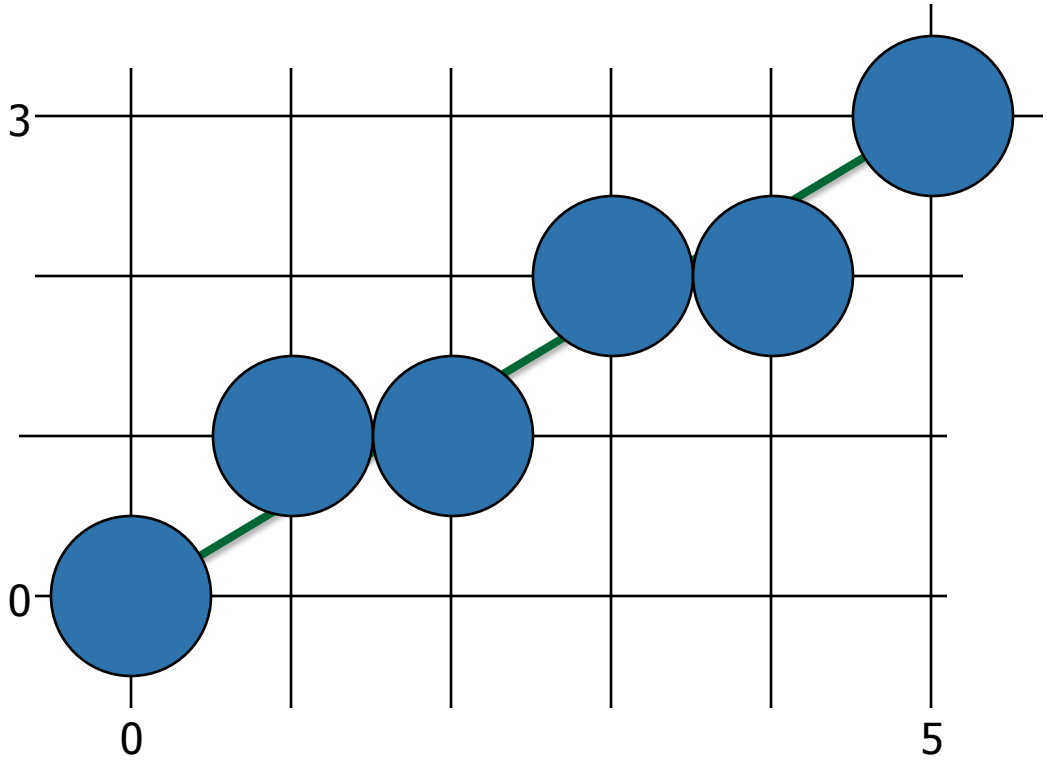
- Initialer Wert von d?
  - Startpunkt der Linie sei (x_0,y_0)
  - erster Mittelpunkt ist dann

$$\begin{aligned} F((x_0+1), (y_0+½)) &= a(x_0+1) + b(y_0+½) + c \\ &= ax_0 + a + by_0 + ½\,b + c \\ &= F(x_0, y_0) + a + ½\,b \end{aligned}$$

  - Startpunkt (x_0, y_0) liegt auf der Linie, also $F(x_0, y_0)=0$
  - Initialer Wert der Entscheidungsvariable $d = a + ½\,b$
- aber: keine Integer-Werte
  - Es interessiert nur das *Vorzeichen* von d.
  - Multiplikation von $F(x, y)$ mit 2 hat keinen Effekt auf das Vorzeichen
  - Initialer Wert der Entscheidungsvariable $d = 2a + b$
  - Inkremente: $\Delta_{NE} = 2\Delta y - 2\Delta x$ und $\Delta_E = 2\Delta y$

```
void midpoint-line(int xo, int yo, int xe, int ye, int color)
{
    int dx = xe - xo;
    int dy = ye - yo;
    int d = 2 * dy - dx;
    int incrE = 2 * dy;
    int icrNE = 2 * (dy - dx);
    int x = xo;
    int y = yo;
    setPixel(xo, yo, color);
    while (x<xe) {
          if (d<=0) {
                    d+=incrE;
                    x++;
          } else {
                    d += incrNE;
                    x++; y++;
          }
          setPixel(x, y, color);
    }
}
```

- Beispiel:
  Linie von (0,0) nach (5,3)
- Initiale Werte:
  $\Delta x = 5$, $\Delta y = 3$,
  $\Delta_E = 6$, $\Delta_{NE} = -4$
- Entscheidungsvariable d = 1



x=0, y=0, setPixel(0,0,color)
d=1>0 → Auswahl NE
→ $d=d+D_{NE}=1-4=-3$
→ x=x+1=1, y=y+1=1
x=1, y=1, setPixel(1,1,color)
d=-3<0 → Auswahl E
→ $d=d+D_E=-3+6=3$
→ x=x+1=2, y=1
x=2, y=1, setPixel(2,1,color)
d=3>0 → Auswahl NE
→ $d=d+D_{NE}=3-4=-1$
→ x=x+1=3, y=y+1=2
x=3, y=2, setPixel(3,2,color)
d=-1<0 → Auswahl E
→ $d=d+D_E=-1+6=5$
→ x=x+1=4, y=2
x=4, y=2, setPixel(4,2,color)
d=5>0 → Auswahl NE
→ $d=d+D_{NE}=5-4=1$
→ x=x+1=5, y=y+1=3
x=5, y=3, setPixel(5,3,color)

Zusammenfassung

- Algorithmus arbeitet ausschließlich mit:
  - Integer-Arithmetik
  - Additionen
- Inkrementeller Algorithmus, d. h.
  - Positionen der Pixel werden nach und nach aus denen der vorherigen berechnet
  - kleine Schleifenkörper
  - Implementierung in Hardware möglich

Probleme und Erweiterungen

- Beschränkung des Anstiegs auf $0° \leq m \leq 45°$
  - kein wirkliches Problem
  - Algorithmus kann erweitert werden auf alle möglichen Anstiege (Vertauschen von *x* und *y*, Vorzeichen, etc.)
- lange Linien werden Pixel für Pixel gezeichnet
  - Erweiterungen, die gleich zwei Pixel im Voraus berechnen
  - Linie setzt sich aus „Mustern“ zusammen → Muster finden und als Gesamtheit setzen

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Algorithmen der Rastergrafik

Rastern von Kreisen

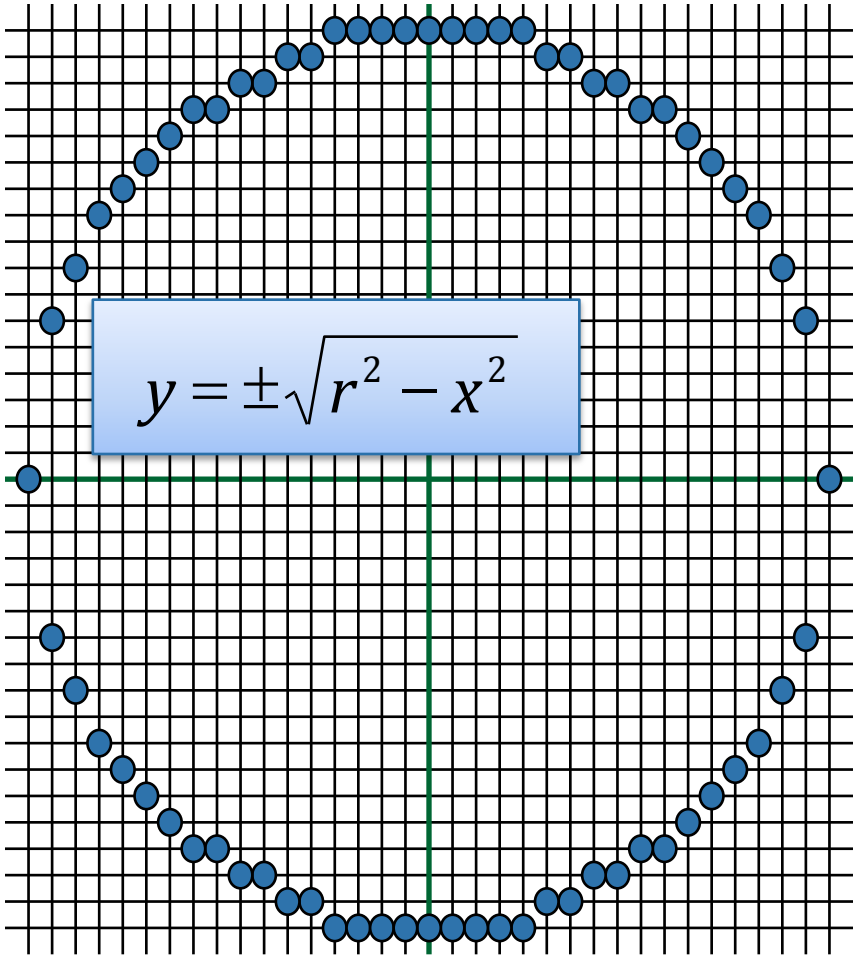
Kreise mathematisch

- Gegeben: Zentrum und Radius
- allgemeine Kreisgleichung für einen Kreis mit dem Zentrum im Ursprung: $F(x, y) = x^2 + y^2 = r^2$

  alle Punkte (x, y), die diese Gleichung erfüllen, liegen auf dem Kreis
- Kreisgleichung, wenn Mittelpunkt nicht im Ursprung liegt, sondern an der Position (x_c, y_c): $F(x, y) = (x - x_c)^2 + (y - y_c)^2 = r^2$

Naiver Algorithmus

- Auflösen der Kreisgleichung nach *y*:
- Schleife über alle *x* von $-r$ bis *r*
- Probleme:
  - aufwändige Berechnung
    - Wurzel
    - Quadrate
  - Lücken in der Umgebung von $|x| = r$

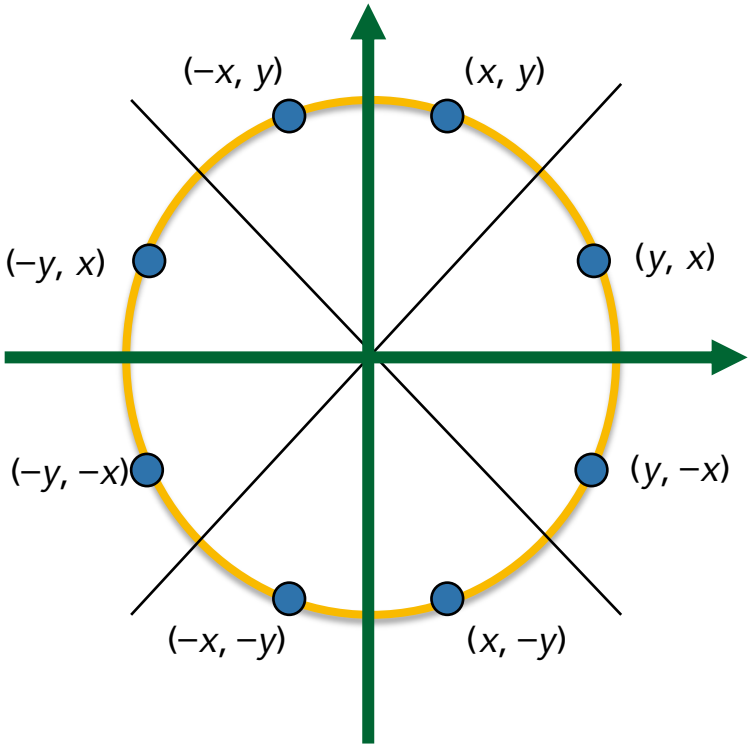


Parametrischer Algorithmus

- löst Problem der Lücken in den Bereichen nahe $|x| = r$
- Idee: Übergang zur Kreisgleichung in Parameterform

  $x = r \cos \varphi$ sowie $y = r \sin \varphi$
- Schleife über φ in gleichen, geeignet groß gewählten Schritten
- Probleme hier:
  - Berechnung der Winkelfunktionen nicht akzeptabel
  - Lösung über Lookup-Tabellen denkbar, jedoch immer noch ineffizient

Symmetrie des Kreises



Verringerung der Anzahl der Rechenoperationen durch Nutzung der Symmetrie des Kreises

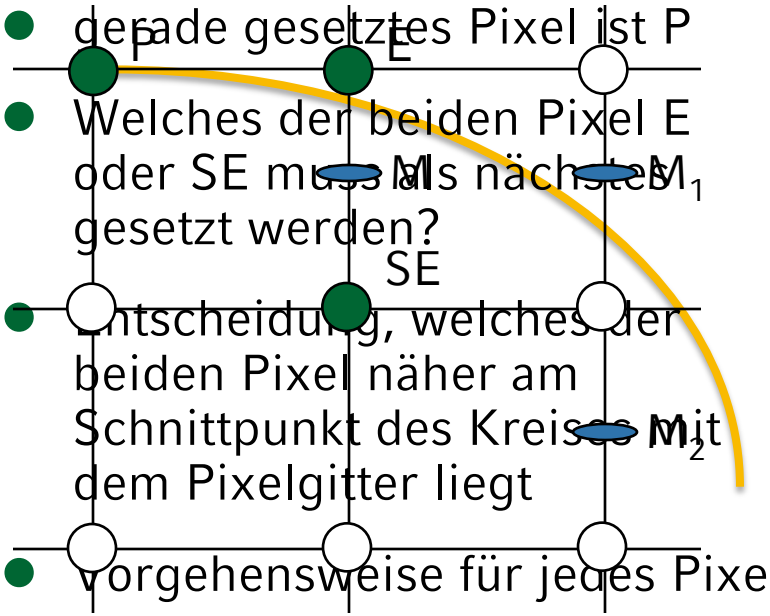
```
void circlePoints( int x, int y )
{
    setPixel( x, y );
    setPixel( y, x );
    setPixel( y, -x );
    setPixel( x, -y );
    setPixel( -x, -y );
    setPixel( -y, -x );
    setPixel( -y, x );
    setPixel( -x, y );
}
```

Bresenham-Algorithmus

- Herangehensweise wie bei Geraden
- Ziele:
  - Integer-Arithmetik
  - Eliminieren aufwändiger Operationen
  - Finden eines inkrementellen Algorithmus
- Basis: implizite Gleichung des Kreises
  - Die Funktion $F(x, y) = x^2 + y^2 - r^2$
  - ist Null für Punkte auf der Kreislinie
  - ist negativ für Punkte innerhalb des Kreises
  - ist positiv für Punkte außerhalb des Kreises

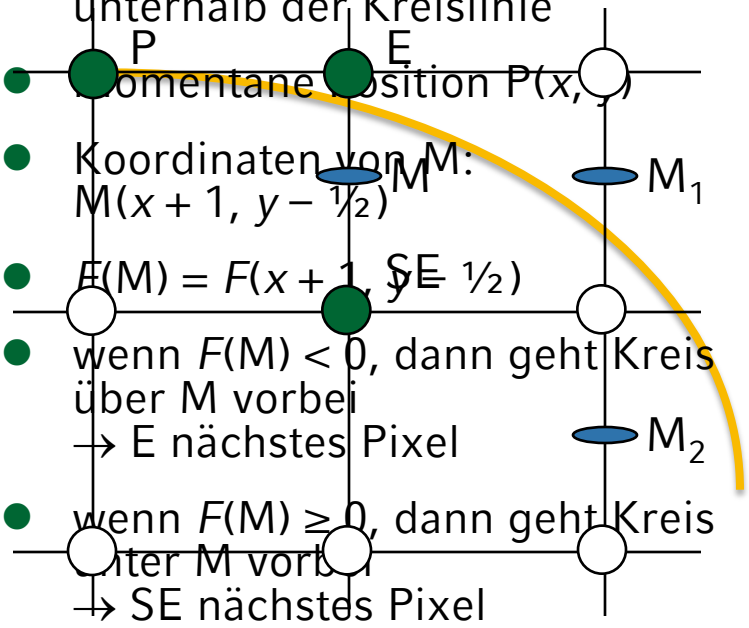
Grundidee

- Berechnung für den zweiten Oktanten
- derade gesetztes Pixel ist P
- Welches der beiden Pixel E oder SE muss als nächstes gesetzt werden?
- Entscheidung, welches der beiden Pixel näher am Schnittpunkt des Kreises mit dem Pixelgitter liegt
- Vorgehensweise für jedes Pixel
- erzeuge alle anderen Oktanten über Symmetrie



Grundidee

- Liegt M innerhalb oder außerhalb des Kreises entspricht in diesem Oktanten: liegt M ober- oder unterhalb der Kreislinie
- Momentane Position P(x, y)
- Koordinaten von M: M($x + 1$, $y - ½$)
- F(M) = $F(x + 1, y - ½)$
- wenn F(M) < 0, dann geht Kreis über M vorbei
  → E nächstes Pixel
- wenn F(M) ≥ 0, dann geht Kreis unter M vorbei
  → SE nächstes Pixel
- wähle d = F(M) = $F(x + 1, y - ½)$ als **Entscheidungsvariable**



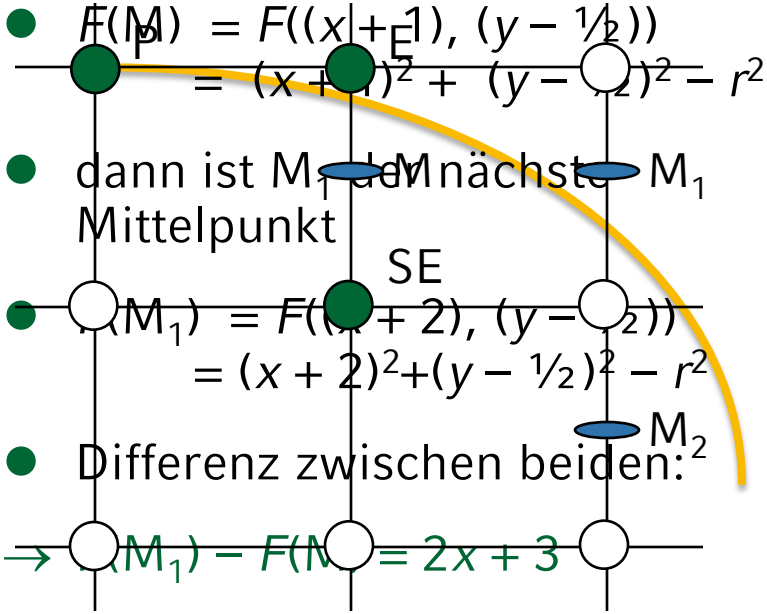
Grundidee

- Fall 1: $d < 0 \rightarrow$ E als nächstes Pixel ausgewählt
- $F(M) = F((x + 1), (y - ½))$
  $= (x + 1)^2 + (y - ½)^2 - r^2$
- dann ist M_1 der nächste Mittelpunkt
- $F(M_1) = F((x + 2), (y - ½))$
  $= (x + 2)^2 + (y - ½)^2 - r^2$
- Differenz zwischen beiden:

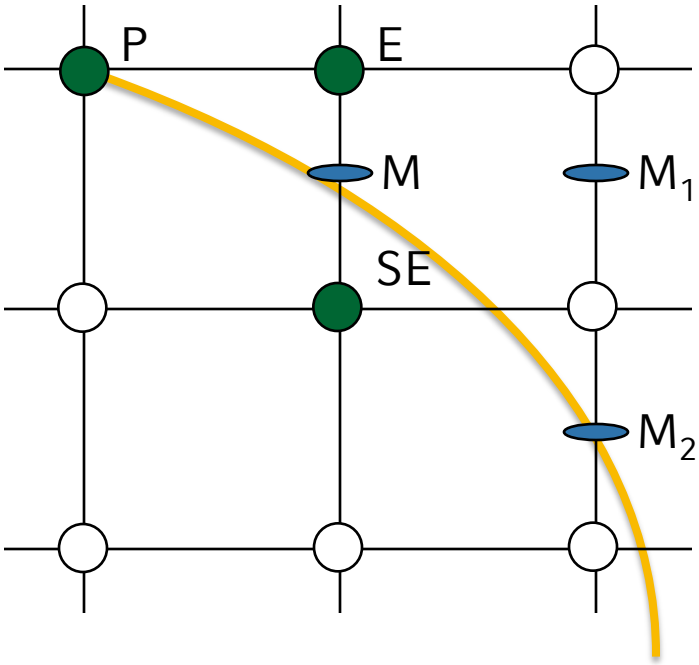
$\rightarrow F(M_1) - F(M) = 2x + 3$

$\rightarrow d = d + (2x + 3)$

$\rightarrow \Delta_E = (2x + 3)$



Grundidee



- Fall 2: $d \geq 0 \rightarrow$ SE als nächstes Pixel ausgewählt
- $F(M) = F((x + 1), (y - ½))$
  $= (x + 1)^2 + (y - ½)^2 - r^2$
- dann ist M_2 der nächste Mittelpunkt
- $F(M_2) = F((x + 2) = (y - {}^3/_2))$
  $= (x + 2)^2 + (y - {}^3/_2)^2 - r^2$
- Differenz zwischen beiden:

$\rightarrow F(M_2) - F(M) = 2x - 2y + 5$

$\rightarrow d = d + (2x - 2y + 5)$

$\rightarrow \Delta_{SE} = (2x - 2y + 5)$

Grundidee

- Initialer Wert von d
  - erstes Pixel des Kreises (besser: des berechneten Kreisausschnittes) ist $P(0, r)$
  - Erster Mittelpunkt ist damit $M(1, r - ½)$

    $$d = F(1, r - ½) = 1^2 + (r - ½)^2 - r^2 = {}^5/_4 - r$$
- allerdings: kein Integer-Wert
  - neue Entscheidungsvariable $h = d - {}^1/_4$
  - Anfangswert: $h_{start} = 1 - r$
  - Entscheidungskriterium: $d < 0$ wird zu $h < -¼$
  - da die Berechnung von h mit ganzzahligen Werten beginnt und h nur ganzzahlig (durch Δ_E, Δ_{SE}) inkrementiert wird, kann $h \leftarrow ¼$ durch $h < 0$ ersetzt werden

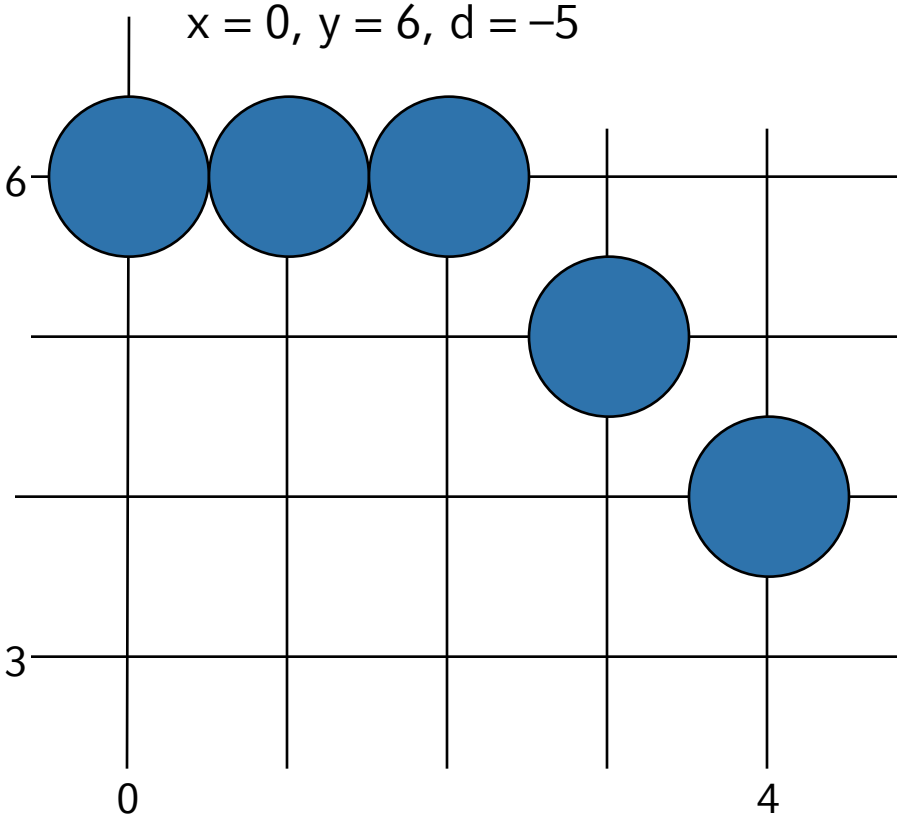
Algorithmus

```
void MidpointCircle (int radius)
{
    int x = 0; /* Initialisierung */
    int y = radius;
    int d = 1 - radius;
    circlePoints (x, y);

    while (y > x) {
           if (d < 0)
           { /* Auswahl von E */
                      d = d + 2  x +3;
                      x = x + 1
           } else { /* Auswahl von SE */
                      d := d + 2  (x-y) + 5;
                      x := x + 1;
                      y := y - 1;
           }
    circlePoints (x, y)
    }
}
```

Beispiel

- Beispiel:
  Kreis um (0,0) Radius 6
- Initialisierung:
  x = 0, y = 6, d = –5



x=0, y=6, d=-5, circlePoints(0,0)

d=-5<0 → Auswahl E
→ d=d+2x+3=-2
→ x=x+1=1

circlePoints(1,6)

d=-2<0 → Auswahl E
→ d=d+2x+3=5
→ x=x+1=2

circlePoints(2,6)

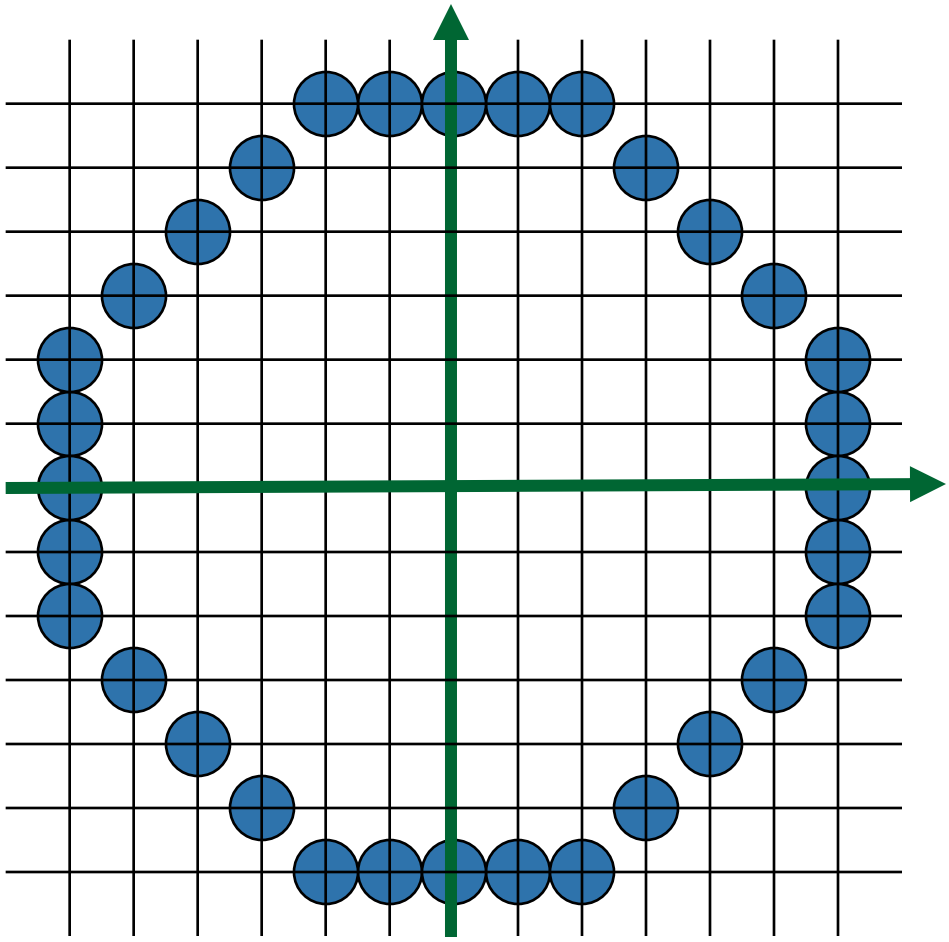
d=5>0 → Auswahl SE
→ d=d+2(x-y)+5=2
→ x=x+1=3, y=y-1=5

circlePoints(3,5)

d=5>0 → Auswahl SE
→ d=d+2(x-y)+5=3
→ x=x+1=4, y=y-1=4

circlePoints(4,4)

Beispiel



(x,y)–(y,x)–(y,-x)–(x,-y)–(-x,-y)–(-y,-x)–(-y,x)–(-x,y)

(0,6)–(6,0)–(6,-0)–(0,-6)–(-0,-6)–(-6,-0)–(-6,0)–(-0,6)
(1,6)–(6,1)–(6,-1)–(1,-6)–(-1,-6)–(-6,-1)–(-6,1)–(-1,6)
(2,6)–(6,2)–(6,-2)–(2,-6)–(-2,-6)–(-6,-2)–(-6,2)–(-2,6)
(3,5)–(5,3)–(5,-3)–(3,-5)–(-3,-5)–(-5,-3)–(-5,3)–(-3,5)
(4,4)–(4,4)–(4,-4)–(4,-4)–(-4,-4)–(-4,-4)–(-4,4)–(-4,4)

Erweiterungen

- Algorithmus ist relativ effizient, weil:
  - inkrementeller Algorithmus
  - Integer-Arithmetik
  - nur ein Achtel des Kreises muss berechnet werden
- Aber:
  - immer noch Multiplikationen notwendig, um die Inkremente zu berechnen
  - zur Erinnerung: $\Delta_E = (2x + 3)$ und $\Delta_{SE} = (2x - 2y + 5)$
  - Linienalgorithmus arbeitet mit konstanten Inkrementen
- Geht das hier auch?
  - Idee: Berechnen der Inkremente Δ_E und Δ_{SE} auch inkrementell
  - dazu: betrachten nicht nur einen Pixelschritt, sondern zwei

Differenzen zweiter Ordnung

- Fall 1: im aktuellen Schritt wurde E gewählt
  - Bewertungspunkt (aktuelles Pixel) wandert von (x, y) nach $(x + 1, y)$
  - Inkremente am „alten“ Pixel (x, y) waren:
    - $\Delta_{E_{alt}} = 2x + 3$
    - $\Delta_{SE_{alt}} = 2x - 2y + 5$
  - Inkremente am neuen Pixel wären:
    - $\Delta_{e_{neu}} = 2(x + 1) + 3$
    - $\Delta_{SE_{neu}} = 2(x + 1) - 2y + 5$
  - Differenzen zwischen beiden:
    - $\Delta_{e_{neu}} - \Delta_{E_{alt}} = 2$
    - $\Delta_{SE_{neu}} - \Delta_{SE_{alt}} = 2$

Differenzen zweiter Ordnung

- Fall 2: im aktuellen Schritt wurde SE gewählt
  - Bewertungspunkt (aktuelles Pixel) wandert von (x, y) nach $(x + 1, y - 1)$
  - Inkremente am „alten“ Pixel (x, y) waren
    - $\Delta_{E_{alt}} = 2x + 3$
    - $\Delta_{SE_{alt}} = 2x - 2y + 5$
  - Inkremente am „neuen“ Pixel wären:
    - $\Delta_{E_{neu}} = 2(x + 1) + 3$
    - $\Delta_{SE_{neu}} = 2(x + 1) - 2(y - 1) + 5$
  - Differenzen zwischen beiden:
    - $\Delta_{E_{neu}} - \Delta_{E_{alt}} = 2$
    - $\Delta_{SE_{neu}} - \Delta_{SE_{alt}} = 4$

Algorithmus

```
void MidpointCircle(int radius)
{
     int x = 0;
     int y = radius;
     int d = 1 - radius;
     int deltaE = 3;
     int deltaSE = -2 * radius + 5;
     circlePoints(x, y); /* draws 8 points */
     while (y > x) {
             if (d < 0) { /* select E */
                          d += deltaE;
                          deltaE += 2;
                          deltaSE += 2;
             } else { /* select SE */
                          d += deltaSE;
                          deltaE += 2;
                          deltaSE += 4;
                          y--;
             }
             x++;
             circlePoints(x, y);
     }
}
```

LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

Algorithmen der Rastergrafik

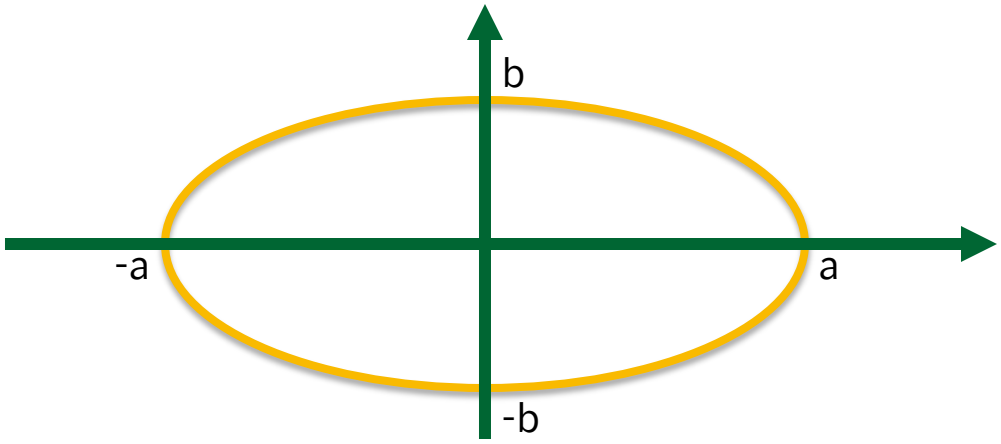
Rastern von Ellipsen

Grundlegendes

- Gleichung der Ellipse mit Mittelpunkt im Ursprung

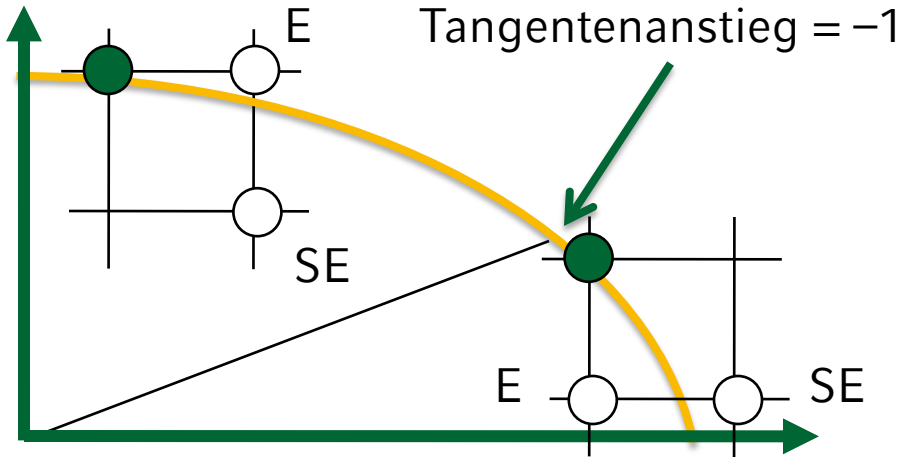
$$F(x, y) = b^2 x^2 + a^2 y^2 - a^2 b^2 = 0$$

- hier nur „achsenparallele“ Ellipsen
- Algorithmus für ersten Quadranten reicht aus, Rest durch Spiegelungen
- Ziel: inkrementeller Mittelpunkt-Algorithmus



Herleitung für den Algorithmus

- Schwierigkeit: zwei Gebiete mit
  - entweder Auswahl zwischen E und SE
  - oder Auswahl zwischen SE und S
- Grenze: Punkt, an dem die Ellipse den Anstieg –1 hat
- hier ohne Herleitung:
  - Übergang, der erste Mittelpunkt, für den gilt:
  - $y^2(y - ½) \leq b^2(x - 1)$



Herleitung für Algorithmus

- Gebiet 1: Mittelpunkt liegt zwischen E und SE

$$d_{alt} = F(x+2,\ y-½) = b^2(x+1)^2 + a^2(y-½)^2 - a^2b^2$$

  - Wahl von E als nächstes Pixel:

$$d_{neu} = F(x+2,\ y-½) = b^2(x+2)^2 + a^2(y-½)^2 - a^2b^2$$

$$\Delta_E = d_{neu} - d_{alt} = b^2(2x+3)$$

  - Wahl von SE als nächstes Pixel:

$$d_{neu} = F(x+2,\ y-3/2) = b^2(x+2)^2 + a^2(y-3/2)^2 - a^2b^2$$

$$\Delta_E = d_{neu} - d_{alt} = b^2(2x+3) + a^2(-2y+2)$$

- Gebiet 2: Mittelpunkt liegt zwischen SE und S – analog

Herleitung für Algorithmus

- Initialer Wert für d
  - es ist $(0, b)$ das erste Pixel, damit $(1, b - ½)$ der erste Mittelpunkt

    $$F(1, b - ½) = b^2 + a^2(b - ½)^2 - a^2b^2 = b^2 + a^2(-b + ¼) = d$$

  - Beim Wechsel der Gebiete ist ein neuer Initialwert für d zu berechnen, er liegt zwischen S und SE bei $(x + ½, y - 1)$

    $$F(x + ½, y - 1) = b^2 (x + ½)^2 + a^2 (y - 1)^2 - a^2b^2 = d$$

  - sonst Algorithmus wie beim Kreis
  - Ausnutzen der vierfachen Symmetrie

Algorithmus

```
void MidpointEllipse(int a, int b, int color)
{
     int x = 0; int y = b;
     double d1 = b*b-a*a*b-(a*a/4);
   ellipsePoints( x, y );
     while( a*a*(y-0.5) > b*b*(x-1)) {
             if d1 < 0 { d1 += b*b*(2*x+3)}
             else {
                          d1 += (b*b*(2*x+3)+a*a*(-2*y+2));
                          y--;
             }
             x++
             ellipsePoints( x, y );
     }
     d2 = b*b*(x+.5)*(x+.5)+a*a*(y-1)*(y-1)-a*a*b*b;
     while(y > 0) {
             if d2 < 0 {
                          d2 += b*b*(2*x+2)+a*a*(-2*y+3);
                          x++;
             } else { d2 += a*a*(-2*y+3) }
             y--;
             ellipsePoints( x, y );
     }
}
```

Zusammenfassung Ellipse

- Algorithmus kann natürlich noch erweitert werden:
  - Ellipsen nicht im Ursprung (trivial)
  - Ellipsen nicht achsenparallel (schwierig)
  - Differenzen zweiter Ordnung (trivial)
  - Float-Berechnungen (schwieriger)
- aber auch hier
  - inkrementeller Algorithmus
  - relativ einfache Berechnungen
- Aufgrund der Eigenschaften der Ellipsen machen sich eigenständige Algorithmen für sehr schmale Ellipsen (*a* oder *b* sehr klein) bezahlt.

Zusammenfassung Rasteralgorithmen

- werden oft aufgerufen, daher besondere Anforderungen:
  - schnell
  - einfach
  - Integer-Arithmetik
- inkrementelle Algorithmen
  - nach dem Mittelpunkt-Schema
  - Linien, Kreise, Ellipsen
  - erweiterbar für allgemeine Lagen der Primitive
- Beispiel für Herleitung von Algorithmen in der Rastergraphik

Offene Fragen

- Rasterung allgemeiner Kurven → würde Rahmen sprengen
- Bisher waren alle Linien genau ein Pixel breit, was passiert, wenn man dickere Linien haben möchte? → nächstes Kapitel
- Beim Rastern wurden bisher alle Pixel mit einer Farbwert gesetzt, das führt zu unschönen Artefakten – Aliasing – Wie kann man dem begegnen? → Antialiasing – übernächstes Kapitel
- Was passiert mit dem Inneren von Figuren – Binnenstrukturen? → Füllen – überübernächstes Kapitel

Rasteralgorithmen

Breite bzw. dicke Primitive

Breite Primitive

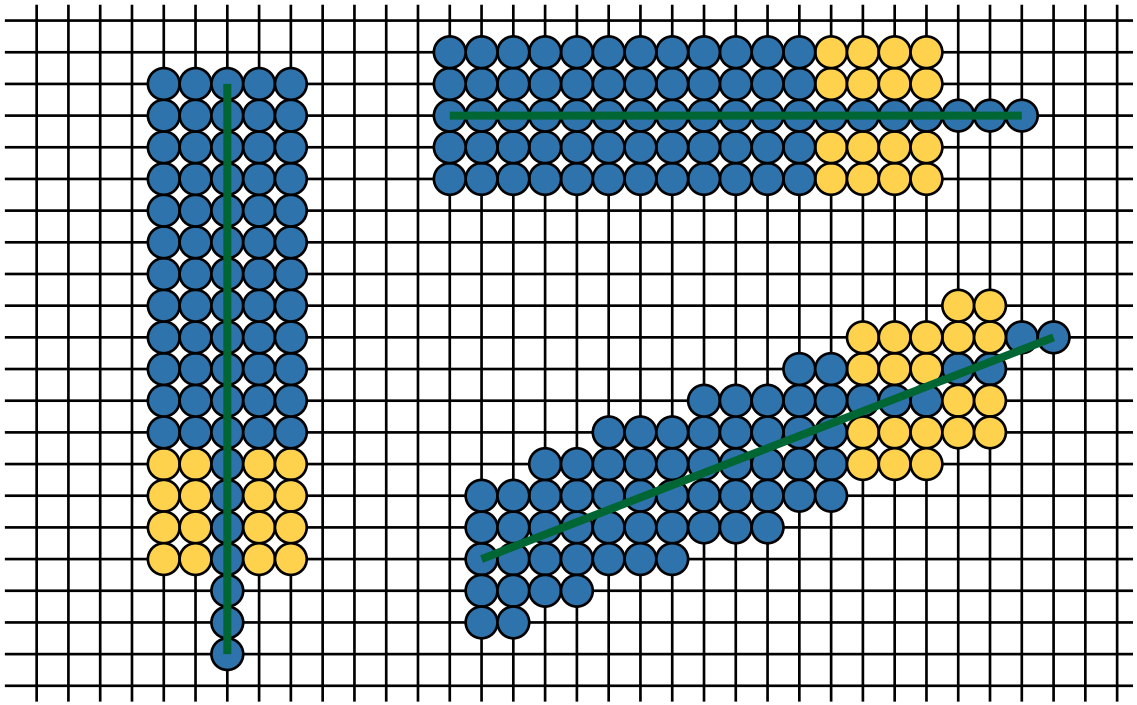
- bisher:
  - Alle Linien sind genau 1 Pixel breit.
  - vorgestellte Algorithmen eignen sich sehr gut dafür
- aber:
  - Wie wird verfahren, wenn die Linien dicker sein sollen?
  - Welche Probleme treten dabei auf?
  - Wie kann man diese lösen?

Breite Primitive

- *Ansatz:* anstelle eines Pixels werden an jedem Punkt mehrere Pixel gezeichnet – Fragen:
  - Wie sieht die Pixelgruppe aus, die gezeichnet werden soll (rund, rechteckig, etc.)
  - Wie ist die Orientierung der Pixelgruppe – insbesondere, wenn sie nicht rund ist?
  - Wie sehen die Linienenden aus?
  - Wie sehen Knotenpunkte aus, an denen zwei Linien aneinanderstoßen?
- Drei Methoden für breite Primitive:
  - Pixelwiederholung
  - Bewegliche Pinsel
  - Ränder zeichnen und dazwischen füllen

Pixelwiederholung

- Duplizieren von Pixeln in Zeilen oder Spalten
- funktioniert gut für Linien
  - spaltenweises Duplizieren für Anstieg $-1 < m < 1$
  - zeilenweises Duplizieren sonst

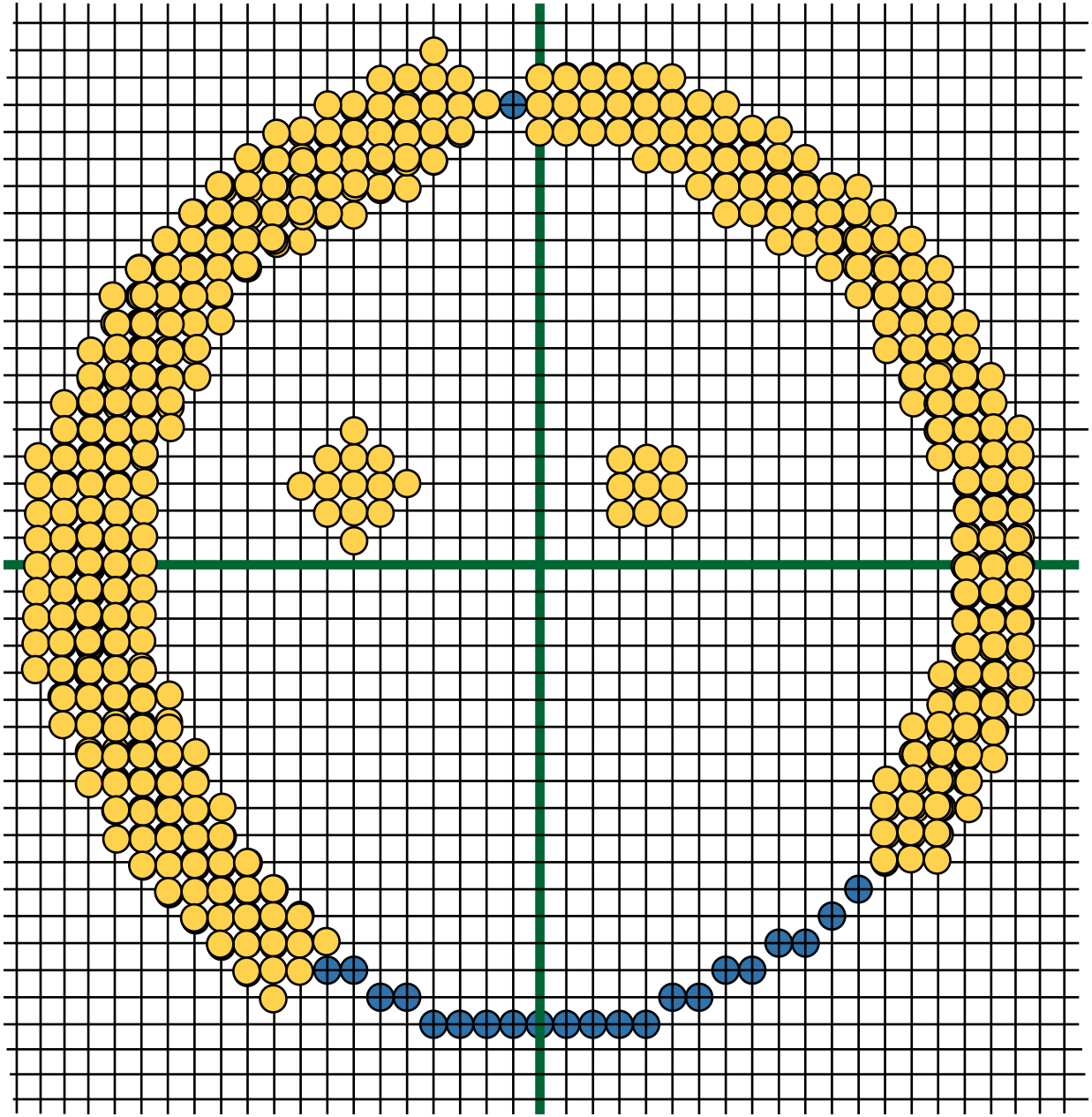


Probleme

- Linienenden sind immer horizontal oder vertikal gerade.
- Horizontale und vertikale Linien haben eine sichtbar unterschiedliche Breite im Vergleich zu schrägen Linien.
  - Sei t die Breite einer Linie (in Pixeln)
  - Horizontale und vertikale Linien haben die Breite t.
  - 45°-Linien haben eine Breite von $t/\sqrt{2}$.
- Geradzahlige Linienbreiten
  - Welches Pixel liegt genau auf der berechneten Linie?
  - Ober- und Unterkante (rechte/linke Kante) fallen aus

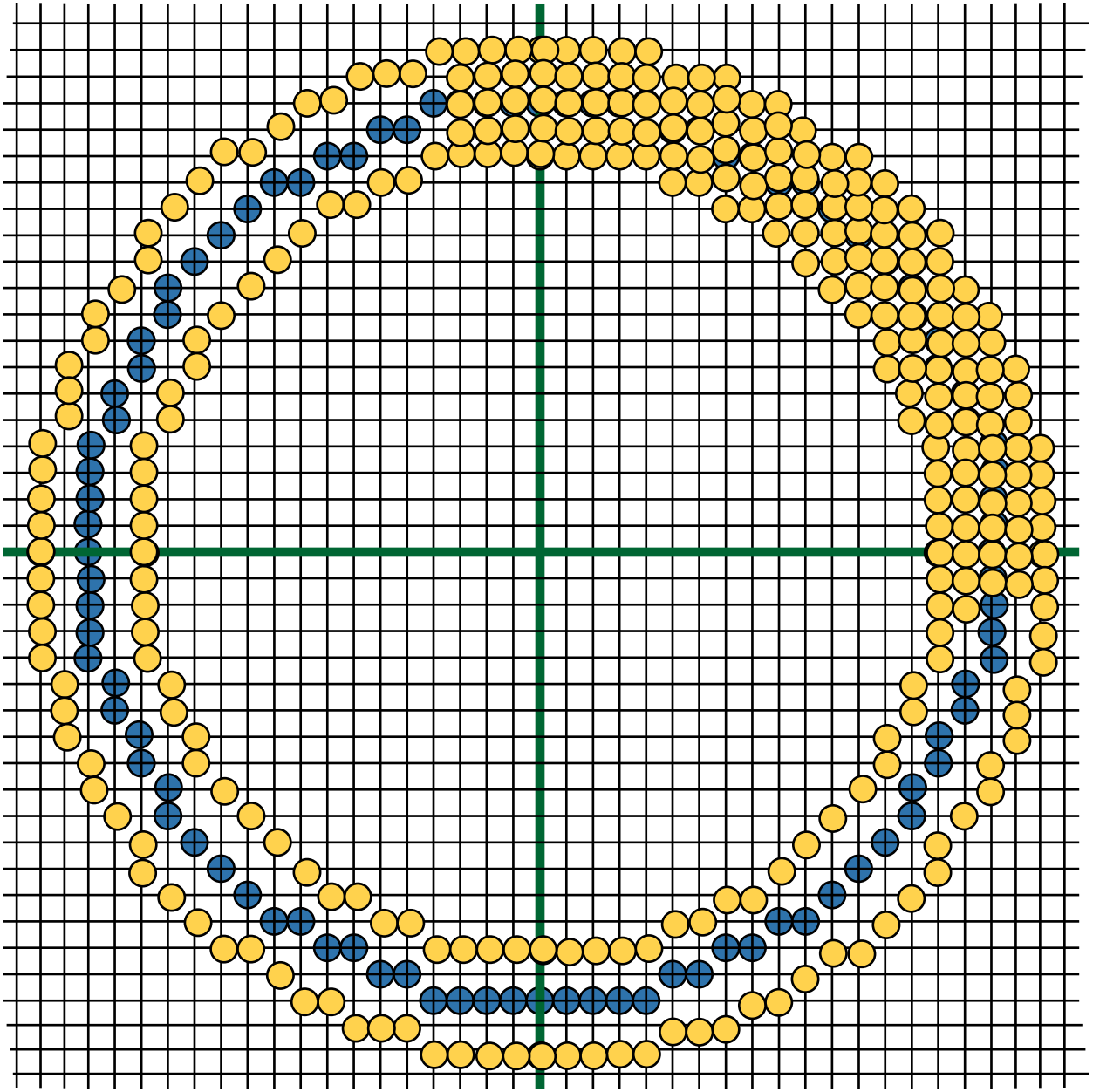
Bewegliche Pinsel

- Pinsel hinterlässt ein spezifisches Pixelmuster (Footprint)
- Bewegen des Pinsels entlang der gerasterten Linie und Zeichnen des Musters an jeder Stelle
- Probleme:
  - Linienenden nicht an der genauen Pixelposition
  - Unterschiedliche Breite der Linien bei unterschiedlichen Anstiegen
  - Viele Pixel werden mehrfach gesetzt
  - Aussehen der Linie stark abhängig vom Aussehen des Pixelmusters



Ränder zeichnen und Füllen

- Anstelle der Rasterung eines Primitivs werden zwei gerastert, mit Abstand $\pm t/2$ vom idealen Primitiv.
- Der Raum zwischen den beiden Primitiven wird dann gefüllt.
- Eigenschaften bzw. auftretende Probleme:
  - funktioniert auch mit geradzahligen Linienbreiten
  - Breite nicht abhängig vom Anstieg
  - Probleme beim Finden der „versetzten“ Linien
    - Geraden sind unproblematisch
    - Kreise sind unproblematisch
    - Ellipsen sind problematisch – Kurven, die $\pm t/2$ von einer Ellipse entfernt gezeichnet werden, sind keine Ellipsen
    - allgemeine Kurven funktionieren nicht



LFE Medieninformatik • Prof. Dr. Ing. Axel Hoppe

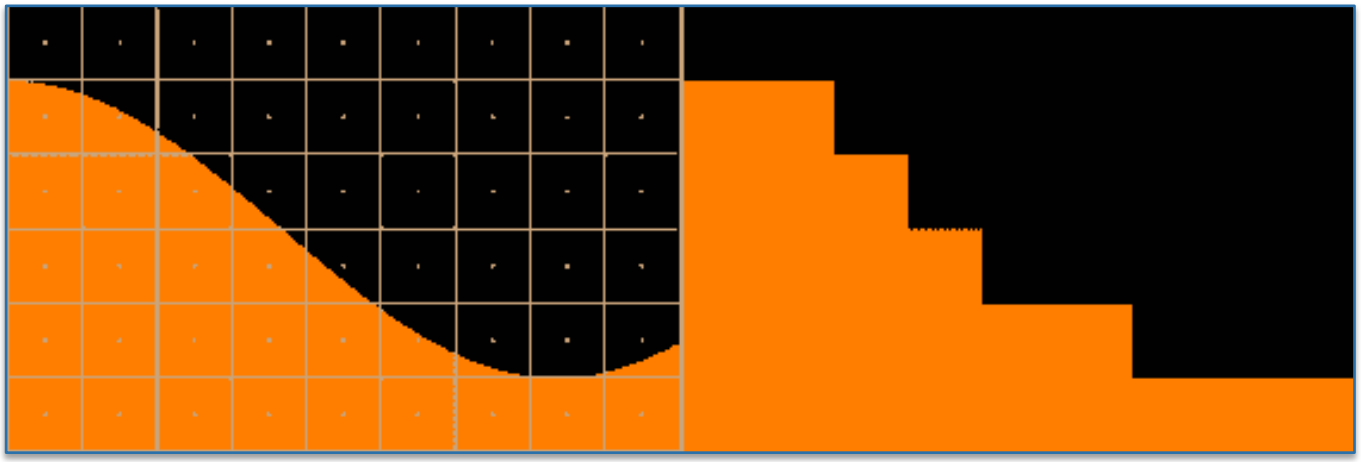
Rasteralgorithmen

Kantenglättung – Antialiasing

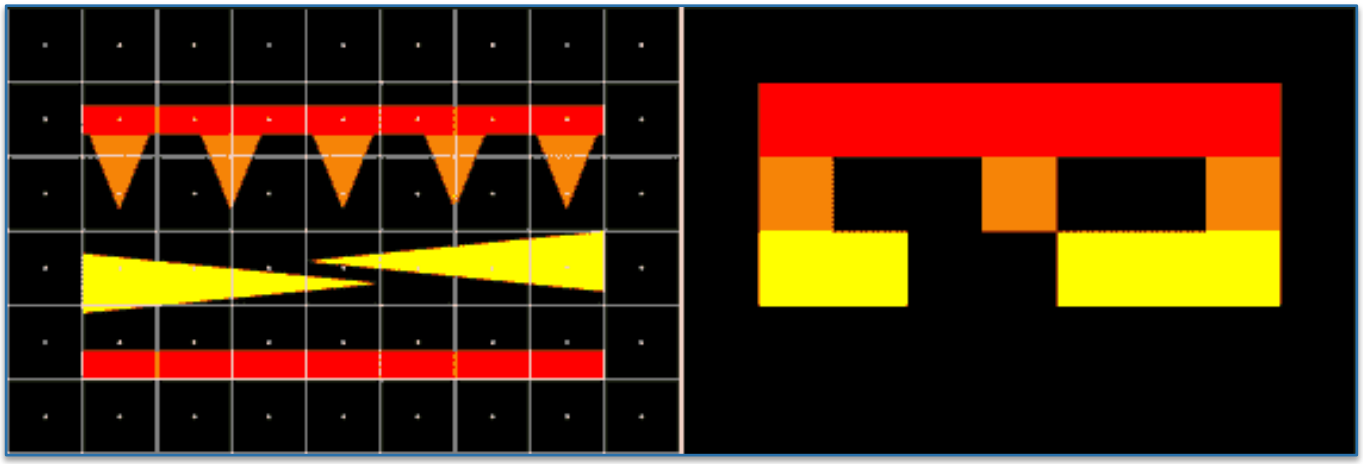
Antialiasing

- Neues Problem: Treppenstufen-Effekte beim Zeichnen von Linien, Details gehen verloren, etc.
- Aliasing
- Ursachen:
  - Diskretisierung eines kontinuierlichen Signals
  - unterschiedliche Abtastfrequenzen (Abtasttheorem)
- Lösung: **Antialiasing**
  - verschiedene Verfahren je nach Anwendung
  - hier insbesondere: Antialiasing beim Rastern von Linien
- erhöht die Qualität von Graphiken
- in Hardware verfügbar

Aliasing-Beispiele

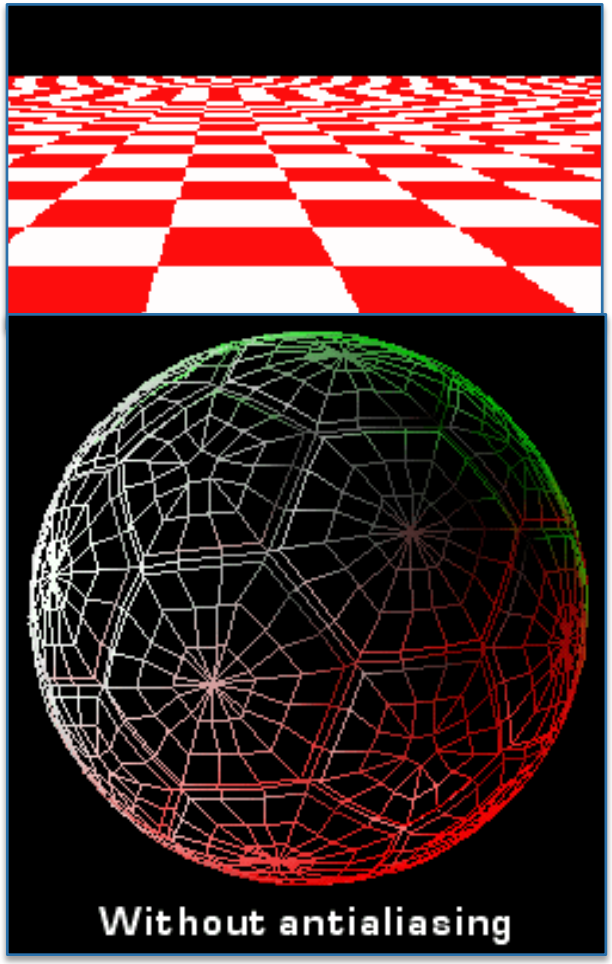


Treppenstufen

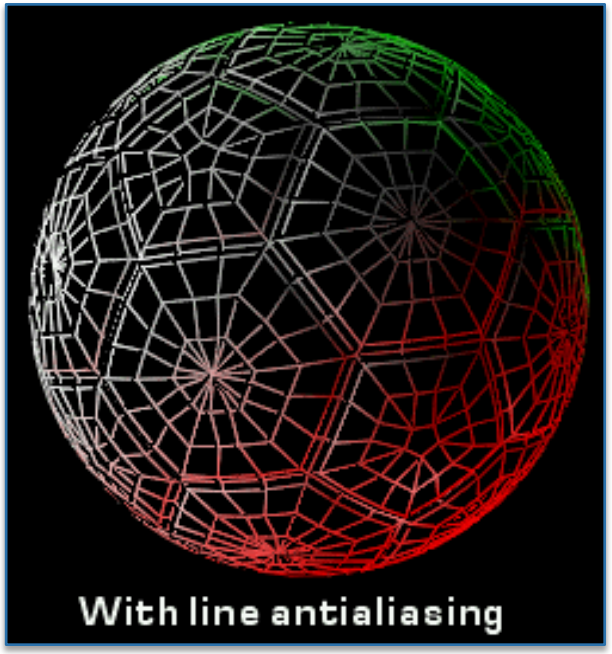


Binnen-Strukturen: Details gehen verloren

Aliasing-Beispiele



Moiré-Muster und Verwischen bei Texturen



Methoden

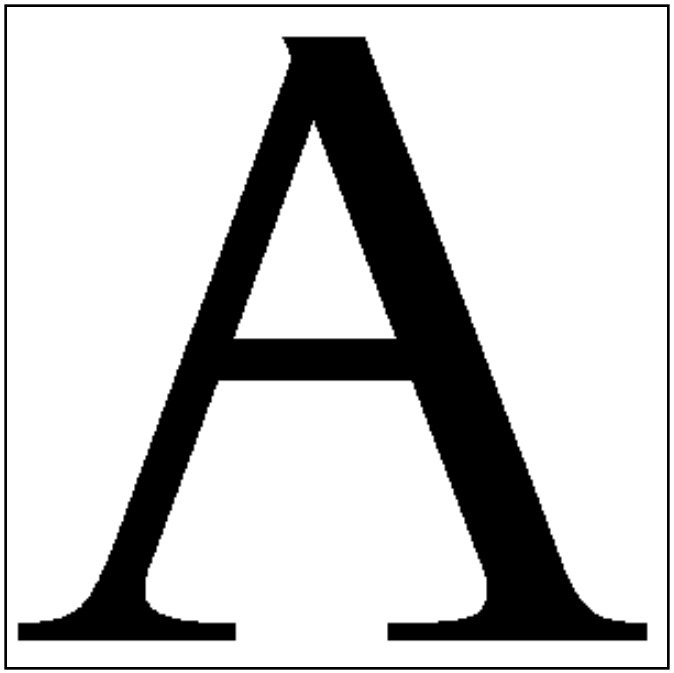
- Allgemeines Verfahren:
  - Erhöhen der Auflösung des Zielbildes und Zusammenfassen mehrerer Pixel zu einem
  - Supersampling
- Einbau in die Rasterungs-Algorithmen
  - Adaptive Anpassung der Pixelfarbe (Grauwert) an die Entfernung von der exakten Linie
  - Gewichtete Flächenbewertung
  - Ungewichtete Flächenbewertung

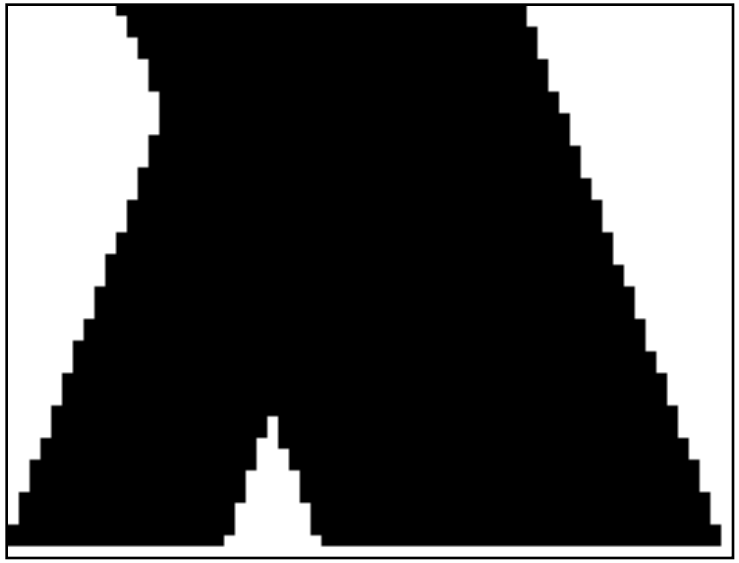
Supersampling

- einfachste Methode:
  - Bild wird in einer höheren Auflösung gesampled (berechnet)
  - danach: Downsampling auf die gewünschte Auflösung (gewichtetes Mittel der Sub-Pixel, die auf einen Pixel abgebildet werden)
- Praktikable Ansätze unterscheiden sich in:
  - Auswahl der „Super-Auflösung“
  - Lage der Subpixel zum Pixel an sich (Gitter)
  - Auswahl des Downsampling-Filters
    - einfaches Averaging
    - Box-Filter
    - Convolution

Einfaches Supersampling

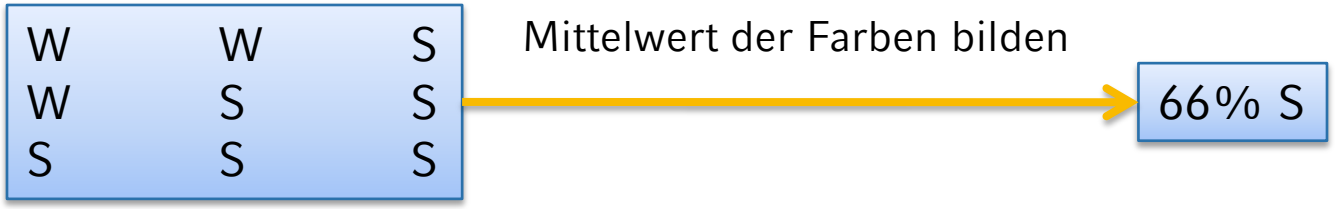
- wähle eine höhere Auflösung für ein „virtuelles Bild“
  - z. B. dreifache Auflösung
  - stelle das Bild mit dieser höheren Auflösung dar





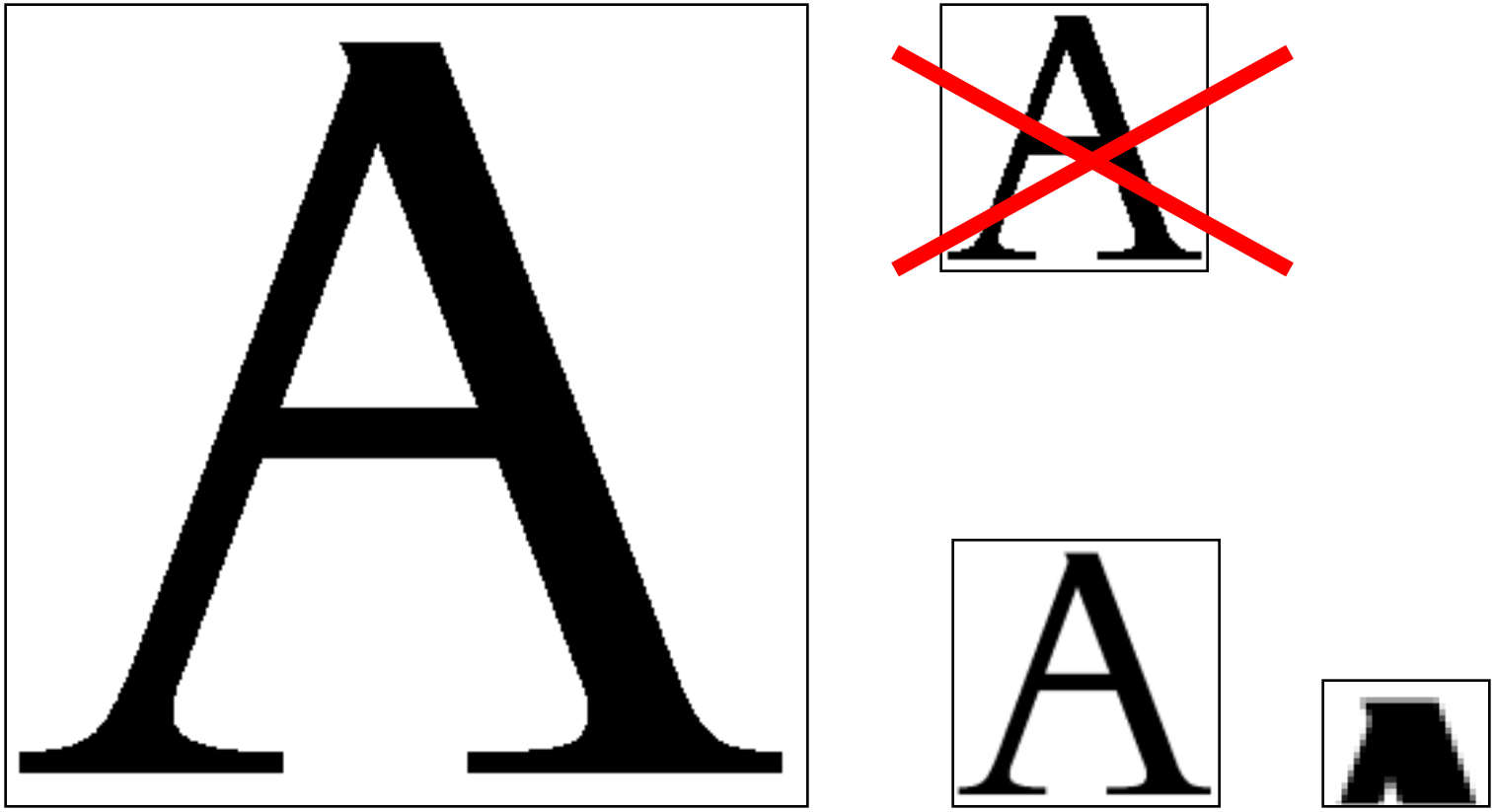
Einfaches Supersampling

- fasse jeweils *n* Pixel des „virtuellen Bildes“ zu einem Pixel des Ergebnisbildes zusammen (im Beispiel jeweils 3 × 3 Pixel)



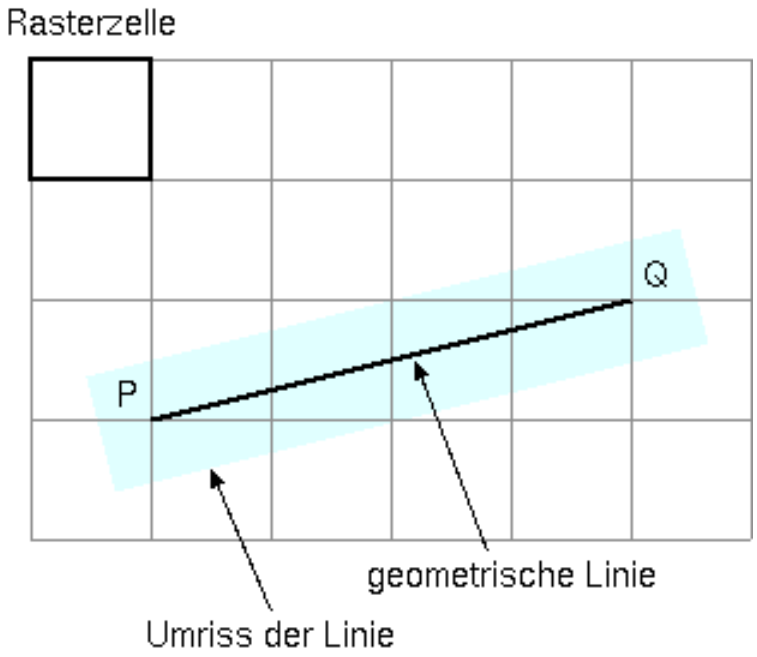
Einfaches Supersampling

- Ausgabe des Ergebnisbildes



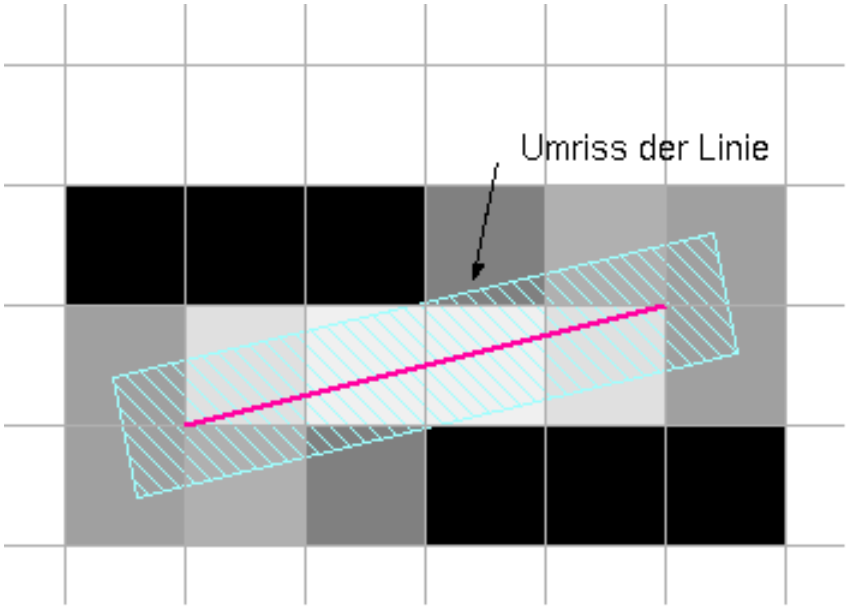
Flächenbewertungs-Algorithmen

- Anwendung direkt bei der Rasterkonvertierung von Primitiven (Linien, etc.)
- Geometrische Linie hat keine Breite, a*ber*: eine gerasterte Linie ist (wenigstens) 1 Pixel breit
- Geometrischer Umriss einer Linie = langes, schmales Rechteck
- Rechteckfläche überlappt Pixelgrundflächen teilweise, ungleichmäßig
- Bisherige Vereinfachung: Pixel gesetzt oder nicht zu grob → führt zu Treppenstufen-Effekten



Ungewichtete Flächenbewertung

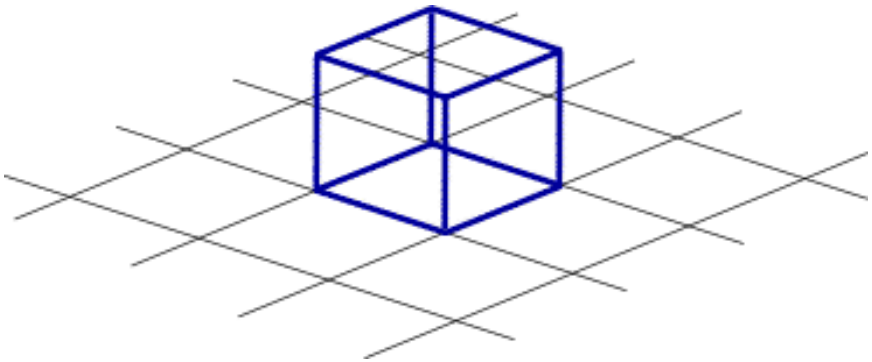
- Idee: Pixelhelligkeit proportional zum Anteil der Überdeckung dem Linienrechteck dosieren
- Annahme: Jedes Pixel hat quadratische Grundfläche der Größe 1 × 1 = Rasterzelle.



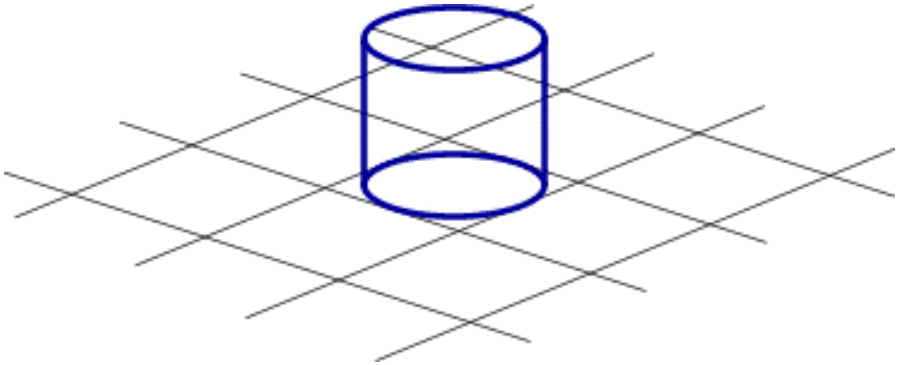
Grad der Überdeckung
Rasterzelle/Linienumriss
= Pixel-Helligkeit



Gewichtete Flächenbewertung

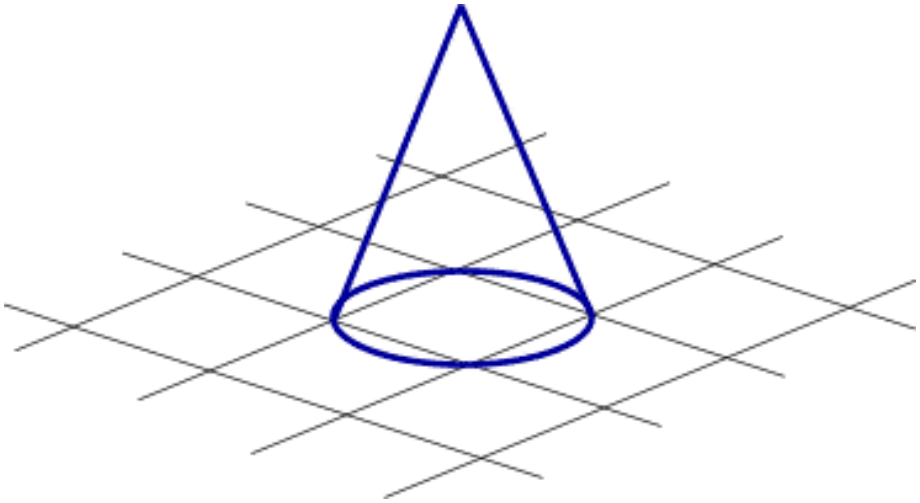


Helligkeitsbestimmung bei ungewichteter Flächenbewertung: Leuchtvolumen als Würfel, Anteil der überdeckten Linienfläche

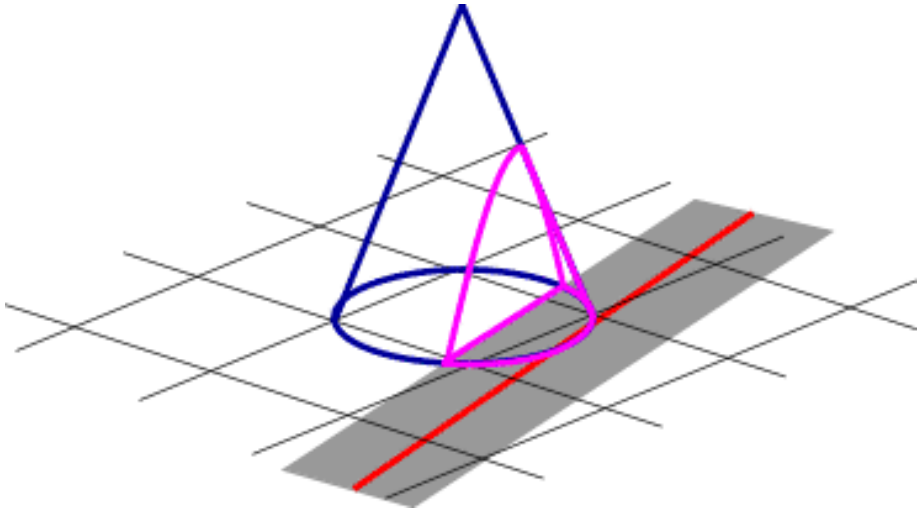


Realität: Pixelgrundflächen sind keine Quadrate sondern Kreise, daher Leuchtvolumen eher als Zylinder ansehen

Gewichtete Flächenbewertung



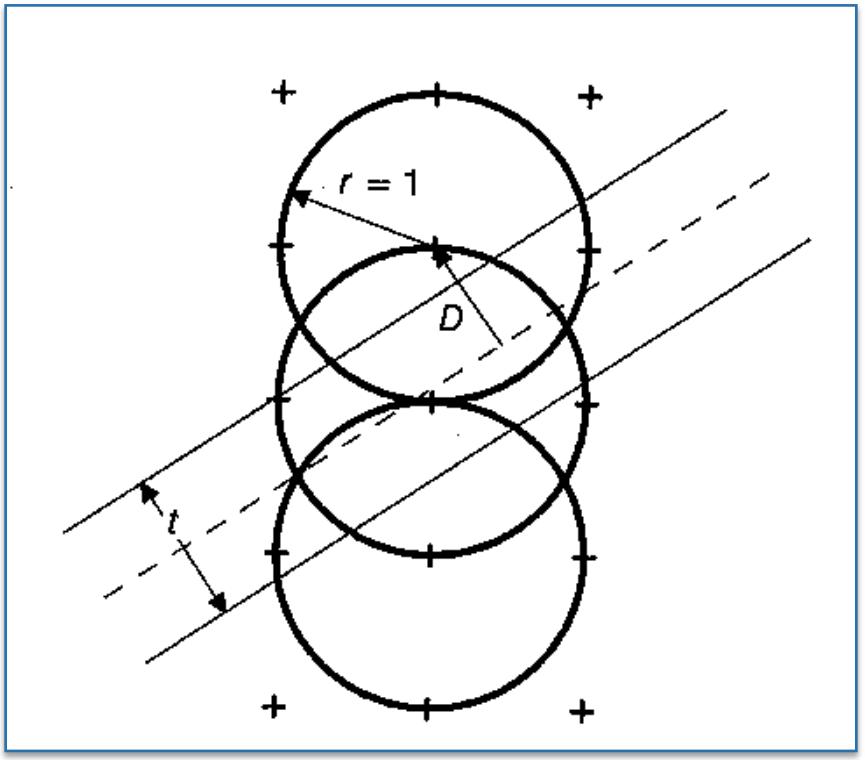
Pixelflächen auch nicht gleichmäßig hell ausgeleuchtet, heller im Zentrum, Licht fällt auch in Nachbarpixelzellen, daher Leuchtvolumen als Kegel mit größerer Grundfläche



Gewichtete Flächenbewertung: Anteil der
Pixelhelligkeit entspricht abgeschnittenem Volumenanteil des Kegels

Gupta-Sproull-Linien

- Verbindung von gewichteter Flächenbewertung mit dem Bresenham-Algorithmus für Linien
- Geometrisches Modell:
  - Mittelpunkte der Pixel auf Gitterpunkten mit Abstand 1
  - Pixel besitzen Einfluss-Bereich, der einer Kreis-scheibe mit Radius 1 entspricht
  - Linie besitzt Dicke t
  - Abstand Pixel zu Gerade wird durch den Abstand Pixelmittelpunkt zur Mittellinie des Primitivs bestimmt



Gupta-Sproull-Linien

- Ablauf des Verfahrens
  1. Berechne Rasterpunkt (Bresenham)
  2. Berechne Distanz D zwischen Rasterpunkt und Linie
  3. Gehe einen Schritt entlang der y-Achse aufwärts und berechne Du
  4. Gehe einen Schritt entlang der y-Achse abwärts und berechne Dl
  5. mit größerem D, Dl, Du ergibt sich kleinerer Grauwert
  6. Jede Linie ist somit 3 „Pixel dick“
- Grauwerte können vorberechnet werden
  1. max. Abstand ist 1.5, unterteilen je nach Anzahl der Graustufen (16, 256, etc.)
  2. Abspeichern in Tabelle und Lookup über Abstand

Zusammenfassung Antialiasing

Vorteile	Nachteile
• Eckige Kanten und harte Übergänge werden reduziert • Pixel- und Linien-Flimmern wird reduziert	• Verringerte Schärfe • Kleine Schriften werden „beschädigt“ • Größere Datenmengen • Längere Rechenzeiten

Quellen

- Foley, van Dam, Feiner, Hughes: *Computer Graphics, Principles and Practice*. Zweite Auflage, Addison Wesley. ISBN 0-201-84840-6.
- Bernhard Preim: *Computergraphik 1*. Universität Magdeburg, Vorlesungsskript, Juli 2005.

Computergrafik 1
2. Teil: Bildverarbeitung

Digitalfotografie, Abtastung von Bildern,
Konvolution und Korrelation,
Fouriertransformation

Themen heute

- Bilder aus der Digitalfotografie
  - Kontrastumfang, Histogramme
  - Kurzeinführung Zonensystem
  - High Dynamic Range (HDR) Bilder
- Abtastung von Bildern
  - Sampling
  - Aliasing
  - Moire Muster
- Konvolution und Korrelation
- Fourier-Transformation (Anfang)

Bilder aus der Digitalfotografie

Kontrastumfang einer Szene

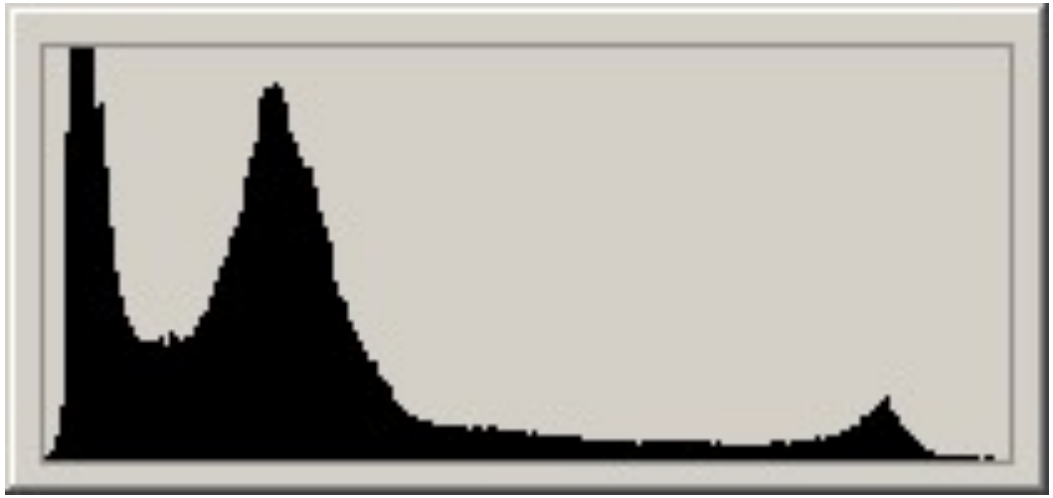


Kontrastumfänge versch. Medien

- Auge:
  - 2^{10} ohne Irisanpassung
  - 2^{20} mit Irisanpassung
- Schwarzweißfilm: ca. 2^8
  - Push-Entwicklung: 2^7
  - Pull-Entwicklung: 2^9
- Moderner Farbnegativfilm: 2^{10}
- Diafilm: 2^5
- Digitalkamera:
  - EOS 10D bei 400 ASA: 2^9

Histogramm eines Bildes

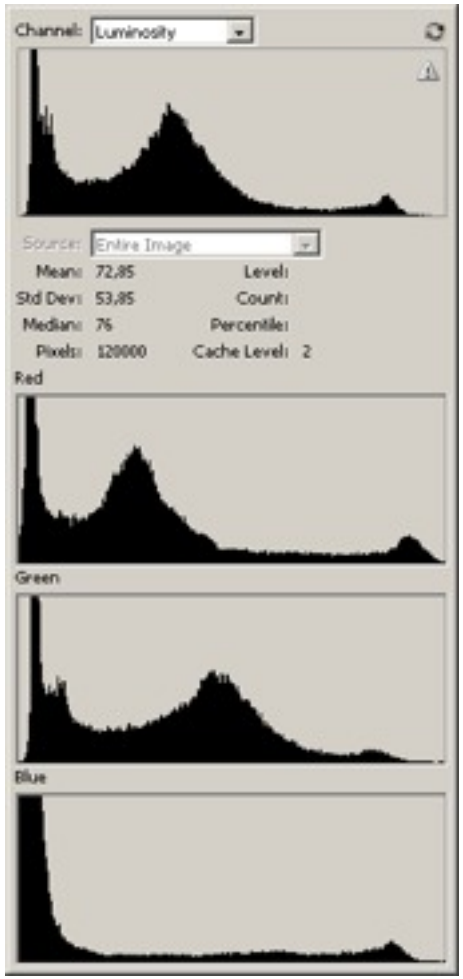




- Relative Anzahl der Pixel im Bild je Helligkeit
- Wichtiges Instrument zur Beurteilung der Datenqualität

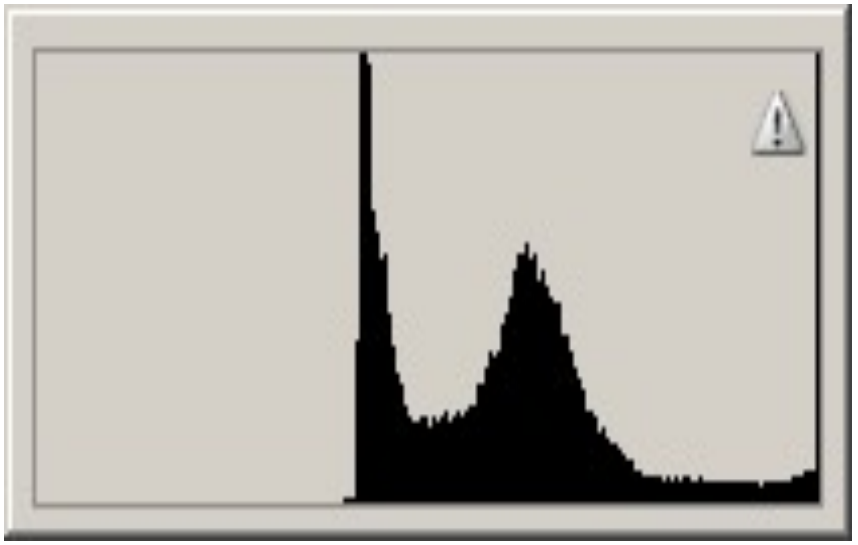
Histogramme der Farbkanäle



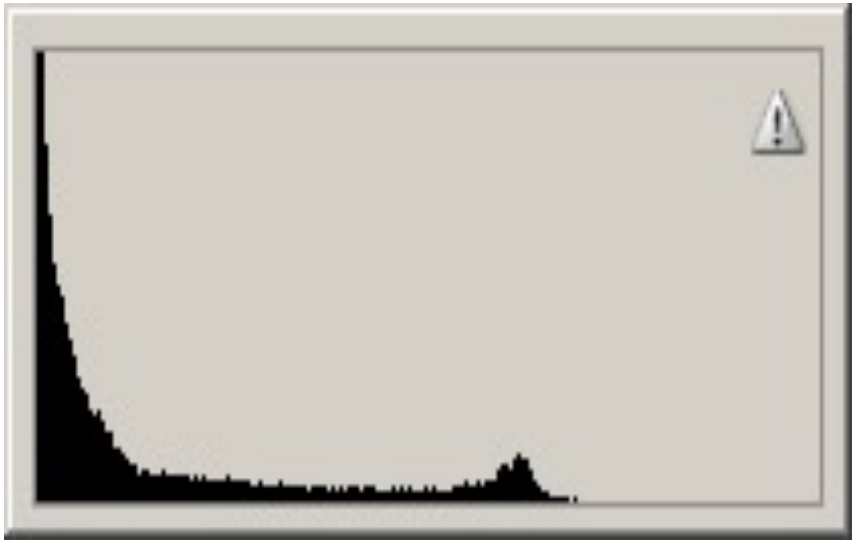


Über- und Unterbelichtung



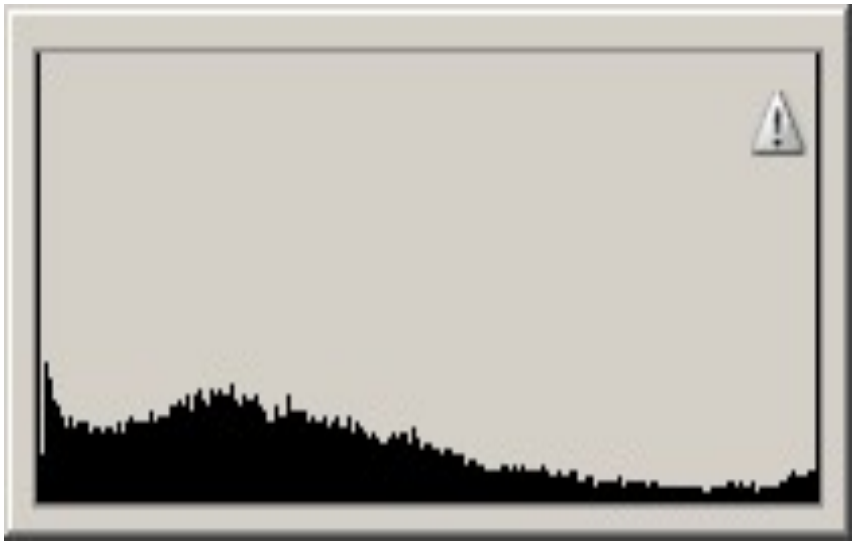




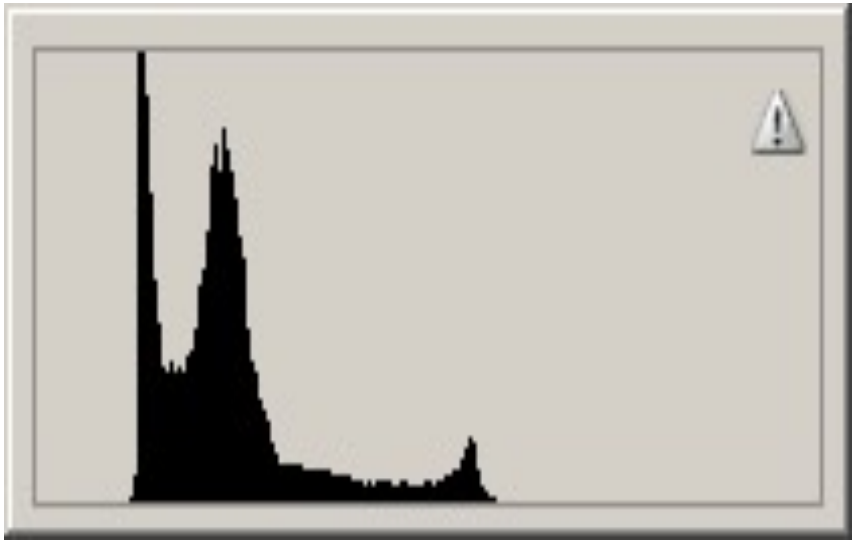


Kontrastanhebung und -absenkung



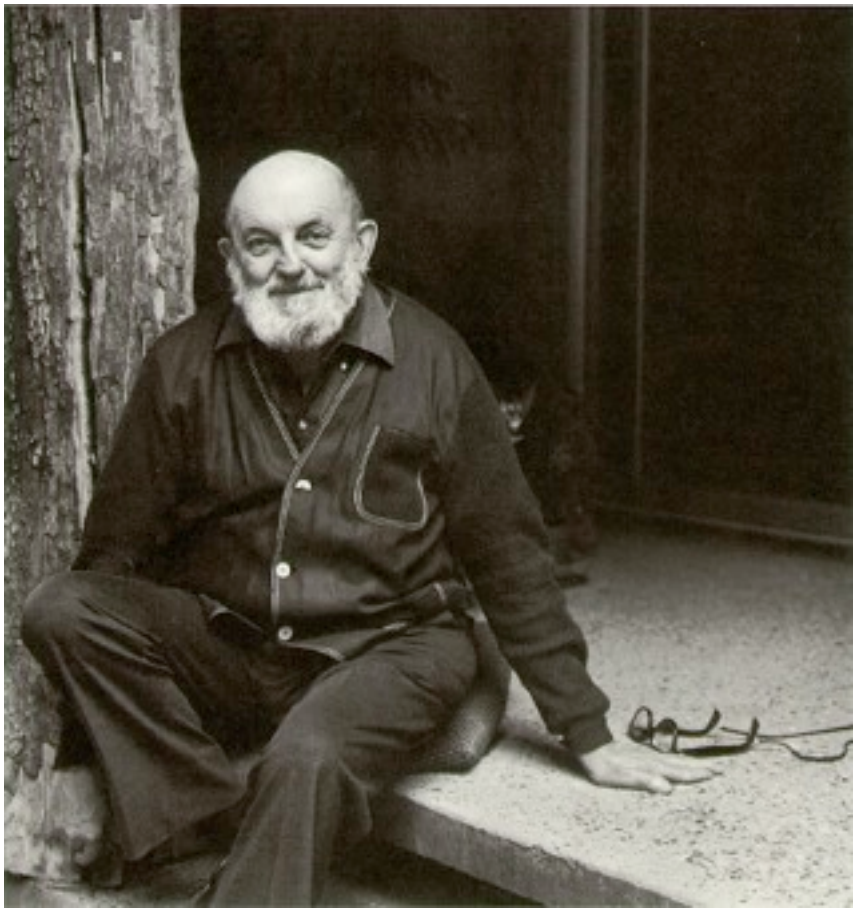




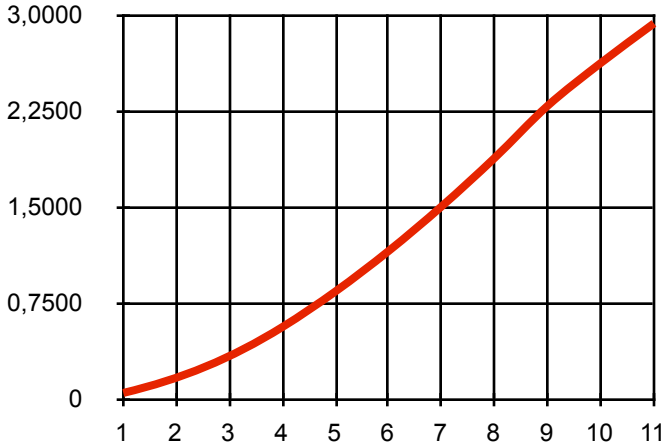


Exkurs: Zonensystem (Ansel Adams)

- Ansel Adams [1902-1984]
- Landschaftsfotografie in S/W in Perfektion
- Verwendeter Prozess:
  - S/W Negativfilm (Großformat)
  - Abzug auf S/W Papier
  - Komplette Steuerung der Entwicklungsprozesse für Negativ und Positiv
  - Abwedeln und Nachbelichten zur Steuerung lokaler Bildhelligkeit
- Definition von 10 „Zonen“ für Helligkeiten im Bild



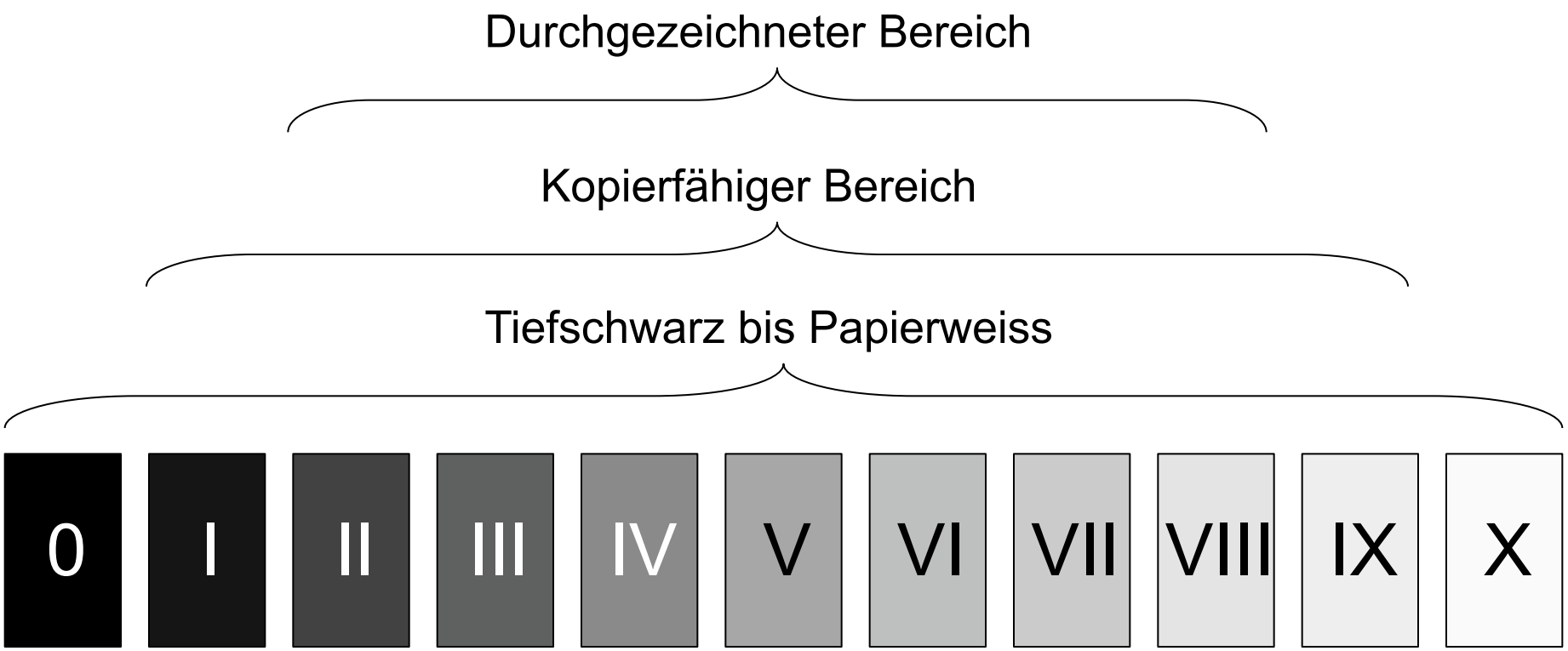
Schwärzungskurve von Filmen



- X-Achse: Logarithmus der Belichtung (entspr. Blenden- oder Zeitstufen)
- Y-Achse: Dichte des geschwärzten Films, ebenfalls logarithmische Einheit
- Kurve best. aus Fuß, linearem Bereich, Schulter

Definition der Zonen

- Jede Zone entspricht einer Belichtungsstufe, also der Verdopplung der Lichtmenge
  - Durch Wahl der nächstkleineren Blendenzahl
  - Oder durch Verdopplung der Belichtungszeit



Definition der Zonen

- II: tiefe Schatten fast ohne Details
- III: voll durchgezeichnete Schatten
- IV: dunkles Laubwerk
- V: Neutralgrau mit 18% Reflexion
- VI: mittlere Hauttöne
- VII: helle Hauttöne, Schneeflächen
- VIII: noch gezeichnete Lichter
- IX: fast Papierweiss ohne Details





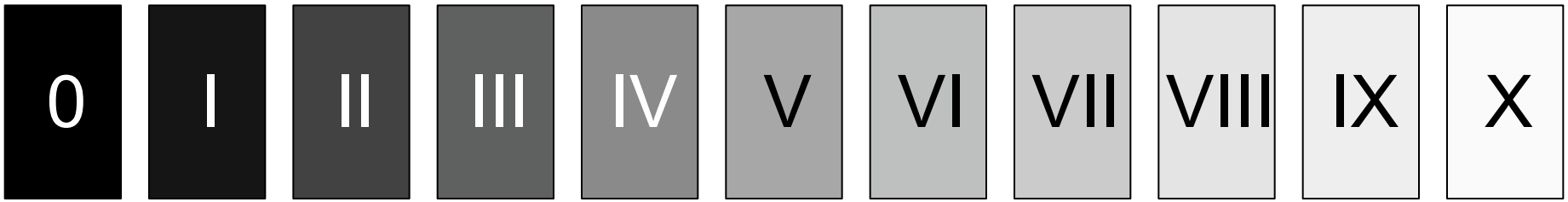
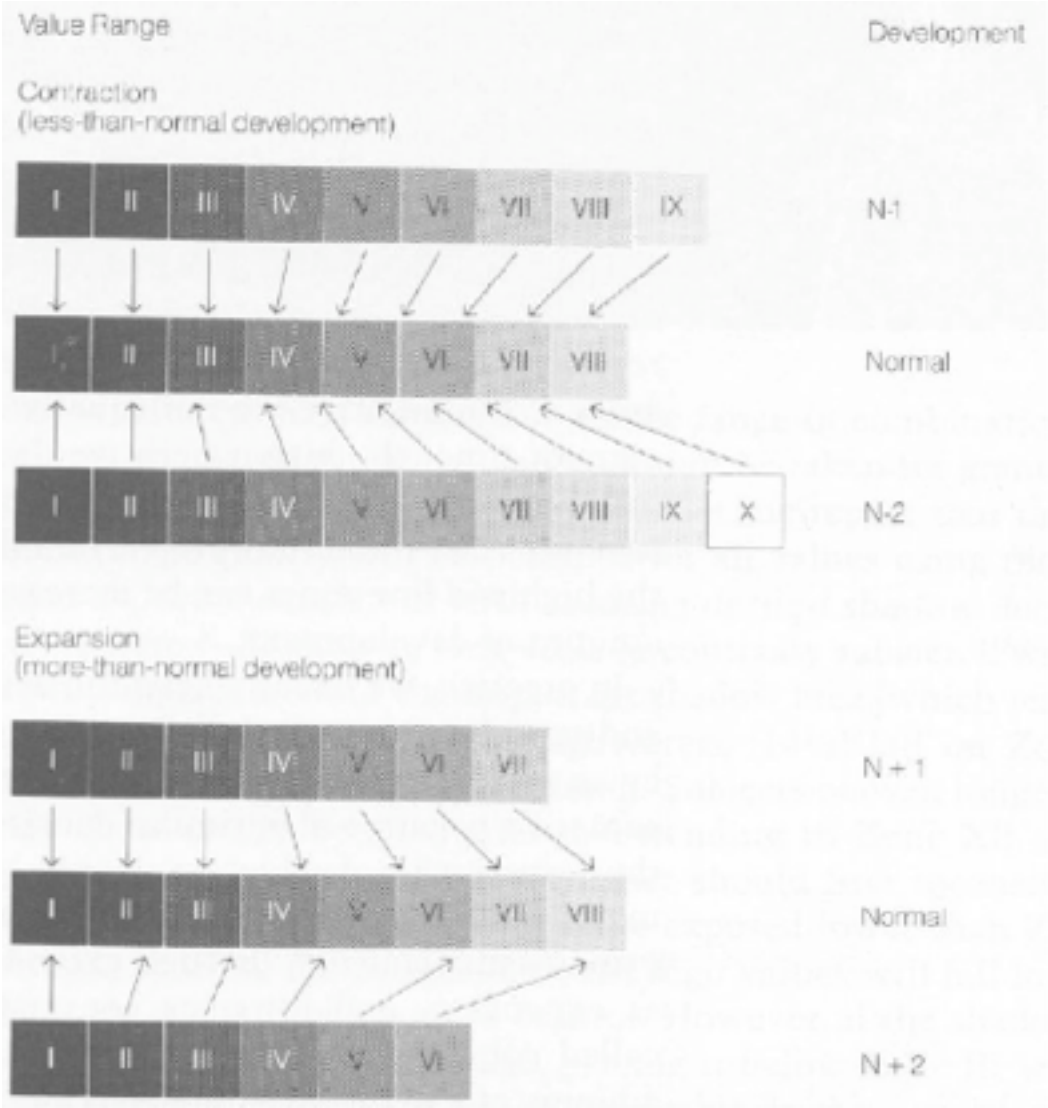
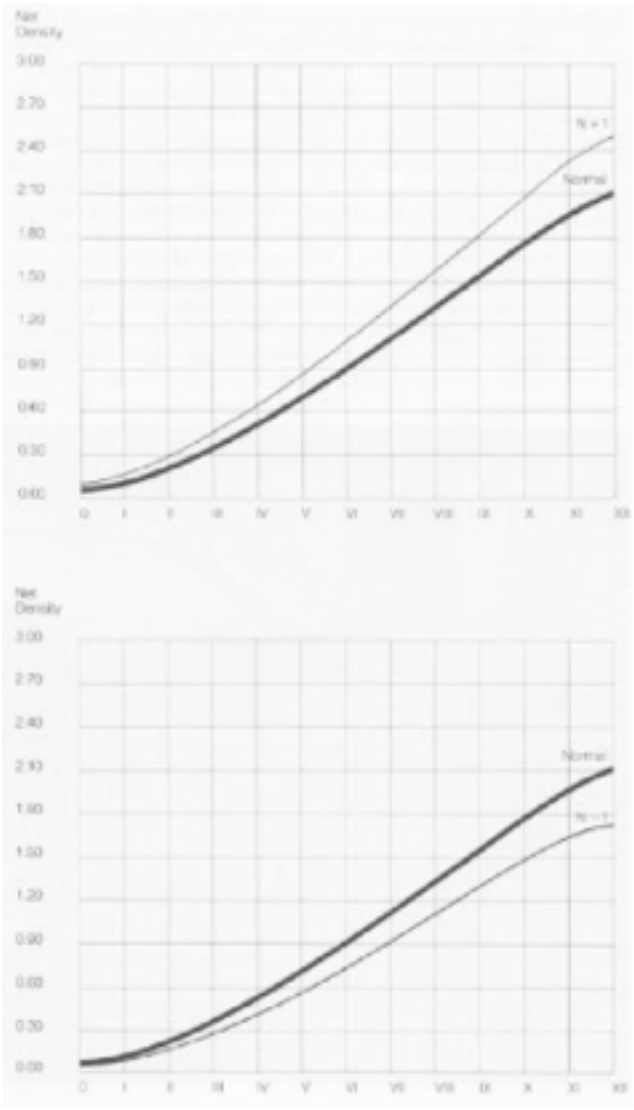


Abbildung der Szene in Zonen

- Bei „normalem“ Kontrastumfang:
  - Schatten in Zone III
  - Lichter <= Zone VIII ?
- Bei zu hohem Kontrastumfang der Szene:
  - Bewusster Verzicht auf Details in Lichtern oder Schatten
  - Verkürzte Filmentwicklung und verlängerte Belichtung
- Bei zu niedrigem Kontrastumfang:
  - Abzug auf „hartes“ Papier (d.h. mit hohem Kontrast)
  - Verlängerte Filmentwicklung und verkürzte Belichtung
- Ziel: volle Ausnutzung des Kontrastumfangs des Films, aber überall noch volle Detailzeichnung

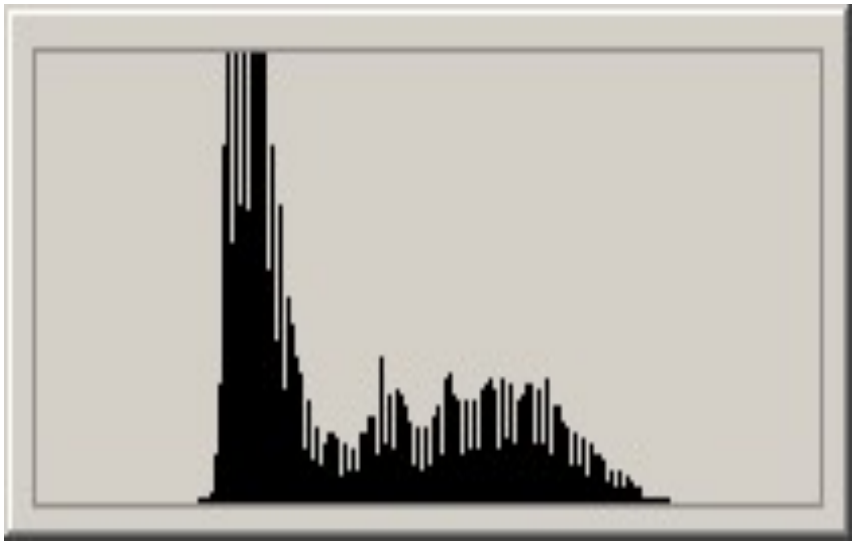
Abbildung der Szene in Zonen



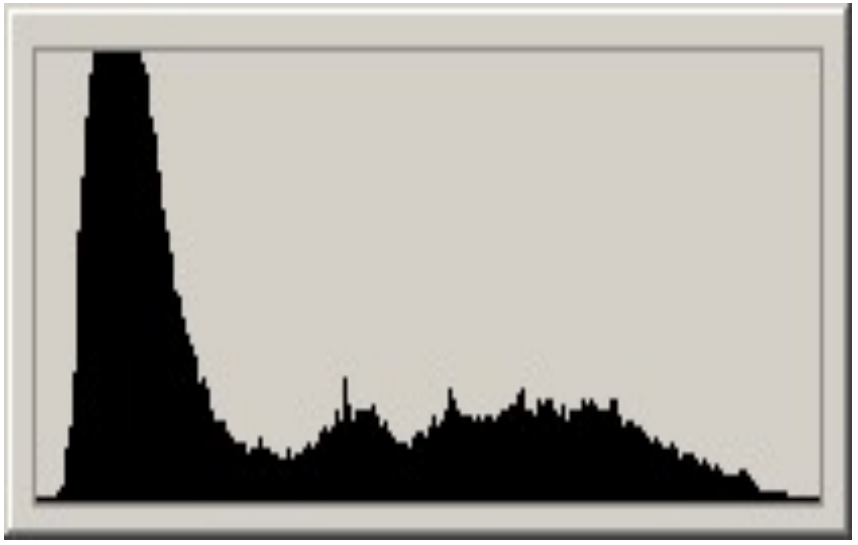


Auswirkung am Histogramm









Übertragung auf digitale Fotografie

- Bewusste Steuerung der Helligkeit durch Spotmessung auf mittleres Grau
- Digitale Bildsensoren kritisch in den Lichtern
  - Warnfunktion mancher Kameras benutzen
  - Im Zweifel eher knapper belichten
  - Wenn Zeit genug: Belichtungsreihe
  - Aufhellblitz zur Kontrastreduktion
- Reduktion vom Bildsensor (12 Bit) auf JPEG (8 Bit) findet schon in der Kamera statt
  - RAW Format verwenden, um Details in Lichtern und Schatten zu erhalten
- Kontrastanpassung später am Histogramm

High Dynamic Range (HDR) Bilder

- Ziel: Kontrastumfang der Natur adäquat darstellen (z.B. 2^{20})
- 1. Problem: beschränkter Dynamikumfang der Aufnahmesensoren
- 2. Problem: Bildformate haben nur 8 oder 16 Bit Dynamikumfang je Farbkanal



Bildquelle: http://www.cybergrain.com/
(Auch folgende Folien)

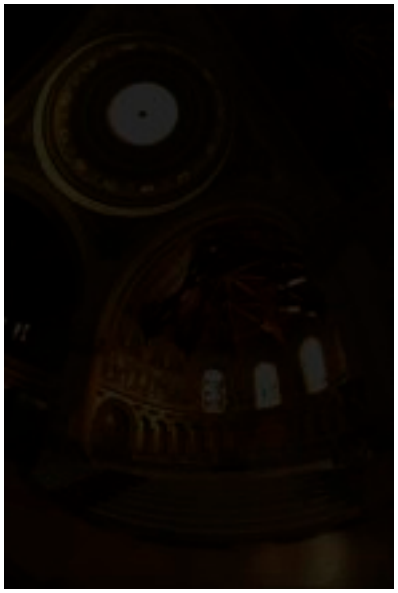
Repräsentation von HDR im Rechner

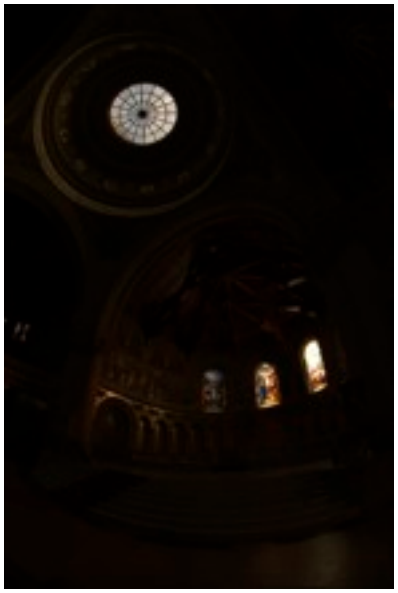
- Einfacher Trick: Pixelwerte nicht als 8 oder 16 Bit Integers, sondern
  - 32 oder 48 Bit Integer
  - 16 Bit Floats
  - 32 Bit Floats
- Verschiedene Formate vorgeschlagen
  - http://www.openexr.com/
  - Portable Float maps (PFM, wie PPM)
  - floating point TIFF

Erzeugung von HDR Bildern



- Spezielle HDR Kameras
- Raytracing bzw. Rendering
- Belichtungsreihe und Kombination mittels Photoshop oder HDRshop







Anzeige von HDR Bildern

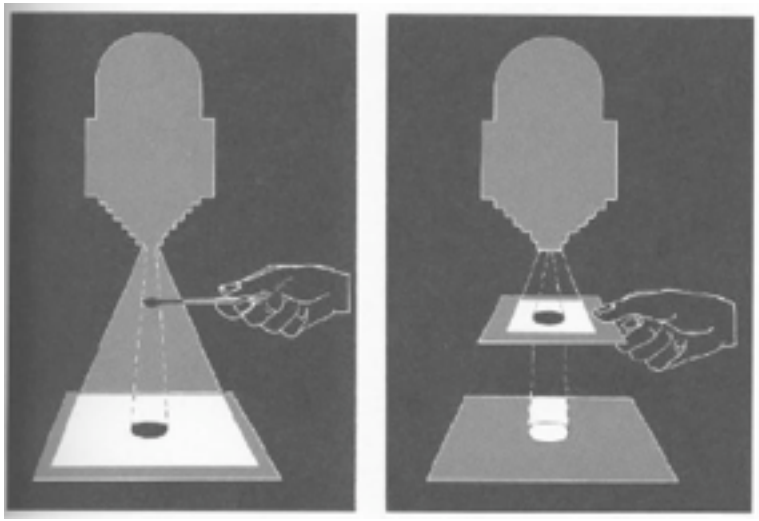
- Bis vor kurzem garnicht ;-)
- Heute HDR-Displays, Kontrastumfang bis 1:200.000 (2^{18}) (Quelle http://www.brightsidetech.com/)
- Grundidee: Hintergrundbeleuchtung aus einzelnen weißen LEDs
- Kann je Pixelgruppe

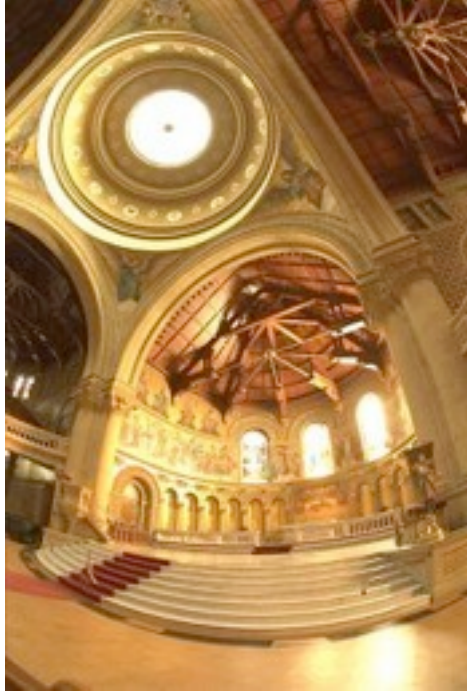
Hell oder Dunkel sein

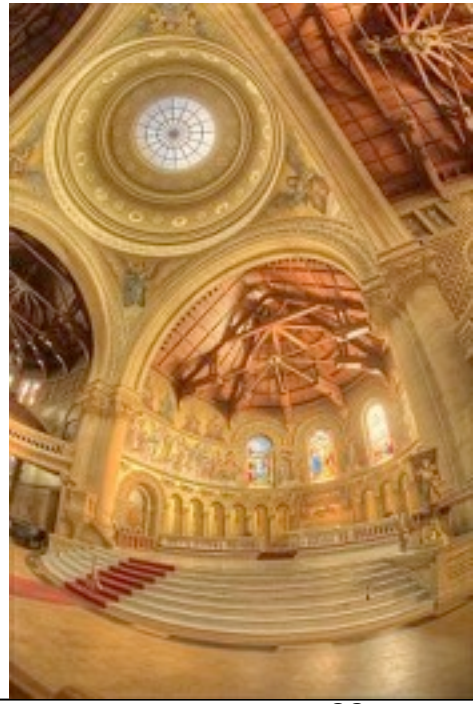


Konvertierung auf LDR

- Verfahren: Tone mapping
- z.B. manuell in Photoshop mittels „dodge and burn“
- Automatisch: aktuelles Forschungsgebiet







Tone mapping mit lokalen Mittelwerten

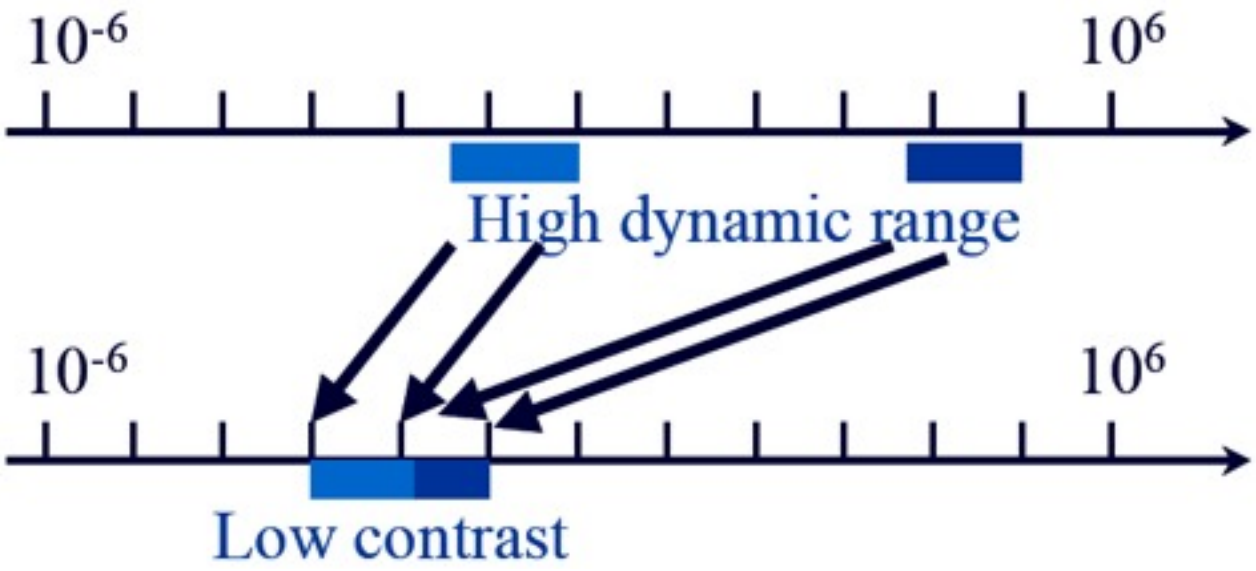


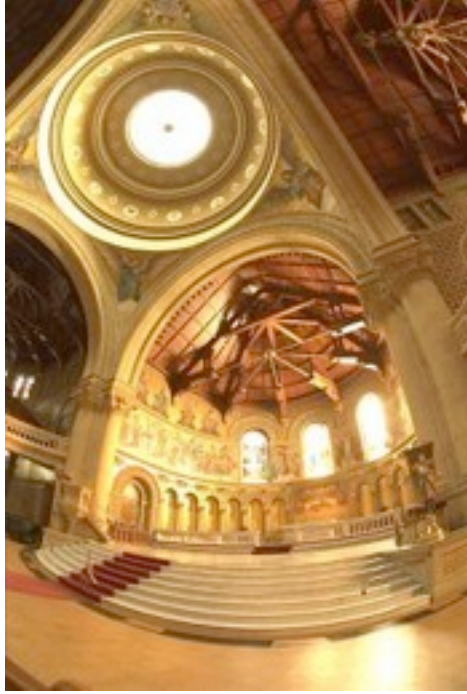
Tone mapping mit lokalen Mittelwerten

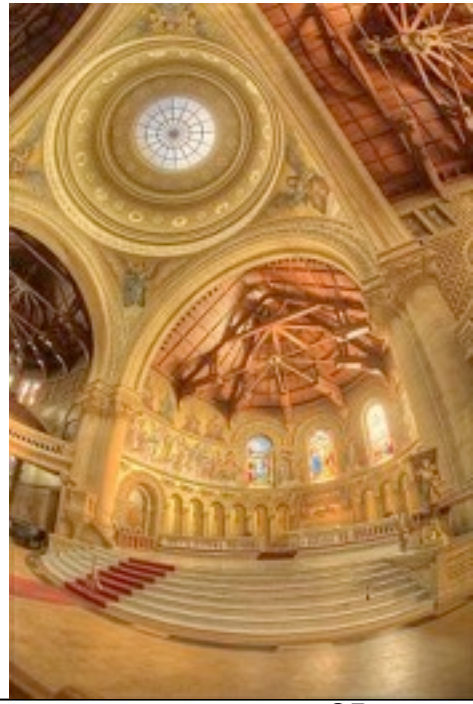


Tone mapping am Histogramm

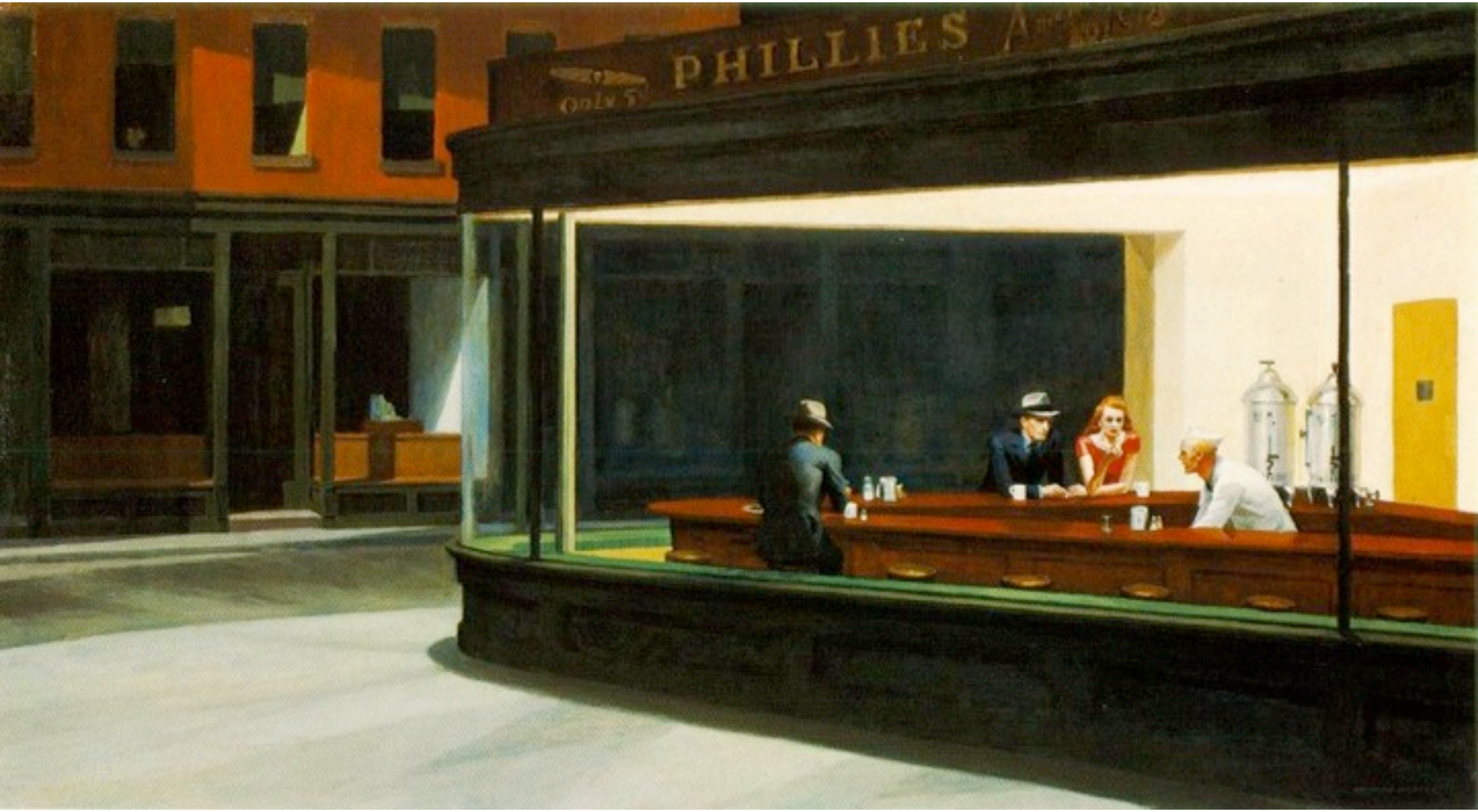
- Kernidee: ignoriere leere Stellen im Histogramm
- Schiebe restliche Bildteile im Histogramm zusammen










PHILLIES

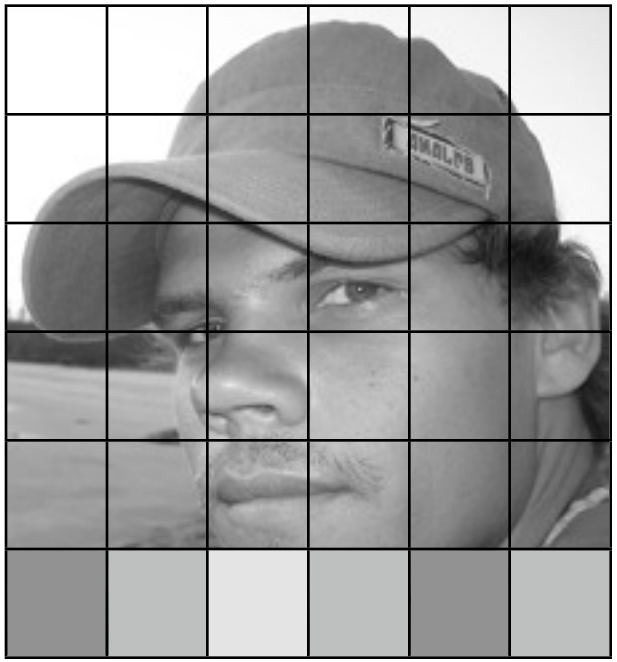
Linksammlung

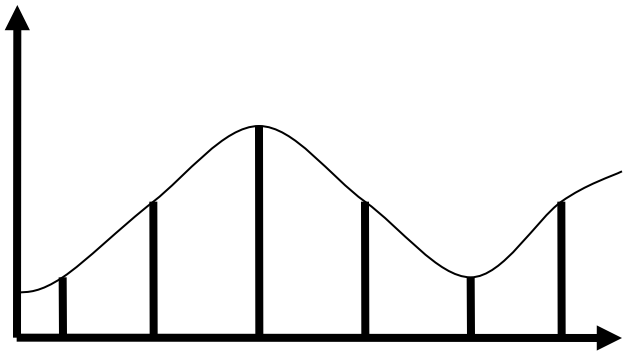
- http://www.anseladams.com/
- http://www.cybergrain.com/tech/hdr/resources.html
- http://www.hdrshop.com/

Abtastung Digitaler Bilder

Abtastung einer Szene

- Bildhelligkeit wird nicht für beliebige x,y kontinuierlich gemessen
- Messung nur an bestimmten Stellen, bzw. Integration über bestimmte Bereiche (Pixel)
- Entspricht mathematisch der Multiplikation mit einer Impulsfolge (Summe verschobener Impulsfunktionen)
- Impulsfunktion δ (oder Dirac-Funktion):



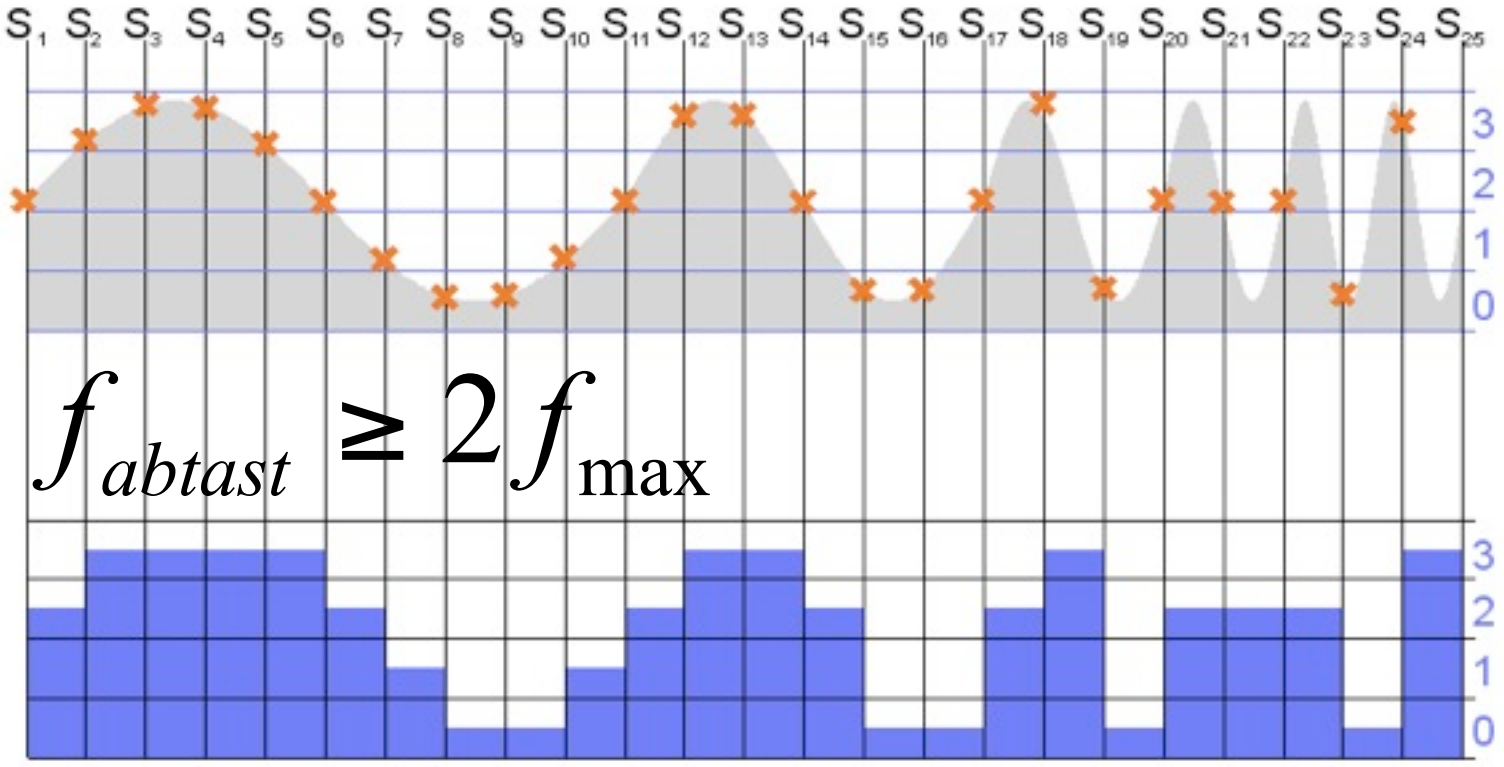


$$\delta(x-a) = \begin{cases} 0 & x \neq a \\ \infty & x = a \end{cases} \qquad \int_{-\infty}^{\infty} f(x)\delta(x-a)\,dx = f(a)$$

http://de.wikipedia.org/wiki/Delta-Distribution

Abtasttheorem von Nyquist-Shannon

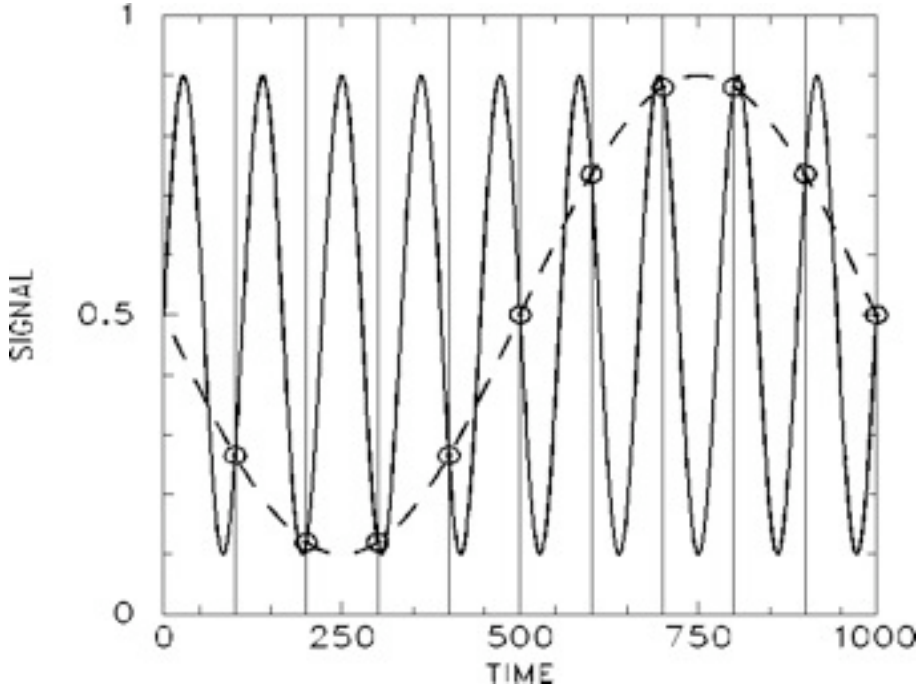
- Ein digitales Signal mit maximalem Frequenzanteil f_{max} muss mit mehr als der doppelten Frequenz $2\,f_{max}$ abgetastet werden



Aliasing

$$f_{abtast} < 2 f_{signal}$$

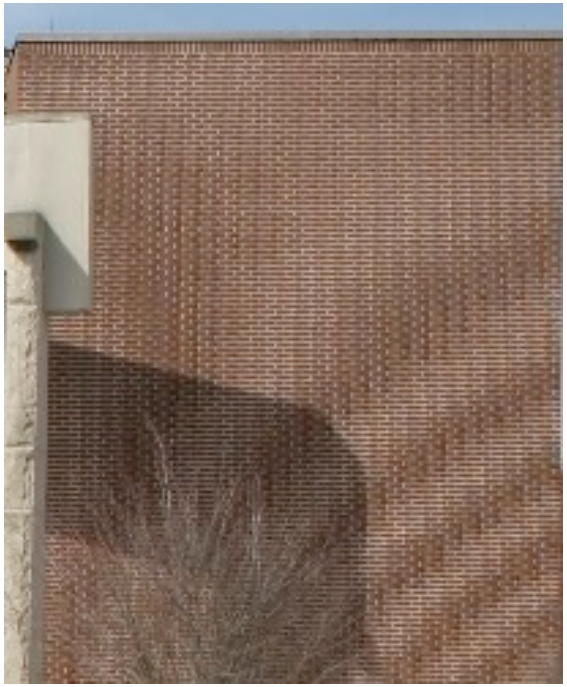
$$f_{alias} = \left| f_{abtast} - f_{signal} \right|$$



- Durch Unterabtastung ($f_{abtast} < 2\ f_{max}$) entstehendes falsches Signal
- Frequenz des Aliasing-Signals = Differenz aus Original- und Abtastsignal
- Schwebungstöne beim Stimmen einer Gitarre

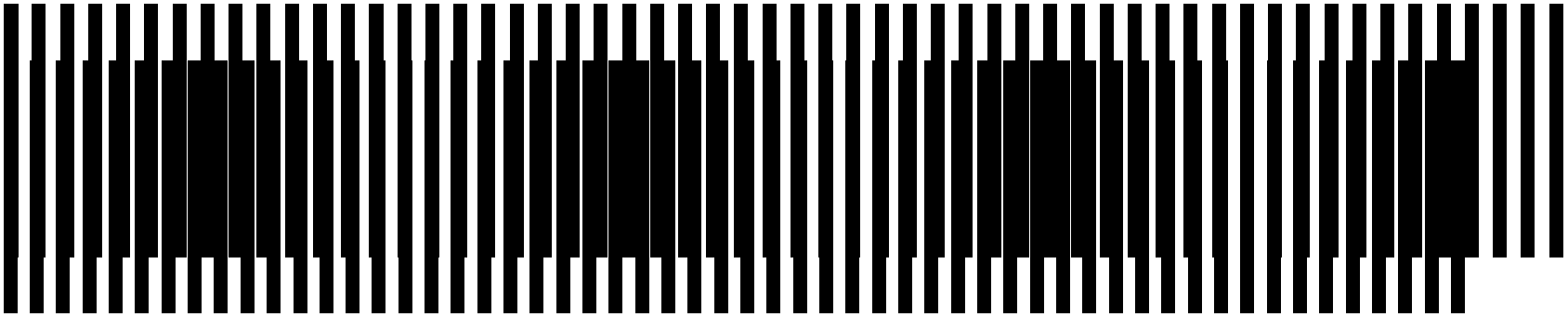
Aliasing in digitalen Bildern

Film-Beispiel

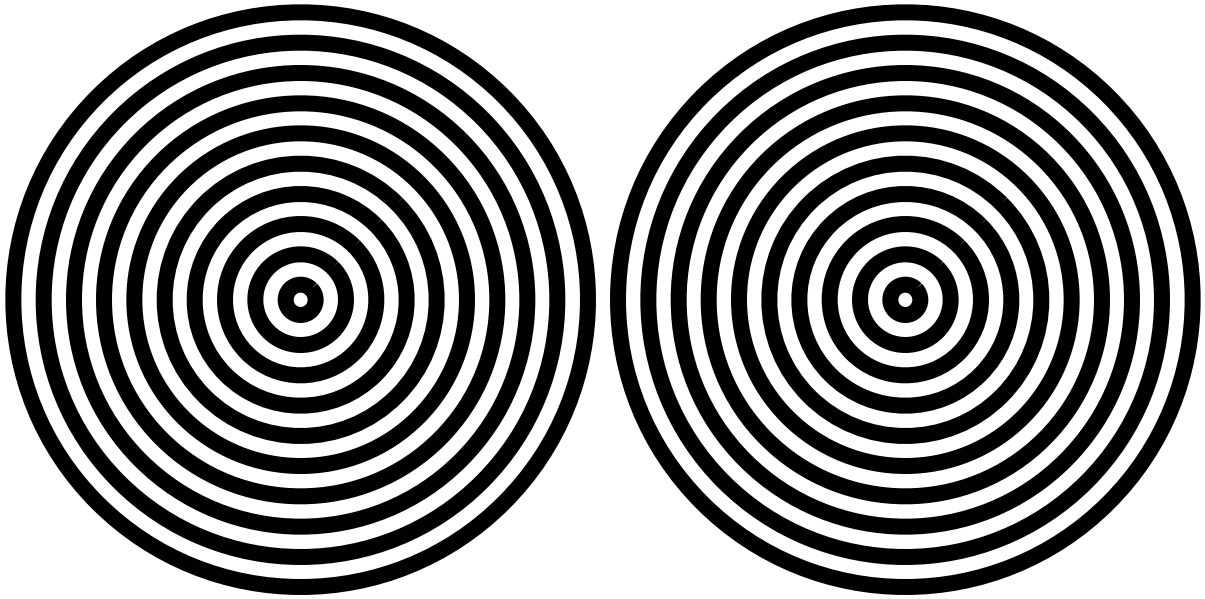








Moiree Muster



Verhinderung von Aliasing-Effekten

- Begrenzung der maximalen Signalfrequenz
  - Bei Digitalkameras: unscharfe Optik
  - Weichzeichner vor dem Bildsensor
- Supersampling mit höherer Samplingfrequenz und dann tiefpassfiltern
- Beispiel: Antialisaing durch Supersampling beim Zeichnen von Linien in einen Framebuffer (siehe 3DCG Teil dieser Vorlesung)

Konvolution, Korrelation

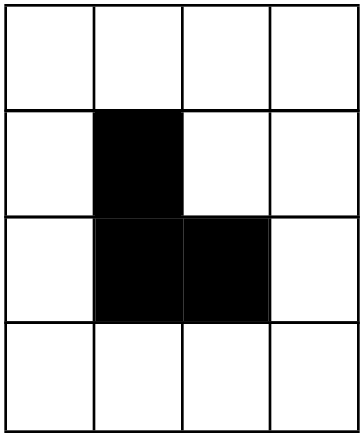
Lineare Operatoren auf Bildern

- Seien f, g abgetastete Bilder, a,b, Skalare
- O ist ein linearer Operator, falls gilt:

$$O(af + bg) = aO(f) + bO(g)$$

- Beispiel: Bild abdunkeln: O(f(i,j)) = 0.5 f(i,j)

Hintereinanderschreiben von Pixeln



- 2-dimensionales Bild wird zu einem langen 1-dimensionalen Vektor

Linearer Operator als Matrix

O lässt sich auch als Matrix ausdrücken:

Seien x,y die hintereinandergeschriebenen Pixel zweier Bilder, dann ist

$$\vec{y} = A\vec{x}$$

Wobei A quadratisch mit Rang der Pixelanzahl und jeder Eintrag $A_{i,j}$ gibt an, mit welchem Gewicht Pixel j aus x auf Pixel i in y abgebildet wird

Verschiebungsinvariante Operatoren

- … sind lineare Operatoren, deren Wirkung unabhängig vom Ort ist:

$$O \circ f(x+a, y+b) = \left[O \circ f\right](x+a, y+b)$$

- Beispiel: gleichmäßige Unschärfe im Bild durch Bewegung der Kamera

Konvolution (Faltung)

- Seien f, g abgetastete Bilder mit unendlicher Größe, m,n, Skalare

$$(g * f)(m,n) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} g(i,j) f(m-i, n-j)$$

Heißt Konvolution der Funktion f mit g

- g heißt die Konvolutionsfunktion
- Funktioniert so nur für unendlich große Bilder

Eigenschaften der Konvolution

- Linear & verschiebungsinvariant
- Kommutativ & assoziativ

$$[g_1 * g_2](m,n) = [g_2 * g_1](m,n)$$

$$g_1 * ([g_2 * g_3](m,n)) = [g_1 * g_2](m,n) * g_3(m,n)$$

- D.h. wir können mehrere Konvolutionen vorab kombinieren und dann gemeinsam anwenden

Konvolution vereinfacht

Hat die Konvolutionsfunktion g nur einen begrenzten Bereich, in dem g != 0, dann heisst dieser Bereich Kern von g (kernel)

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

Konvolution anschaulich

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

	1	2	0	-2	-1	
	2	2	0	-2	-2	
	3	2	0	-2	-3	
	3	2	0	-2	-3	
	2	2	0	-2	-2	

Verwendung der Konvolution



$$*\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} =$$



Punktantwort (point spread function, PSF)

- Faltung eines einzelnen Dirac-Impulses
  - Bild mit einem einzigen schwarzen Pixel
- Abgetastetes Bild = Folge von Dirac-Impulsen * Pixelhelligkeiten
- → Gesamtwirkung durch PSF vollständig beschrieben
- → falls PSF umkehrbar, kann Wirkung theoretisch rückgängig gemacht werden
  - scheitert praktisch an Numerik
- PSF kann experimentell bestimmt werden

Korrelation

- Seien f, g abgetastete Bilder mit unendlicher Größe, m,n, Skalare

$$(g * f)(m,n) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} g(i,j) f(i-m, j-n)$$

Heißt Korrelation der Funktion f mit g

- g heißt die Korrelationsfunktion
- Funktioniert so nur für unendlich große Bilder → Vereinfachung siehe Konvolution

Verwendung der Korrelation



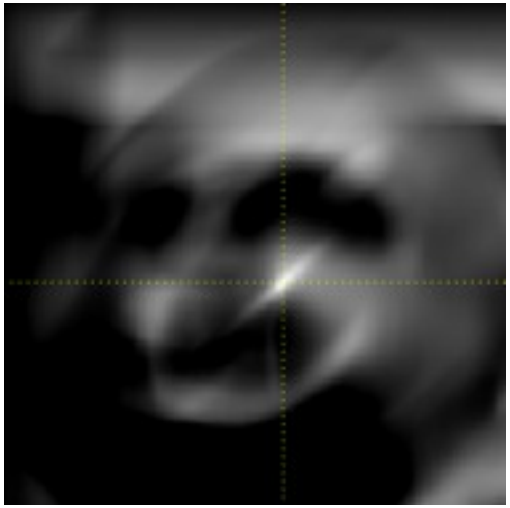
Ausgangsbild (f)



Korrelationsfunktion (g)



Ergebnis (g*f)



..normalisiert

Unterschiede Konvolution/Korrelation

- Nur verschieden in der Reihenfolge der Anwendung von g
- Gleich, falls der Kern von g symmetrisch unter 180° Rotation
- Beispiele:

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

Tiefpass

-1	-1	-1
-1	9	-1
-1	-1	-1

Hochpass

Tiefpass: Wirkung



$*$

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

$=$



Nützlich z.B. gegen Rauschen und Alias-Effekte

Hochpass: Wirkung



$$*\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$



Nützlich zum Scharfzeichnen und Kanten finden

Fouriertransformation

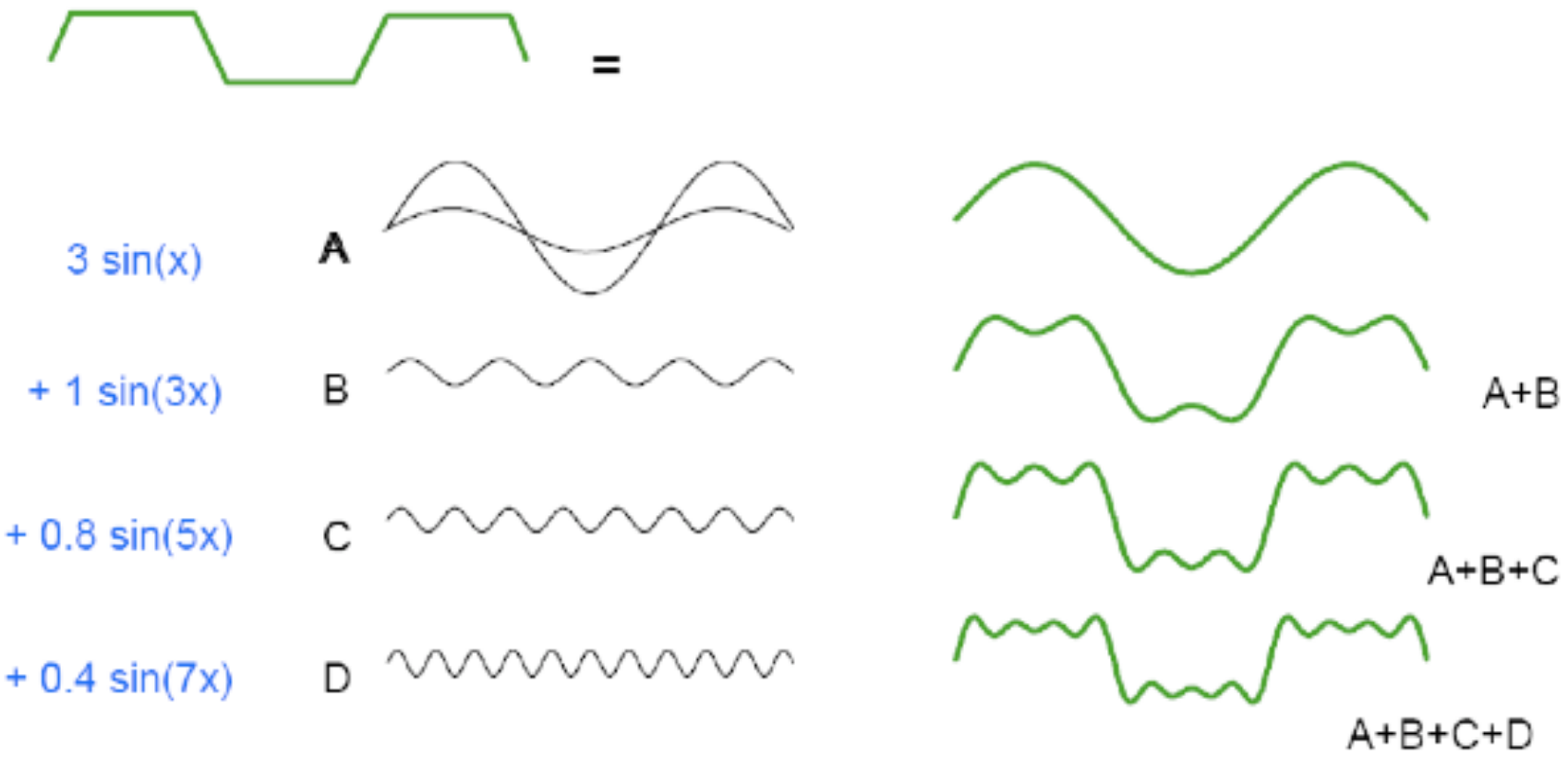
Fourier

- Jean Baptiste Jposeph Fourier (1768-1830)
- Französischer Physiker und Mathematiker
- Erfinder der Fouriertransformation



Fouriertransformation: Grundidee

- Beschreibe beliebige Funktion als gewichtete Summe periodischer Grundfunktionen (Basisfunktionen) mit untersch. Frequenz



Periodische Funktionen

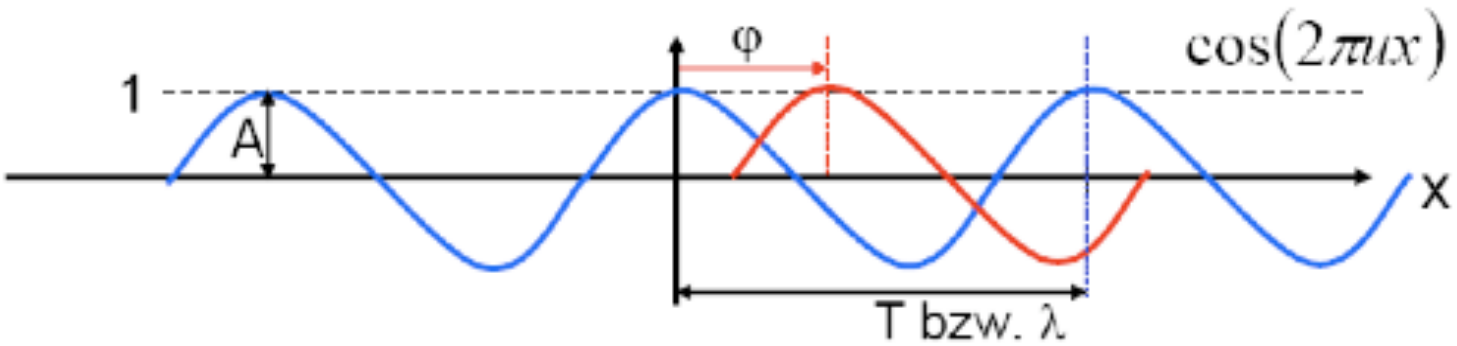
Parameter

A Amplitude: Intensität des Signals

φ Phase: Verschiebung gegenüber dem Ursprung

verschiedene Größen zur Beschreibung der "Frequenz" [Einheit]

zeitlich f(t)	*räumlich f(x)*
T Periodendauer [s]	λ Wellenlänge [m]
f Frequenz f = 1/T [1/s=Hz]	f Raumfrequenz f = 1/λ [1/m]
ω Kreisfrequenz ω = 2πf	k Wellenzahl k=2π/λ



Fouriers Theorem

- **Jede beliebige** periodische **Funktion** läßt sich darstellen als **Summe von cos** und **sin** Funktionen unterschiedlicher Frequenzen.

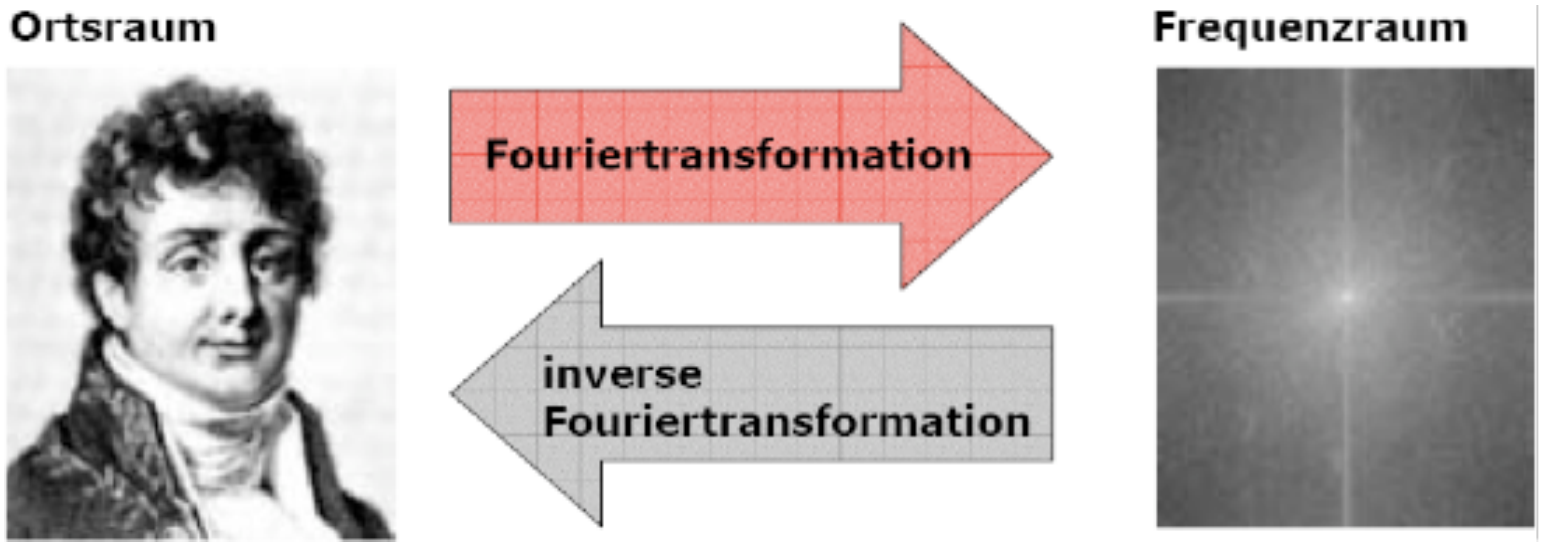
- 1. Ist die Funktion nicht periodisch, aber auf einen bestimmten Definitionsbereich beschränkt, so kann man diesen Bereich einfach kopieren (periodisch fortsetzen) und hat damit wieder eine periodische Funktion.
- 2. Ein Bild kann man als Zeilen und Spalten von nichtperiodischen Funktionen auffassen. Man kann also auch ein Bild fouriertransformieren.

Motivation (1)

- Manche Operationen sind im Ortsraum (d.h. auf den Pixeln des Bildes) schwer
  - Herausfiltern bestimmter Frequenzen
  - Beseitigung störender Details
  - Konvolution, Korrelation
- Ziel: übertrage Bild in einen Raum, in dem diese Operationen leichter sind
  - z.B. Zerlegung des Bildes in Frequenzen
  - Rückweg muss möglich sein!
  - Verschiedene Möglichkeiten, gleiches Prinzip

Motivation (2)

- Bisher: Darstellung des Bildes im **Ortsraum** durch den Grauwert an einem bestimmten Ort
- Jetzt: Darstellung im **Frequenzraum** durch cos und sin Funktionen verschiedener Frequenzen
- Ein Bild kann eindeutig und vollständig in beiden Räumen dargestellt werden.



Fouriertransformation: Eigenschaften

- Transformation: verändert eine Funktion nicht, sondern stellt sie nur anders dar
- Transformation ist umkehrbar → inverse Fouriertransformation

- Analog zum Basiswechsel in der Vektorrechnung

Exkurs: Vektorrechnung

$$\begin{pmatrix}1\\0\\0\end{pmatrix}\begin{pmatrix}0\\1\\0\end{pmatrix}\begin{pmatrix}0\\0\\1\end{pmatrix}$$

bilden eine Basis b_1 des R^3
sind paarweise orthogonal
haben Länge 1

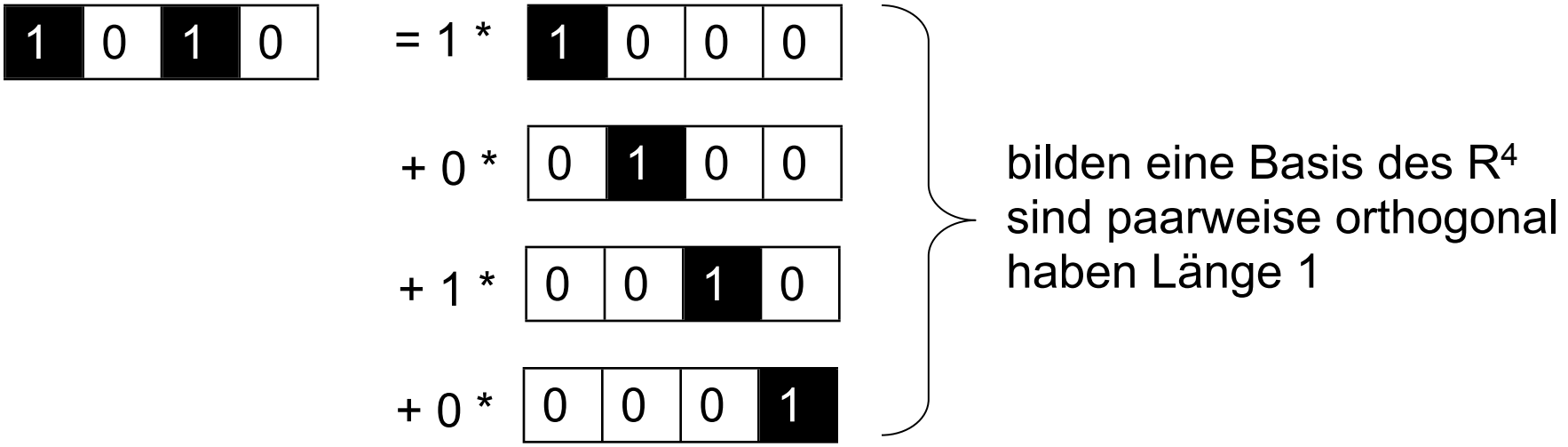
$$\begin{pmatrix}1/\sqrt{2}\\1/\sqrt{2}\\0\end{pmatrix}\begin{pmatrix}1/\sqrt{2}\\-1/\sqrt{2}\\0\end{pmatrix}\begin{pmatrix}0\\0\\1\end{pmatrix}$$

bilden ebenfalls eine Basis b_2 des R^3
sind ebenfalls paarweise orthogonal
Haben ebenfalls Länge 1

$$\begin{pmatrix}1/\sqrt{2} & 1/\sqrt{2} & 0\\1/\sqrt{2} & -1/\sqrt{2} & 0\\0 & 0 & 1\end{pmatrix}$$

Ist orthogonal und normiert
(d.h. $M^T = M^{-1}$)
Ist Basiswechselmatrix
von b_1 nach b_2

Anschaulich: Basisvektoren eines Bildes



- Wahl anderer Basisvektoren ➔ Transformation mittels Basiswechsel
- Basiswechselmatrix vom Rang der Pixelanzahl

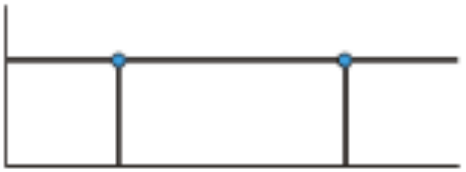
Orthogonale Funktionen

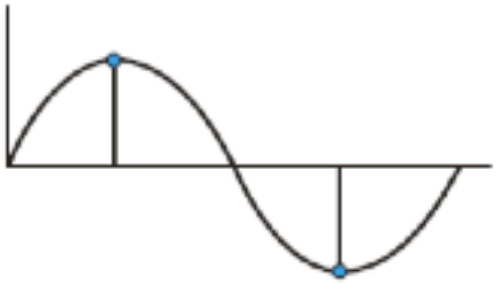
- Seien f_1 und f_2 Funktionen, die an N Stellen abgetastet sind (also N-dim. Vektoren)
- f_1 und f_2 sind orthogonal, falls gilt:

$$f_1 * f_2 = \sum_{k=0}^{N-1} f_1(k) f_2(k) = 0$$

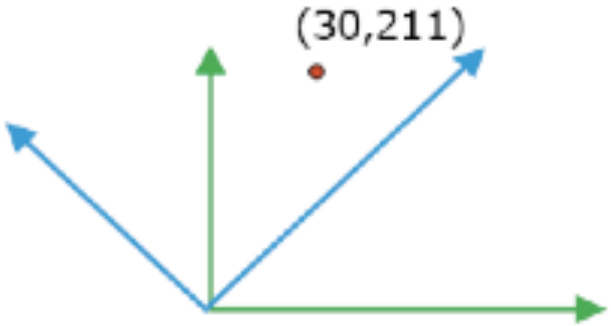
- D.h. das Skalarprodukt der zugehörigen Vektoren ist 0
- N paarweise orthogonale Funktionen $f_1 \ldots f_N$ bilden damit eine orthogonale Basis des N-dim. Raums
- Transformationen zwischen orthogonalen Basen sind immer umkehrbar

Orthogonale Funktionen





Basis:
(1, 1)
(1, -1)



Transformation:

$$(30,211) * \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = (241, -181)$$

Rücktransformation:

$$(241,-181) * \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{T} = (60, 422)$$

Anmerkung:

Das Resultat der Rücktransformation muss skaliert werden, weil die Basis nicht normiert ist.

Orthogonale Funktionstransformationen

- Betrachte abgetastete Funktionen wie Vektoren
- Finde neue geeignete orthogonalen Basis
- Üblicherweise Basisfunktionen, die eine Bedeutung bzgl. der betrachteten Eigenschaft haben
  - Fourier-Basis: komplexe, periodische Funktionen
  - Kosinusbasis: Kosinusfunktionen
- Transformiere Bild in diese Basis
- Betrachte es dort
- Transformiere zurück

$$\vec{y} = A\vec{x}$$

$$\vec{x} = A^{-1}\vec{y}$$

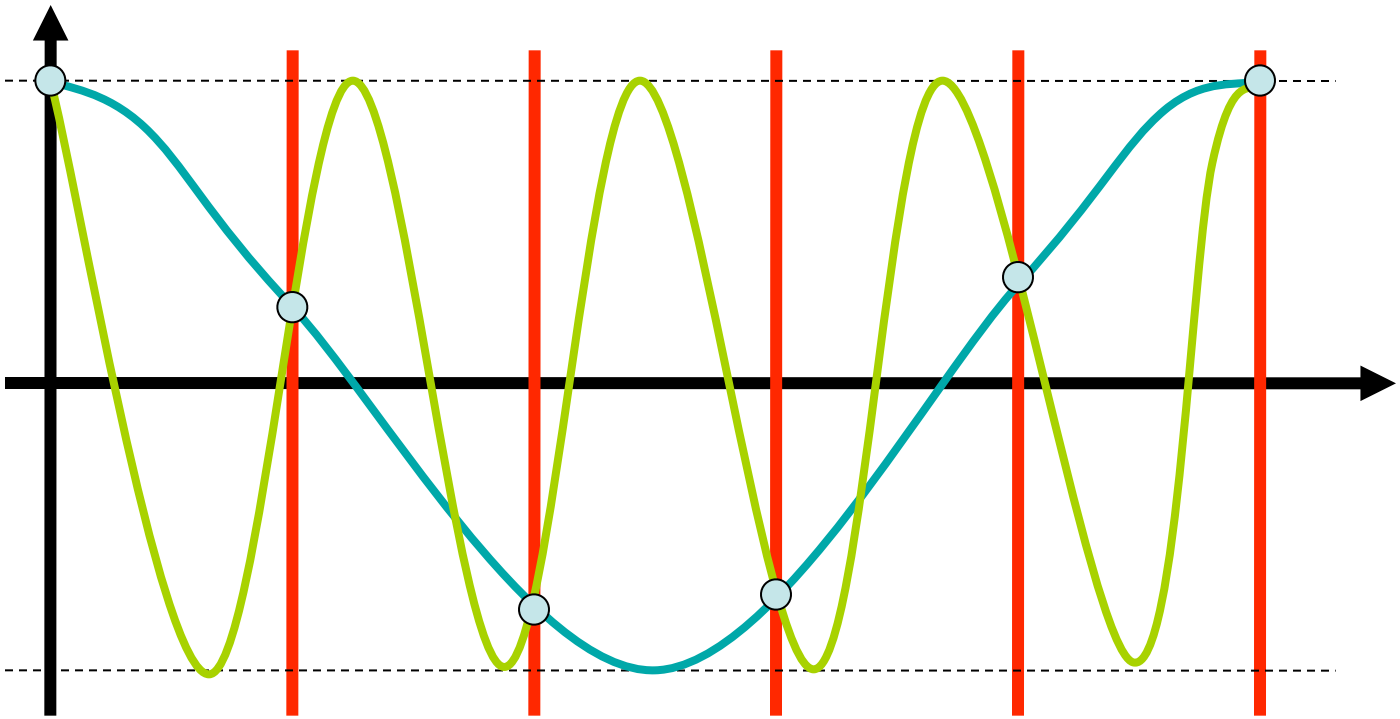
Die Fourierbasis (1)

- Ausgangspunkt: Bildzeile mit N Pixeln
- 1. Versuch: wähle Kosinusfunktionen

$$\cos(u_1 n), \cos(u_2 n), \ldots \cos(u_N n)$$

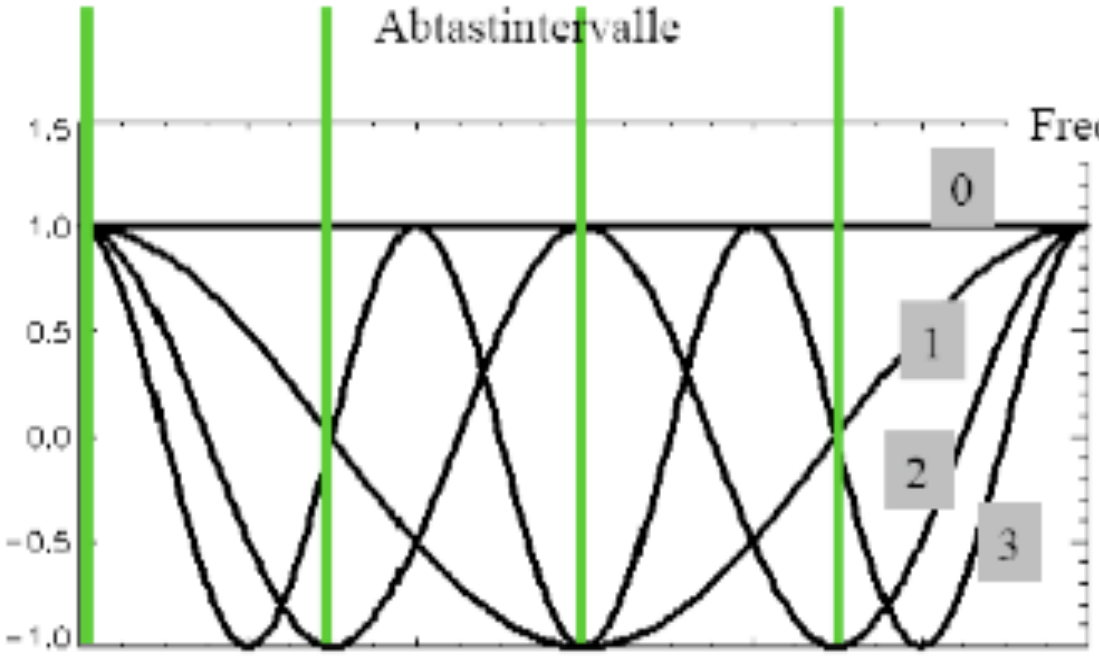
- Wobei u_1 … u_N ganzzahlige Vielfache von $u_0 = 2\pi/N$

Fourierbasis (2)



- N=5, $u_i = 1$, $u_j = 4 = N - u_i$
- Problem: falls $u_i = N - u_j$ ist, sind die Abtastungen gleich
- ➔ nur N/2 Funktionen verfügbar, keine Basis
- Siehe auch Shannon-Nyquist Theorem, Aliasing

Fourierbasis (3)



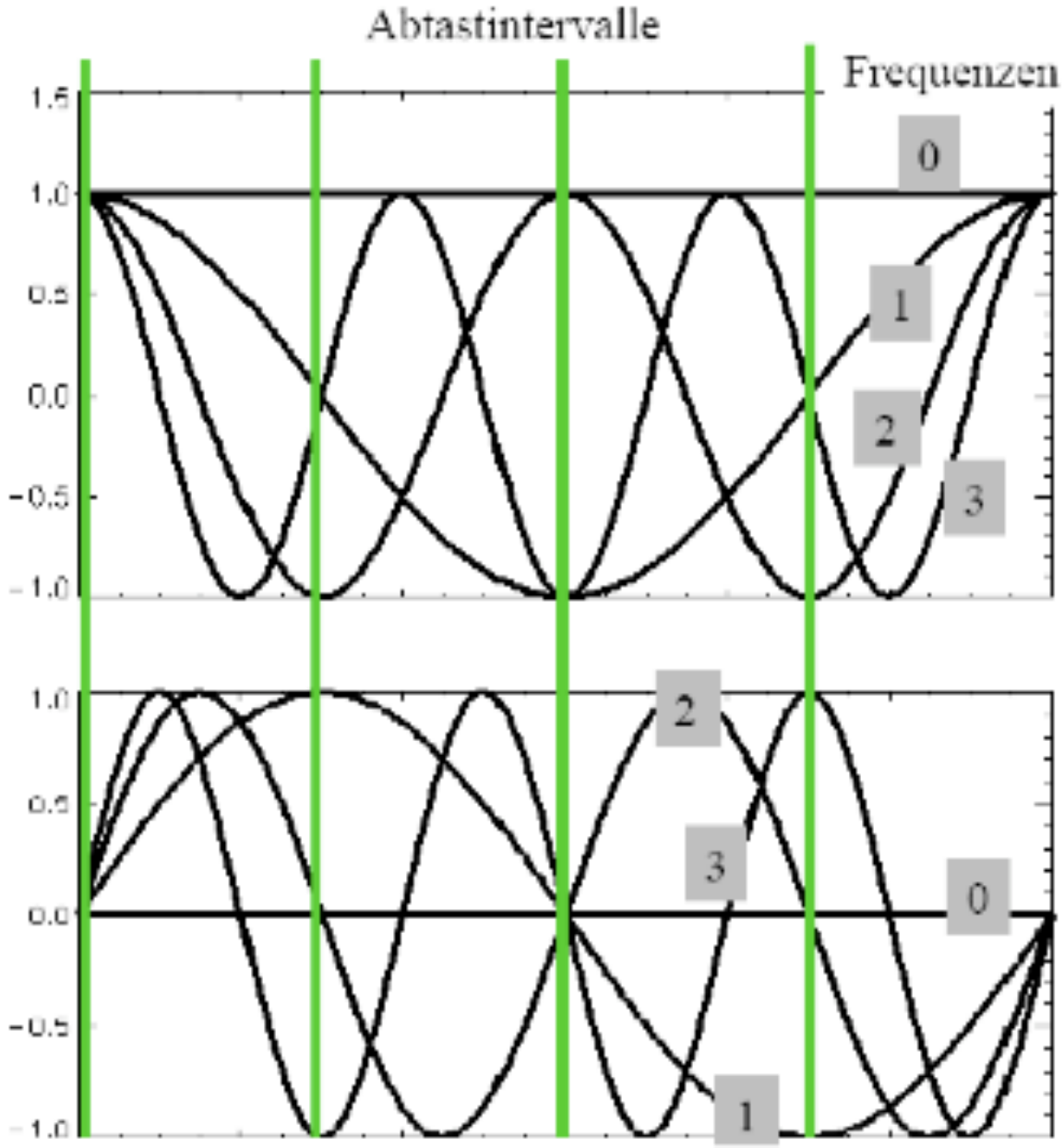
Abgetastete Kosinuswellen:

(1 1 1 1)
(1 0 −1 0)
(1 −1 1 −1)
(1 0 −1 0)

Lösungen:

– frequenzverschobene Perioden (DCT).

– komplexe periodische Funktionen (FT).



Basis-funktions-paare

Abgetastete Kosinuswellen:

(1 1 1 1)
(1 0 –1 0)
(1 –1 1 –1)
(1 0 –1 0)

Abgetastete Sinuswellen:

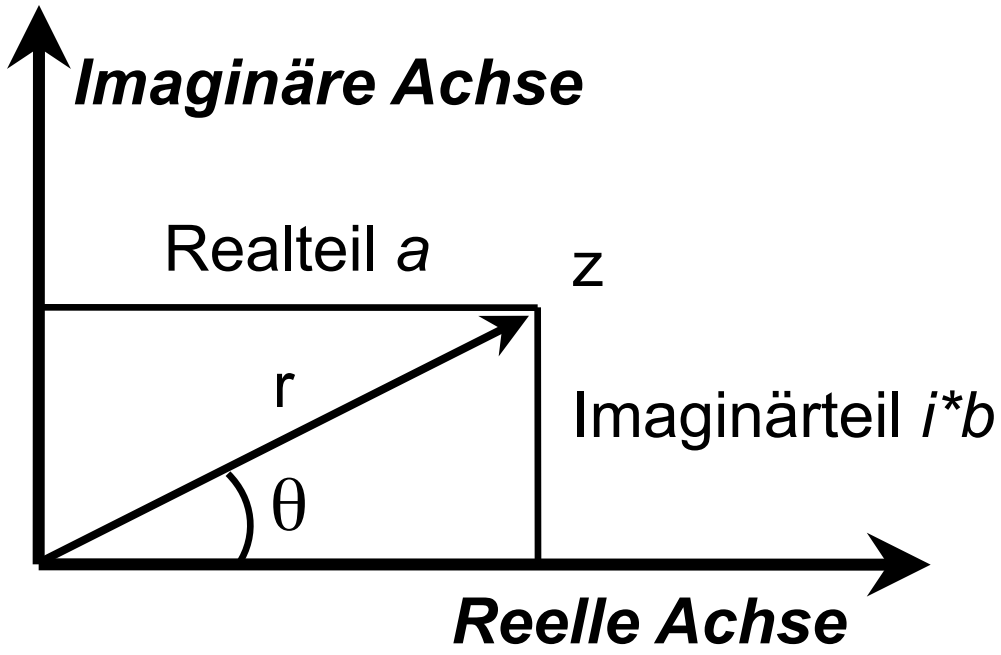
(0 0 0 0)
(0 1 0 –1)
(0 0 0 0)
(0 –1 0 1)

Erinnerung: komplexe Zahlen

- Im Bereich der reellen Zahlen gibt es keine Lösung für $x^2 = -1$
  - Einführung der imaginären Zahlen
  - i: Lösung der Gleichung $x^2 = -1$
- Die Gruppe der reellen und imaginären Zahlen nennt man *komplexe* Zahlen

- Komplexe Zahl *z*: $\boldsymbol{z = a + i*b}$, wobei a und b reell sind

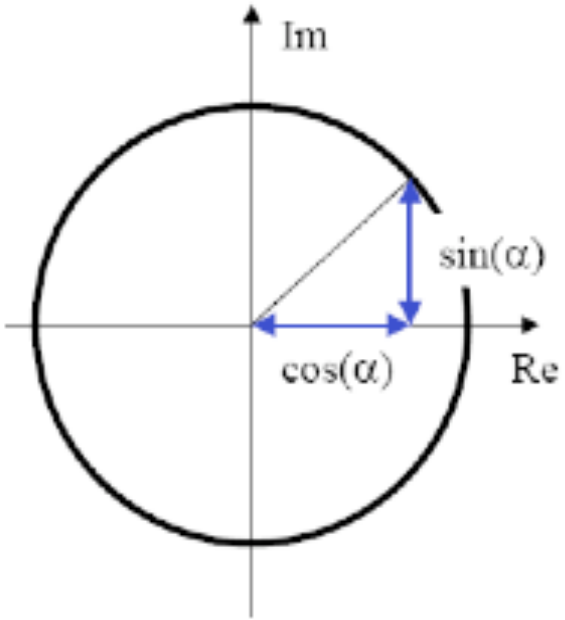
Erinnerung: komplexe Zahlen



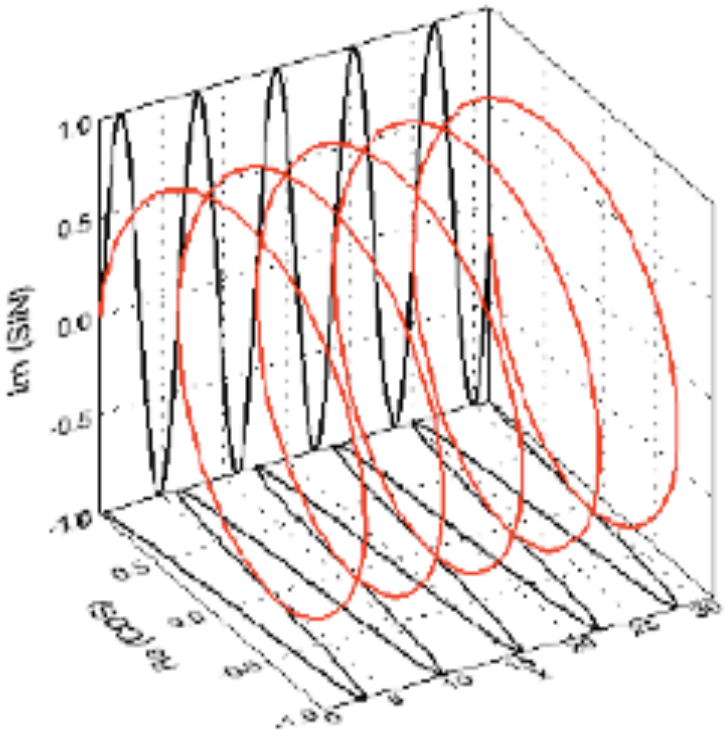
$$z = a + ib = r \times e^{i\theta} = r(\cos(\theta) + i \times \sin(\theta))$$

Komplexe periodische Funktionen

$$\cos(\alpha) + i \cdot \sin(\alpha)$$



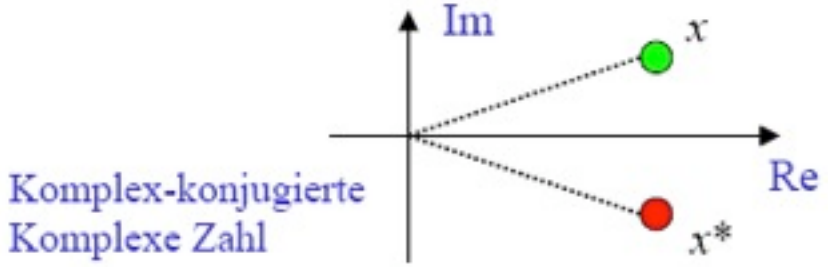
Alle Werte für komplexe Zahlen der Form $\cos(\alpha)+i\sin(\alpha)$ liegen auf einem Kreis mit Abstand 1 in der komplexen Ebene.



Komplexes Skalarprodukt

Skalarprodukt zwischen zwei Vektoren mit komplexen Elementen:

- Summe der Produkte der Komponenten des ersten Vektors mit der komplex-konjugierten Komponenten des zweiten Vektors.
- Die komplex-konjugierte zu $x=a+i\cdot b$ ist $x^*=a-i\cdot b$.



Skalarprodukt:

$$\vec{x} \bullet \vec{y} = \sum_{i=0}^{N-1} x_i \cdot y_i^* = \sum_{i=0}^{N-1} (\mathrm{Re}(x_i) + i\,\mathrm{Im}(x_i))(\mathrm{Re}(y_i) - i\,\mathrm{Im}(y_i))$$

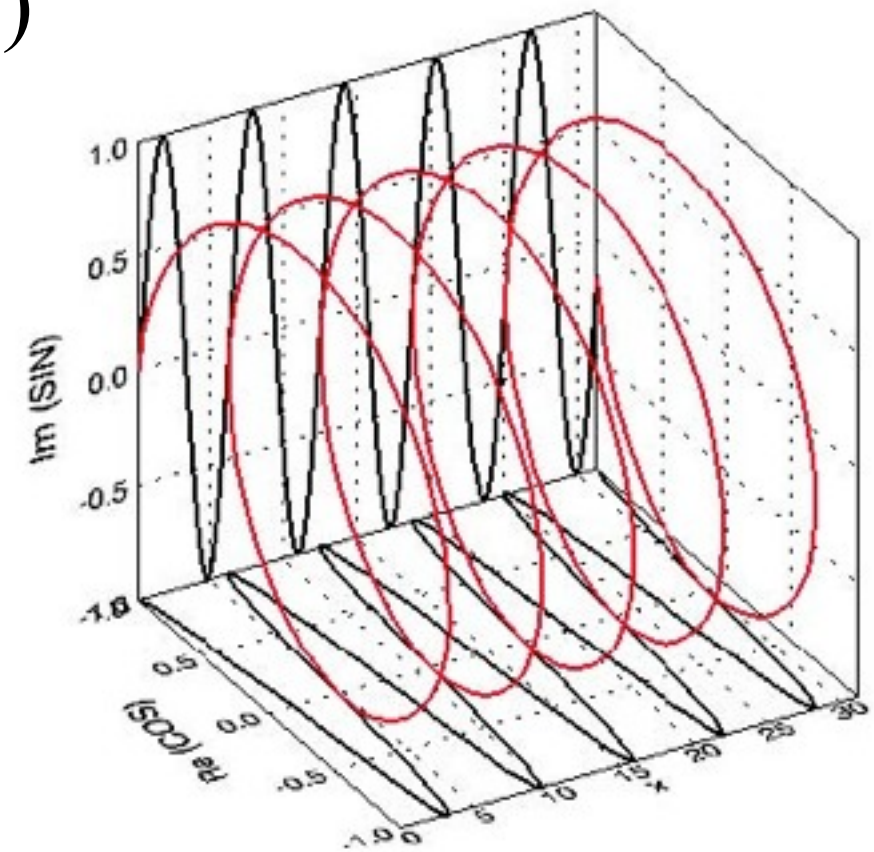
Fourierbasis (3)

- 2. Versuch: wähle komplexe Funktionen

$$f = \cos(un) + i\sin(un)$$

- Wobei u ein ganzz. Vielf. von $u_0 = 2\pi/N$

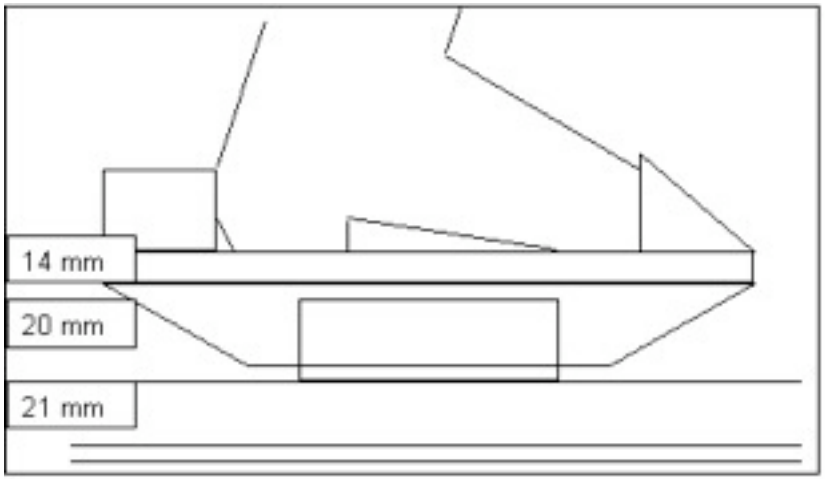
- → N verschiedene Funktionen
- Ist eine Basis



Belastungen beim Carving

Technikanalyse und Kräfte beim Carving

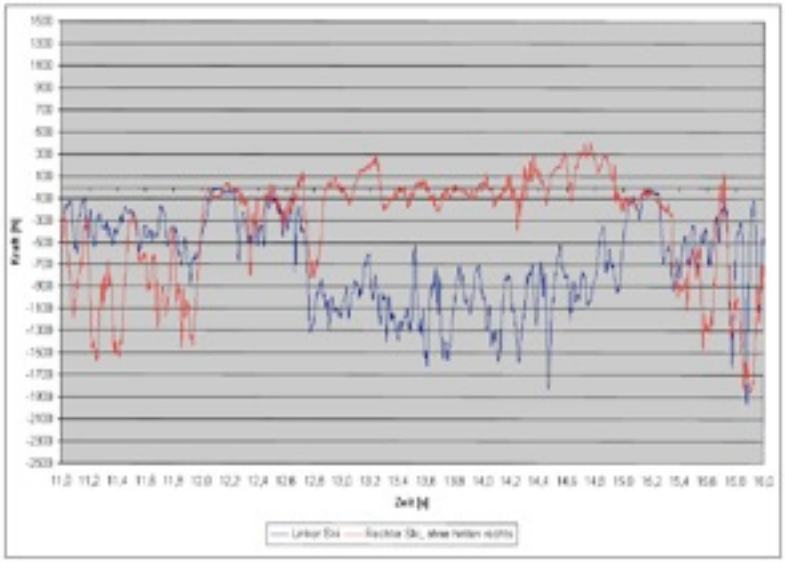
Diplomarbeit zur Erlangung des Eidg. Turn- und Sportlehrerdiploms II







- „Die erheblichen Schwingungen erlauben es auch nicht, eindeutige Aussagen bezüglich absoluter Kräfte zu machen. Eine Fourier- Analyse hätte diesbezüglich eine Vereinfachung, […] der Diagrammauswertung gegeben.“



Computergraphik 1
2. Teil: Bildverarbeitung

Fouriertransformation Ende

FFT, Bildrestauration mit PSF

Transformation, Interpolation

Repräsentation als Exponentialfunktion

- Taylorreihenentwicklung für Kosinus und Sinus:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots \quad \sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

- Taylorreihenentwicklung für e^{ix}:

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots \Rightarrow e^{ix} = 1 + \frac{ix}{1!} + \frac{(ix)^2}{2!} + \frac{(ix)^3}{3!} + \ldots$$

- Es gilt daher wegen $i^2 = -1$: $\cos(x) + i \cdot \sin(x) = e^{ix}$
- Phasenverschiebung α kann in komplexen Funktionen als Multiplikation ausgedrückt werden:

$$\cos(x+\alpha) + i\sin(x+\alpha) = e^{i(x+\alpha)} = e^{ix}e^{i\alpha}$$

1D-Basisfunktionen

Bildfunktion: $f(n)$, n=0,N-1,



also: N Basisfunktionen

$b_u(n) = \exp(i \cdot 2\pi/N \cdot n \cdot u)$, mit Frequenzen u=0,N-1

z.B. $b_0(n) = [\ (1,1), (1,1), \ldots , (1,1)\ ]$

Transformation **FT** : $\mathbf{FT(f) = F = f \cdot B}$ (Vektor-Matrix-Schreibweise)

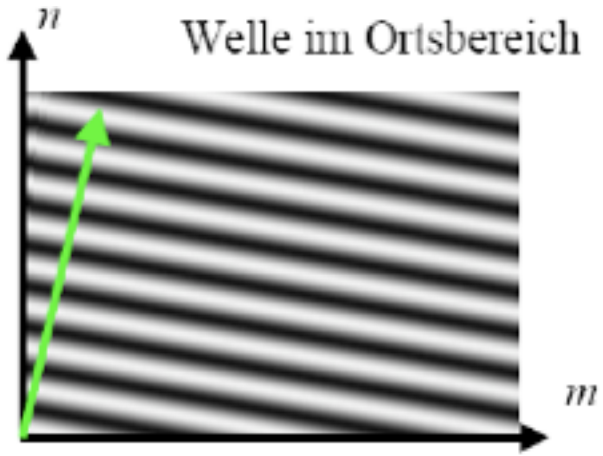
$F(u) = \sum_m f(n) \cdot \exp(-i \cdot 2\pi/N \cdot n \cdot u)$, für alle u=0,N-1

Rücktransformation $\mathbf{FT^{-1}}$: $\mathbf{FT^{-1}(F) = F \cdot B^T}$ (Vektor-Matrix-Schreibweise)

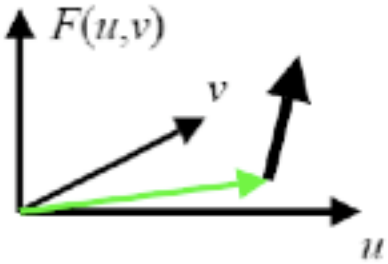
$f(n) = 1/N \cdot \sum_u F(u) \cdot \exp(i \cdot 2\pi/N \cdot n \cdot u)$, für alle n=0,N-1

Skalierungsfaktor, weil die Basisfunktionen nicht normiert sind.

2D-Basisfunktionen







(komplexer) Funktionswert im Frequenzbereich

Basisfunktionen sind Wellen (Frequenz, Richtung, Amplitude, Phase):

$$\exp(i \cdot 2\pi/N \cdot (mu+nv))$$

Richtung ist durch Vektor $(u\ v)$ gegeben.

Die Basisfunktionen der 2-D Fourier-transformation sind zerlegbar:

$$\exp(i \cdot 2\pi/N \cdot (mu+nv)) = \exp(i \cdot 2\pi/N \cdot m \cdot u) \cdot \exp(i \cdot 2\pi/N \cdot n \cdot v)$$

2D-Fouriertransformationspaar

$$F(u,v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \cdot e^{-i \cdot 2\pi \cdot \left(\frac{um}{M} + \frac{vn}{N} \right)}$$

$$f(m,n) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} F(u,v) \cdot e^{i \cdot 2\pi \cdot \left(\frac{um}{M} + \frac{vn}{N} \right)}$$

Transformationspaar für Bilder der Größe MxN

Transformationspaar für quadratische Bilder der Größe NxN

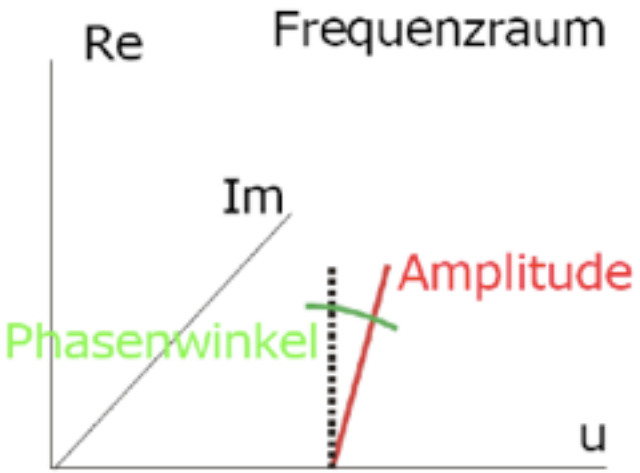
$$F(u,v) = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) \cdot e^{-i \cdot \frac{2\pi}{N} \cdot (um+vn)}$$

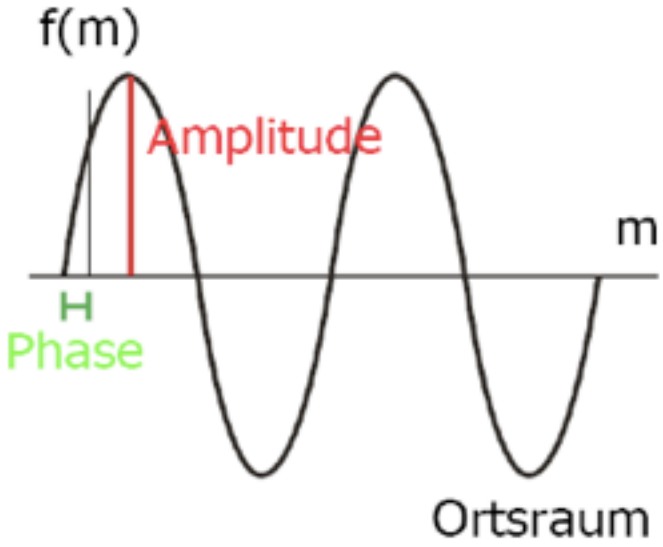
$$f(m,n) = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} F(u,v) \cdot e^{i \cdot \frac{2\pi}{N} \cdot (um+vn)}$$

Phase und Amplitude

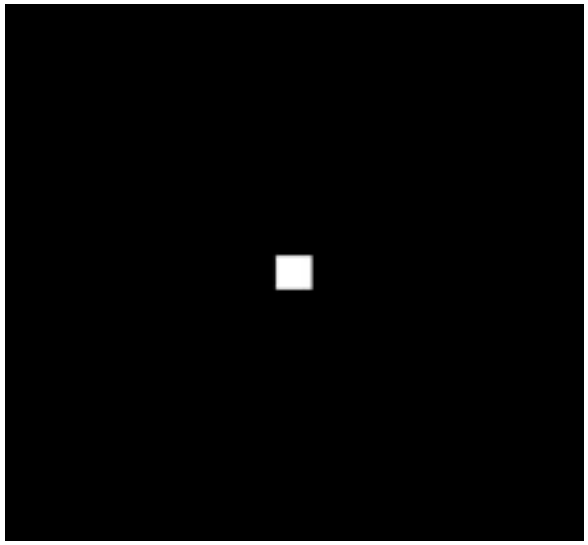
Das Resultat der Fouriertransformation ist eine komplexe Funktion $F(u)$.

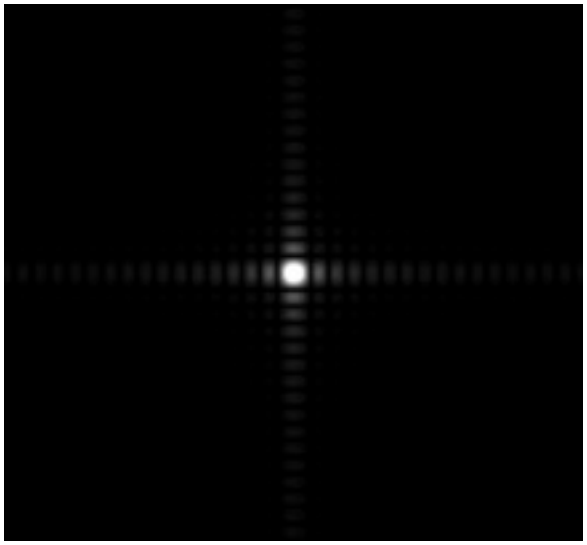
Der Betrag eines Funktionswerts ist die Amplitude und der Winkel zur reellen Achse ist die Phase zur Gewichtung der betreffenden Basisfunktion



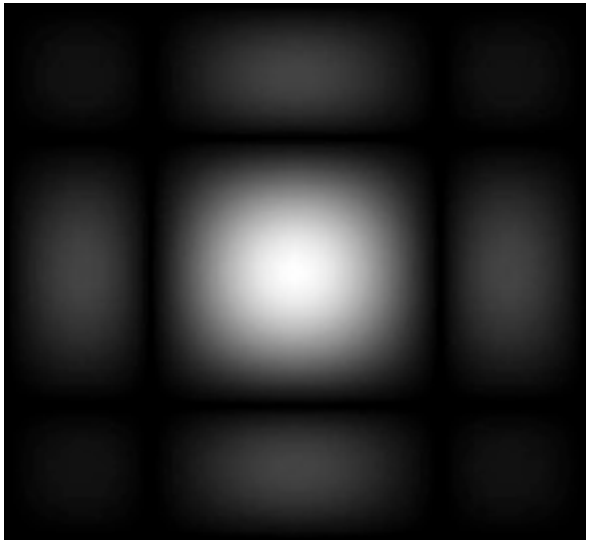


Beispiele f. Amplitude









Darstellungsweise

Original

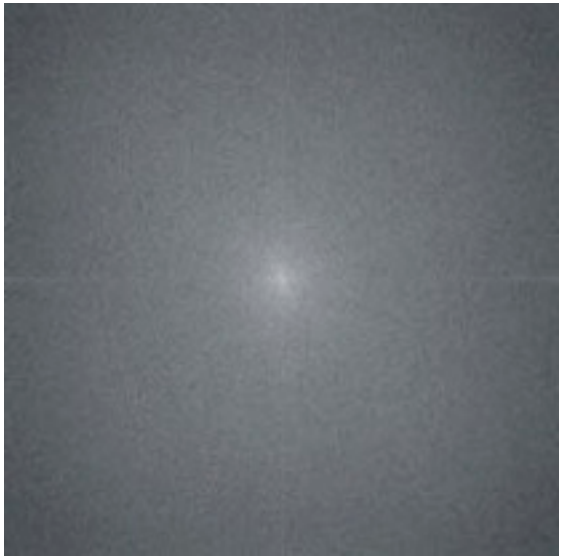




Amplitude (zentriert, d.h. von –N/2 bis N/2)

Amplitude (log. Skala)



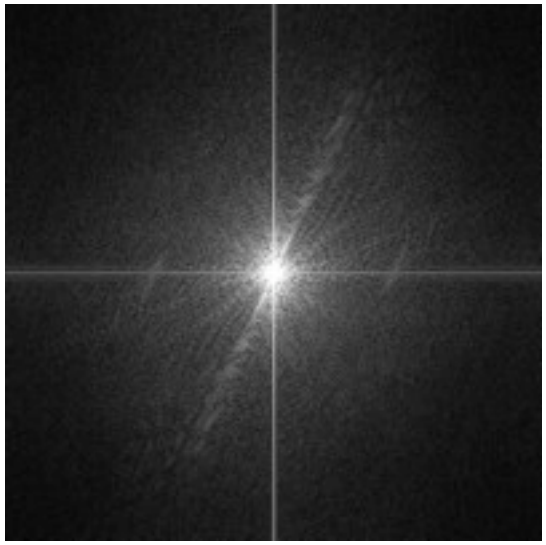


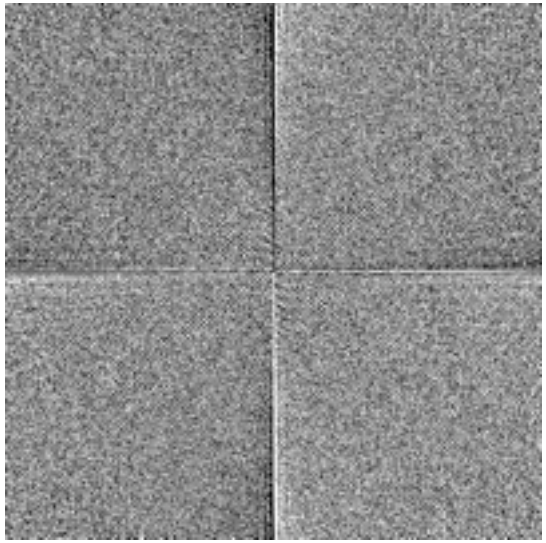
Amplitude (zentriert, log. Skala)

Beispiele



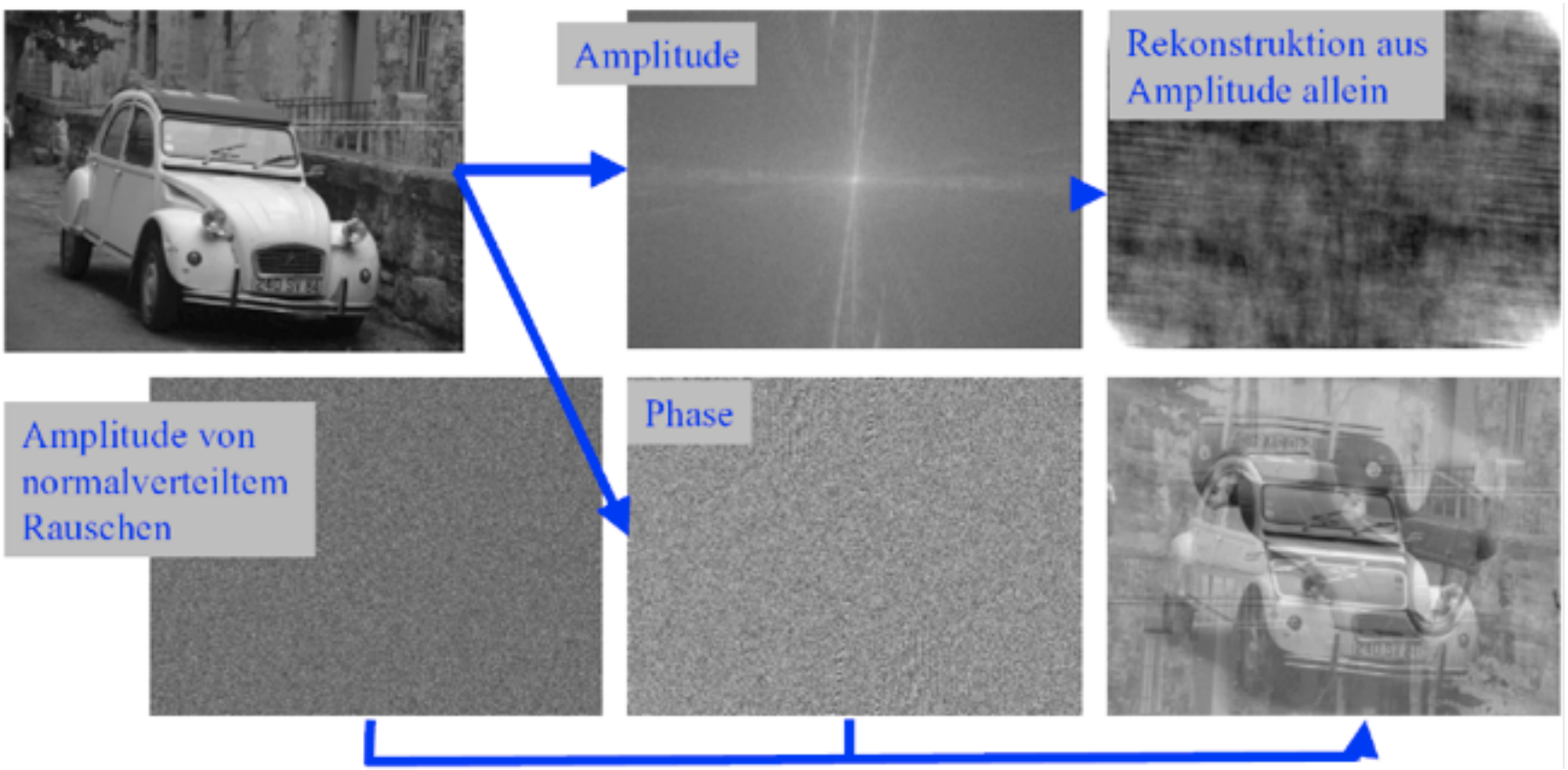
Amplitude



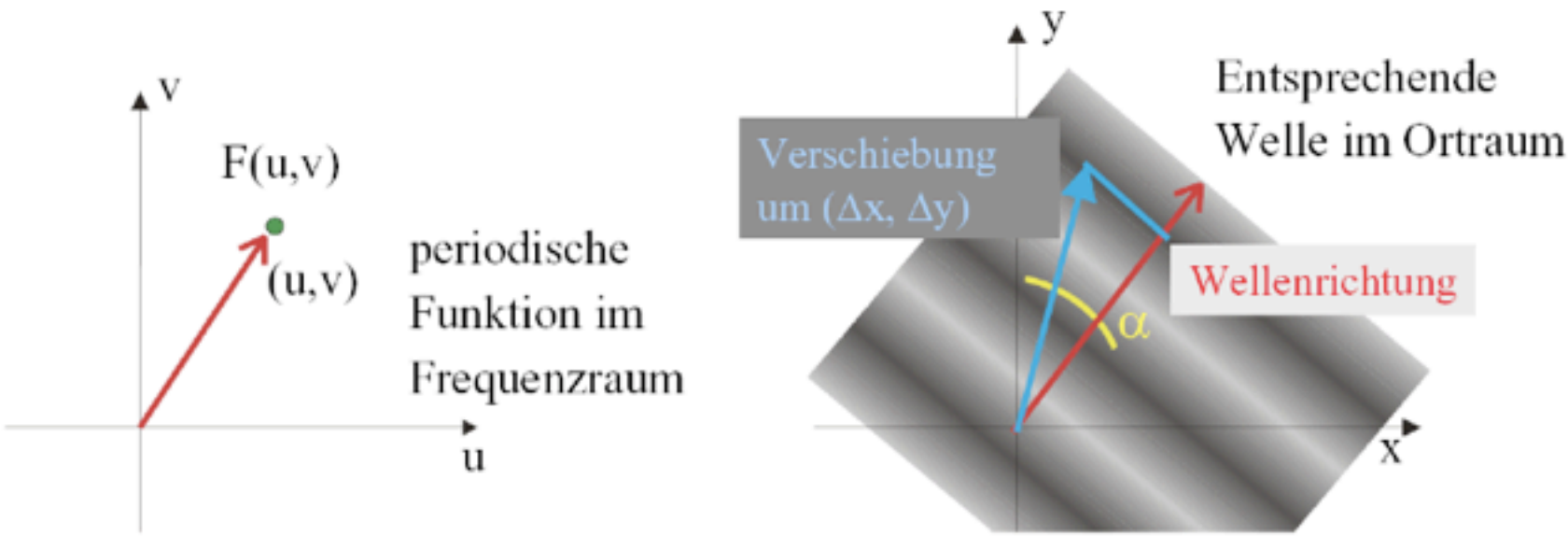


Phase

Einfluss von Amplitude und Phase

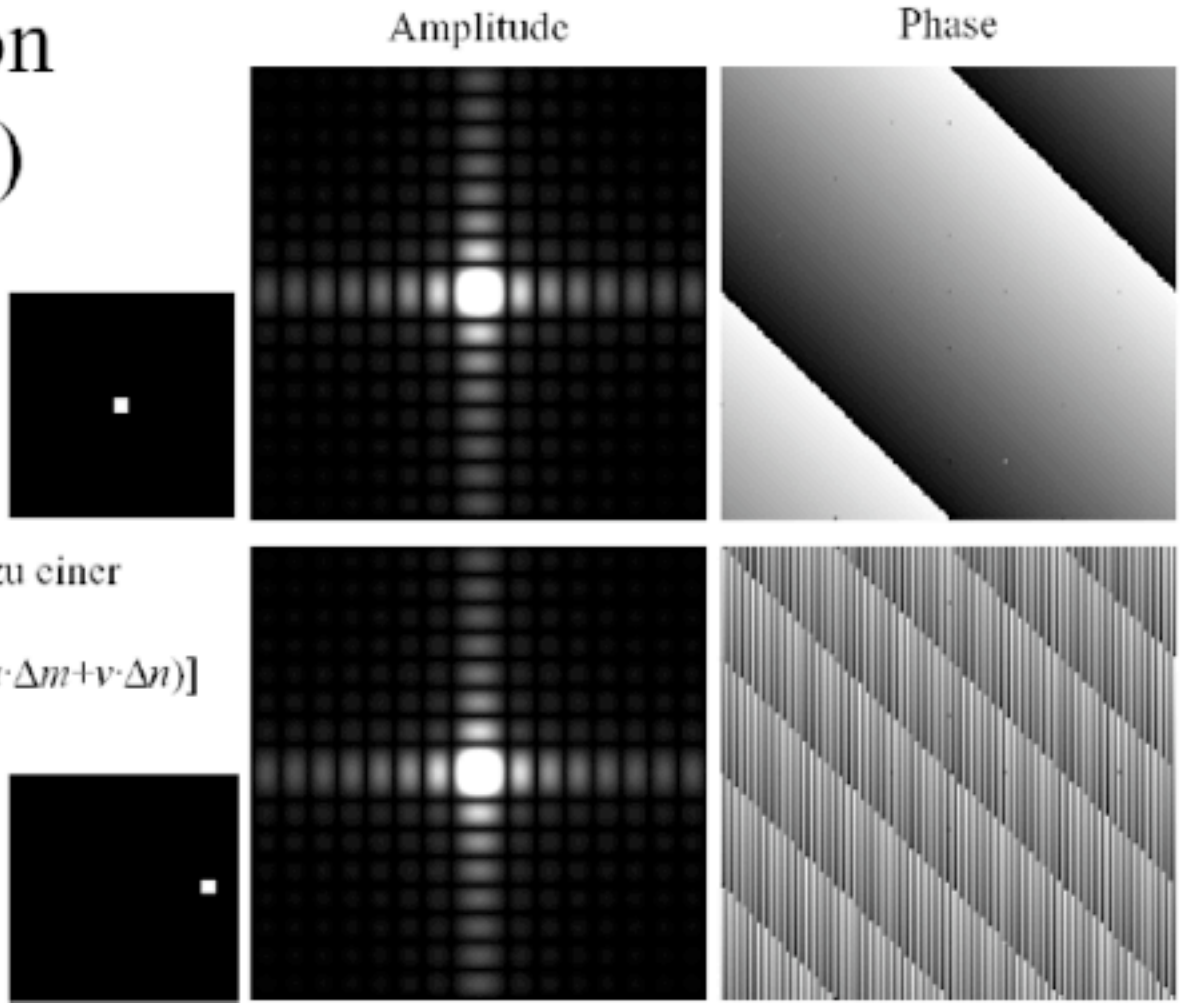


Translation



- Translation im Ortsbereich führt zu einer Translation der zusammensetzenden Wellen.
- Umfang der Translation hängt vom Unterschied zwischen Wellenrichtung (u v) und Translationsrichtung (Δx Δy) ab.
- Im Frequenzbereich bedeutet die Translation eine Phasenverschiebung.

Translation (Beispiel)



Translation um $(\Delta m, \Delta n)$ führt zu einer Phasenverschiebung

$$F'(u,v) = F(u,v) \cdot \exp[-i \cdot 2\pi / N \cdot (u \cdot \Delta m + v \cdot \Delta n)]$$

Phasenverschiebung

$p_{u,v} = |(\Delta x, \Delta y)| \cdot \cos(\alpha) / T_{u,v}$ mit

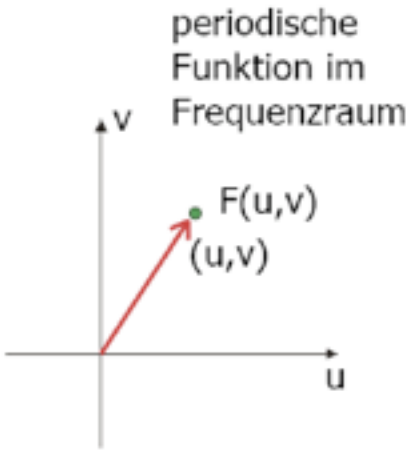
$\cos(\alpha)$ - Winkel zwischen Wellenrichtung und Richtung von $(\Delta x, \Delta y)$

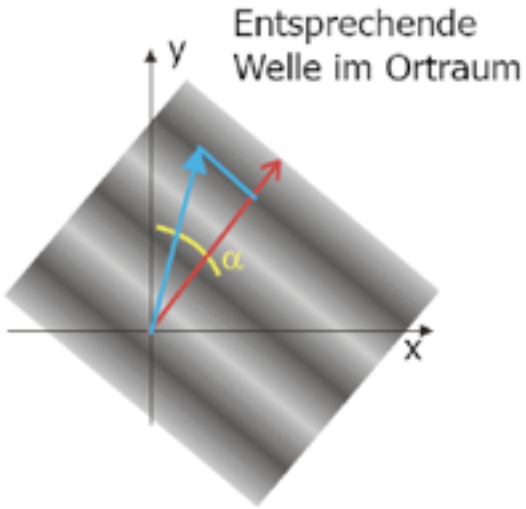
$T_{u,v}$ - Wellelänge = 1/Frequenz

$$\cos(\alpha) = \frac{\langle(\Delta x, \Delta y),(u,v)\rangle}{|(\Delta x, \Delta y)| \cdot |(u,v)|}$$

$$T_{u,v} = N/2\pi \cdot |(u,v)|$$

$$\Rightarrow p_{u,v} = \exp(i \cdot 2\pi/N \cdot (u\Delta x + v\Delta y))$$





Rotation: $F(u,v)$ wird in gleicher Weise rotiert wie $f(m,n)$.



Periodizität und Symmetrie

- Für ein- und zweidimensionale Funktionen mit M bzw. M und N Werten gilt:
  - $F(u) = F(u+M), f(m)=f(m+M)$
  - $F(u,v) = F(u+M,v) = F(u,v+N) = F(u+M,v+N)$
  - $f(m,n) = f(m+M,n) = f(m,n+N) = f(m+M,n+N)$
- Für reellwertige Funktionen f gilt für die Fouriertransformierte:
  - $F(u) = {}^*F(-u)$
  - $F(u,v) = {}^*F(-u,-v)$

  (reduziert die zu berechnenden Werte um die Hälfte)

  *x=a-ib ist die komplex-konjugierte der komplexen Zahl a=a+ib.

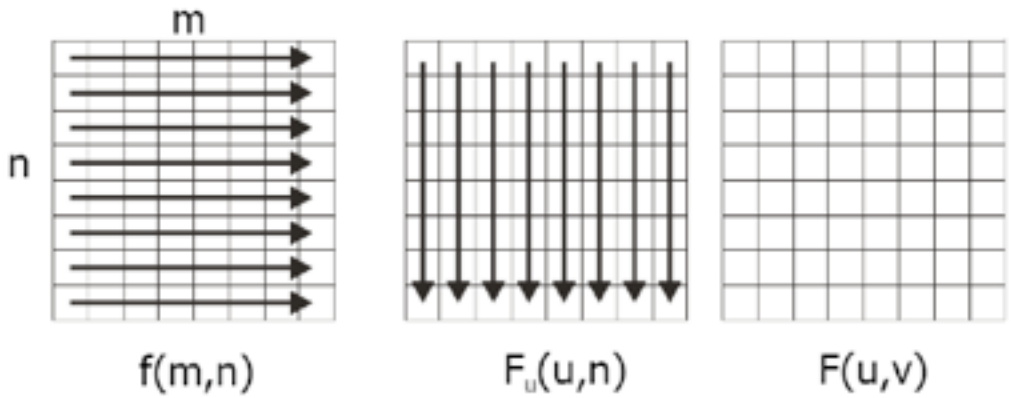
Separabilität

Die Fouriertransformation ist separabel, d.h., sie kann zunächst in M-Richtung und anschließend auf diesen Zwischenergebnissen in N-Richtung ausgeführt werden.

$$
\begin{aligned}
F(u,v) &= 1/N^2 \cdot \sum_v \sum_\mu f(m,n) \cdot \exp(-i \cdot 2\pi \cdot (um+vn)/N) \\
&= 1/N^2 \cdot \sum_v \sum_\mu f(m,n) \cdot \exp(-i \cdot 2\pi \cdot um/N) \cdot \exp(-i \cdot 2\pi \cdot vn/N) \\
&= 1/N^2 \cdot \sum_v \left[\sum_\mu f(m,n) \cdot \exp(-i \cdot 2\pi \cdot um/N)\right] \cdot \exp(-i \cdot 2\pi \cdot vn/N) \\
&= 1/N^2 \cdot \sum_v F_u(m,n) \cdot \exp(-i \cdot 2\pi \cdot vn/N)
\end{aligned}
$$

kann aus der inneren Summe ausgeklammert werden.

Reduziert den Berechnungsaufwand von $O(N^4)$ auf $O(N^3)$.



Konvolution und Korrelation

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

	1	2	0	-2	-1	
	2	2	0	-2	-2	
	3	2	0	-2	-3	
	3	2	0	-2	-3	
	2	2	0	-2	-2	

Worst case Aufwand im Ortsraum: N^4 für Kantenlänge N des Bildes

Konvolution und Korrelation

- Konvolution und Korrelation sind zwei eng verwandte Filter-Operationen.
- Beide können im Ortsraum und im Frequenzraum ausgeführt werden.
- Die Operation im Frequenzraum ist eine einfache Multiplikation (Aufwand N^2).

Konvolution im Frequenzraum

$$F(u)\cdot H(u) = \sum_k f(k)\cdot\exp(-i2\pi uk/N)\cdot\sum_m h(m)\cdot\exp(-i2\pi um/N)$$

$$= \sum_m\sum_k \left[f(k)\cdot\exp(-i2\pi uk/N)\cdot h(m)\cdot\exp(-i2\pi um/N)\right]$$

$$= \sum_m\sum_k \left[f(k)\cdot h(m)\cdot\exp(-i2\pi uk/N)\cdot\exp(-i2\pi um/N)\right]$$

(Verschiebeeigenschaft $h(m)\cdot\exp(-i2\pi uk/N) = h(m-k)$)

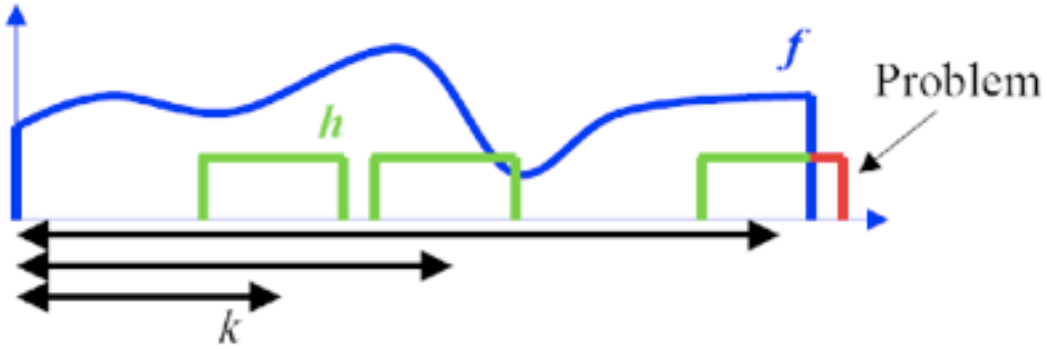
$$= \sum_m\sum_k f(k)\cdot h(m-k)\cdot\exp(-i2\pi um/N)$$

$$= \sum_m \left[\sum_k f(k)\cdot h(m-k)\right]\cdot\exp(-i2\pi um/N)$$

$$= \mathbf{FT}\left[\sum_k f(k)\cdot h(m-k)\right] = \mathbf{FT}\left[[f*h](m)\right]$$

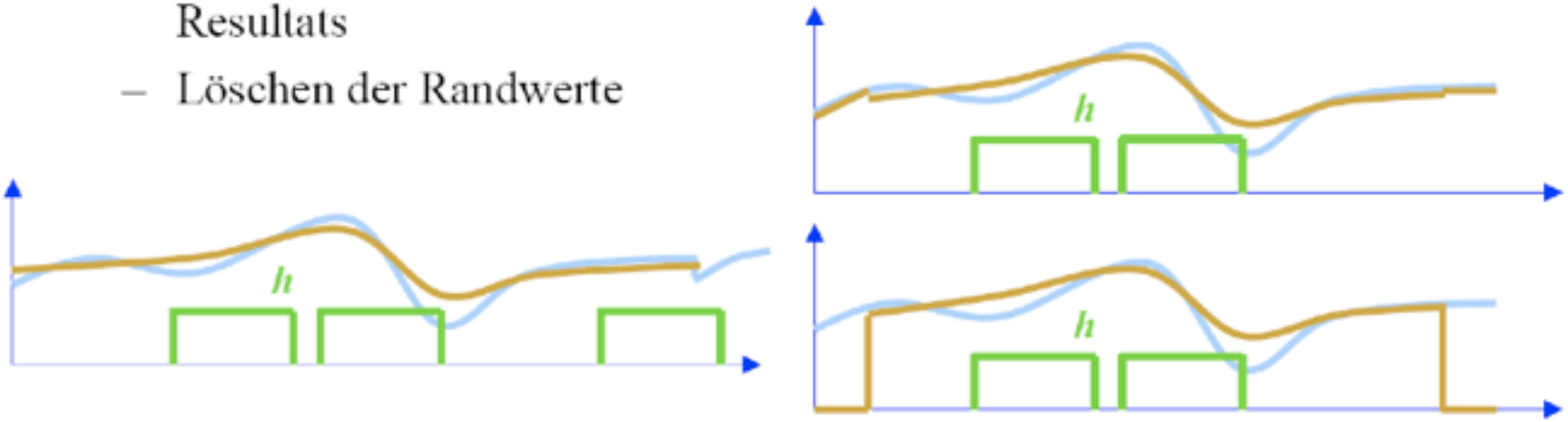
Konvolutionskern

- Konvolution (1D): $[f*h](m)=\Sigma_{k=-\infty,\infty} f(k)\cdot h(m-k)$
- Die meisten Konvolutionsfunktionen h sind nur in einem kleinen Intervall von Null verschieden (Konvolutionskern).
- Mit $h(n)\neq 0$ für $-K\leq n\leq K$ ist: $[f*h](m)=\Sigma_{k=-K,K} f(k)\cdot h(m-k)$.
- Problem: **Rand** von f.



Konvolution am Bildrand

- Lösung 1: periodische Fortsetzung des Bildes
  (diese Lösung wird automatisch gewählt, wenn im Frequenzraum multipliziert wird)
- Lösung 2: Rand ist undefiniert
  - Beibehaltung des ursprünglichen Resultats
  - Löschen der Randwerte





*

Konvolution am Bildrand







Diskrete Kosinustransformation (DCT)

- Zerlegung in Wellen unterschiedlicher Frequenz.
- Sehr ähnlich zur Fouriertransformation.
- Basisfunktionen sind reell.
- Wird vor allem bei Kompressionsverfahren verwendet(JPEG).
- ***Wie lassen sich genügend reellwertige Basisfunktionen finden?***

Basisfunktionen der DCT

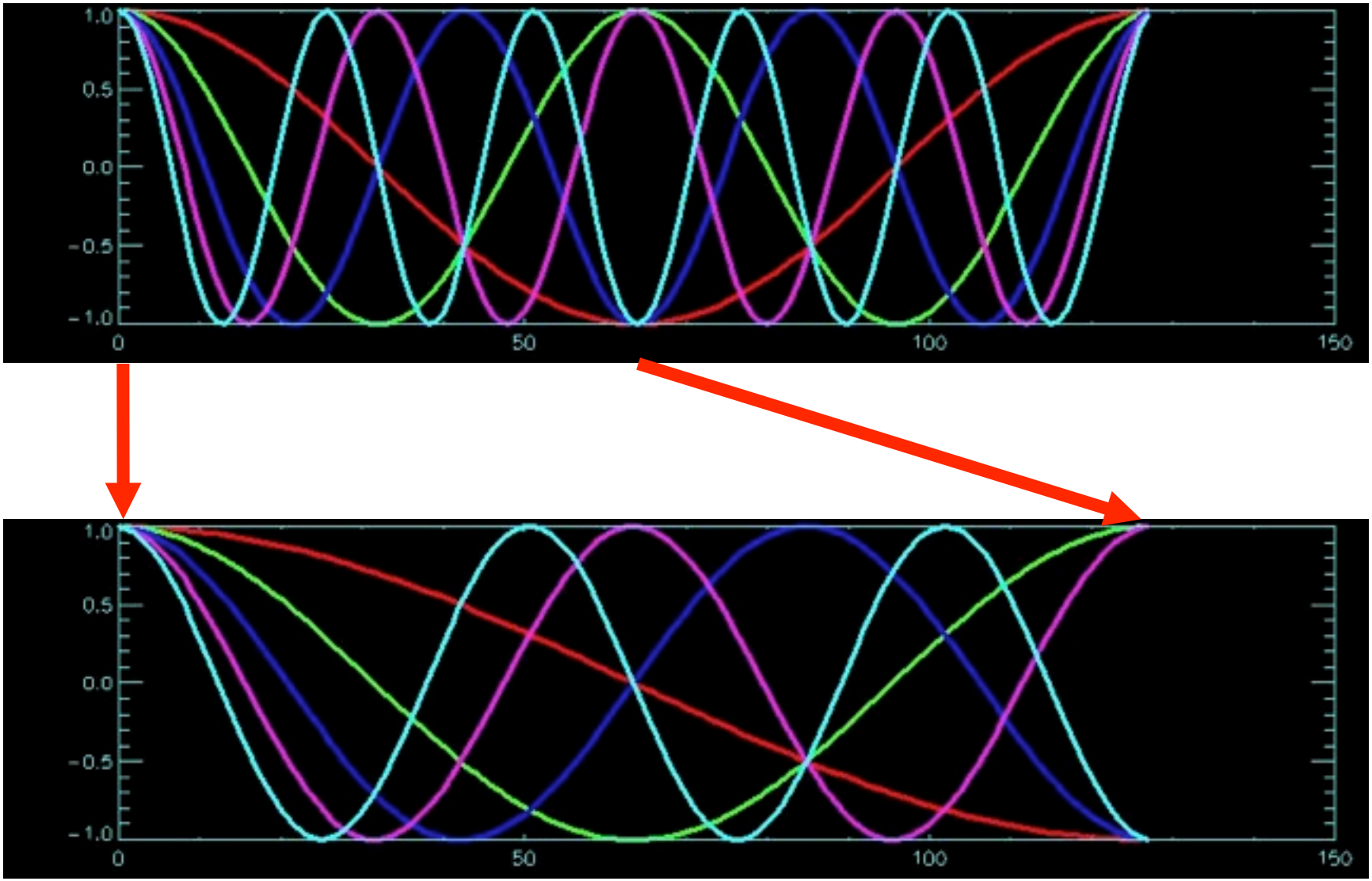
$$C(u,v) = \alpha(u)\alpha(v)\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)\cos\left(\frac{(2m+1)u\pi}{2M}\right)\cos\left(\frac{(2n+1)v\pi}{2N}\right)$$

$$f(m,n) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)C(u,v)\cos\left(\frac{(2u+1)m\pi}{2M}\right)\cos\left(\frac{(2v+1)n\pi}{2N}\right).$$

$$\alpha(u) = \begin{cases} \sqrt{1/M} & \text{, für } u = 0 \\ \sqrt{2/M} & \text{, für } u \neq 0 \end{cases} \quad \text{und} \quad \alpha(v) = \begin{cases} \sqrt{1/N} & \text{, für } v = 0 \\ \sqrt{2/N} & \text{, für } v \neq 0 \end{cases}$$

- Basisfunktionen sind Wellen, deren Wert für jede zweite Frequenz *u* am Anfang und am Ende des Intervalls unterschiedliches Vorzeichen haben.

Basisfunktionen der DCT



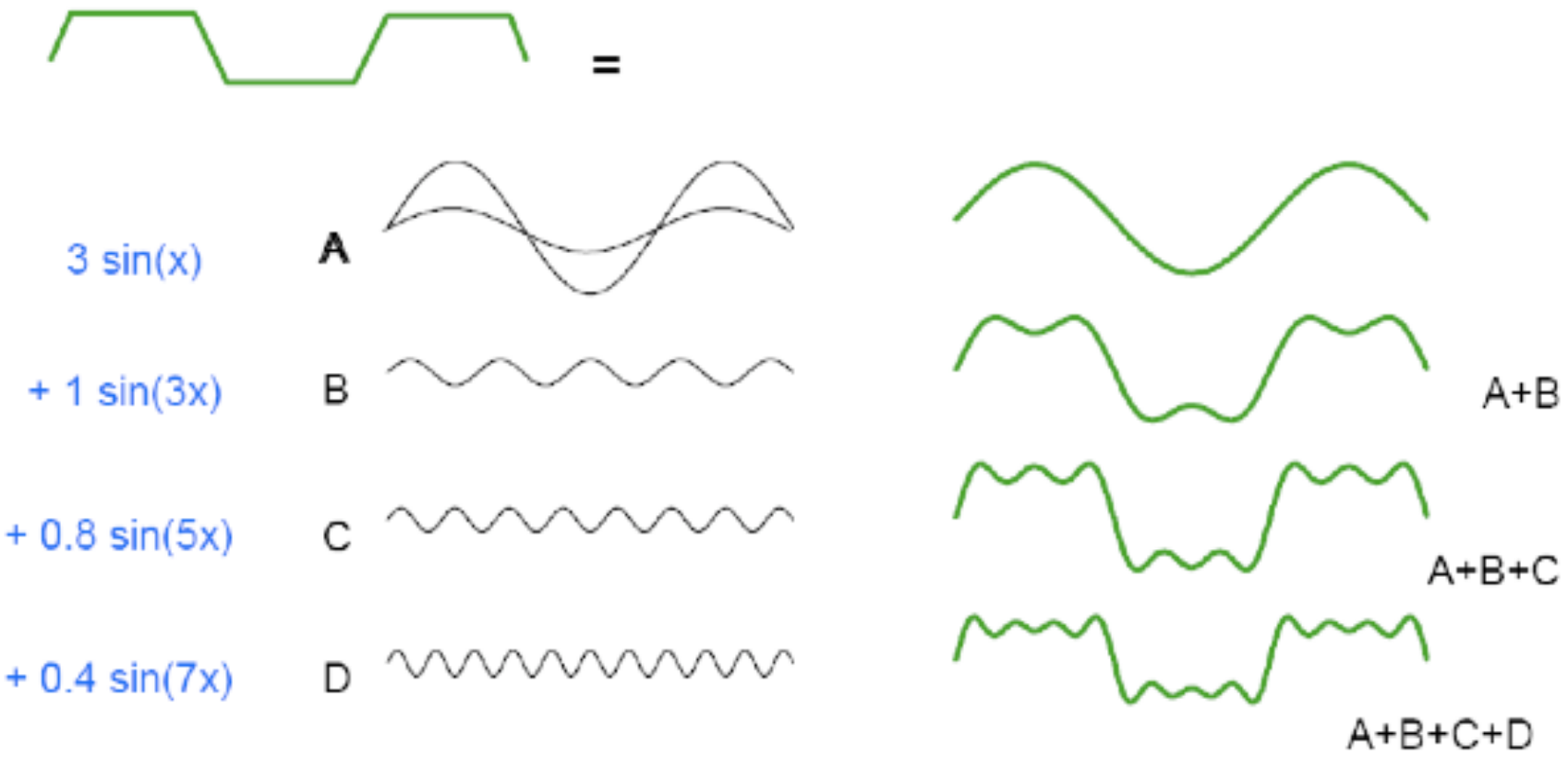
Literatur

- Klaus D. Tönnies: "Grundlagen der Bildverarbeitung", ISBN 3-8273-7155-4
- (daraus auch viele Abbildungen der heutigen Vorlesung)

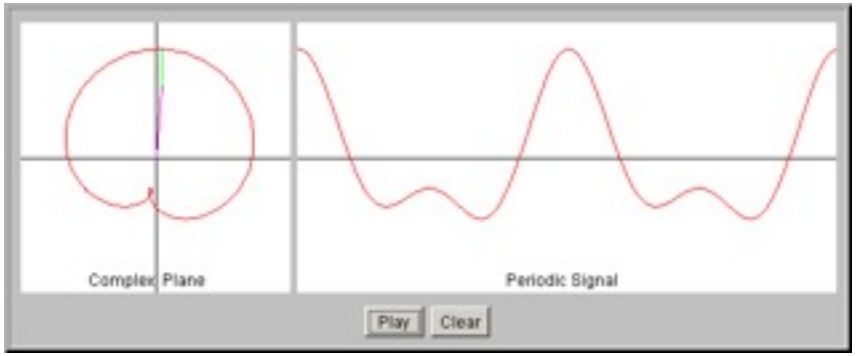
FFT und Anwendungen der Fouriertransformation

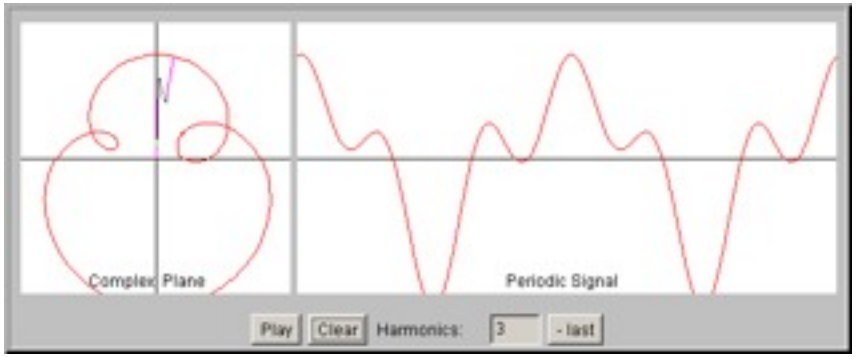
Fouriertransformation: Grundidee

- Beschreibe beliebige Funktion als gewichtete Summe periodischer Grundfunktionen (Basisfunktionen) mit untersch. Frequenz

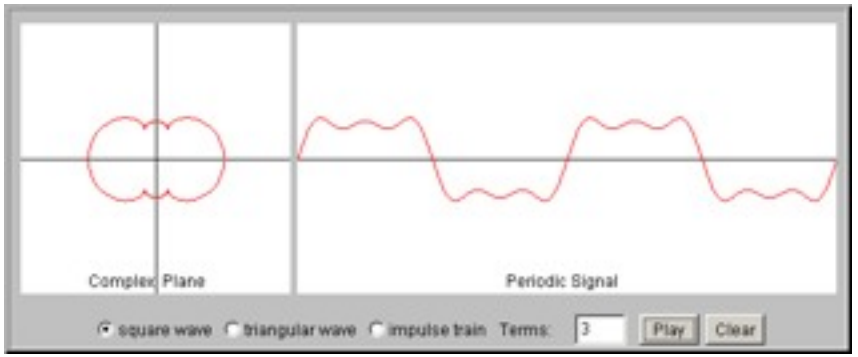


Überlagerung von Schwingungen: anschaulich



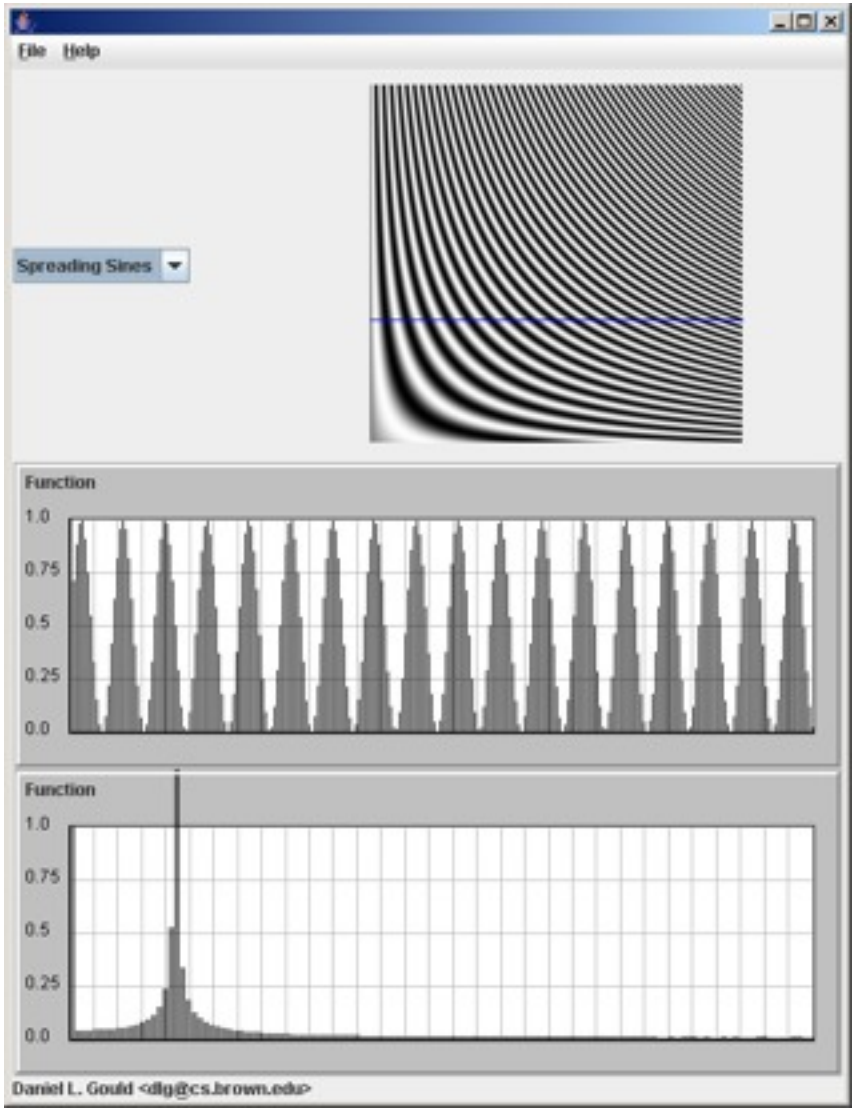


http://www.jhu.edu/~signals/index.html



Fourier Transformation zum Anschauen

- http://www.cs.brown.edu/exploratories/



Fast Fourier Transformation

Vorgehensweise generell

- Vereinfachende Annahme: $N=2^k$, $k>1$
- Nutze Separabilität, um 2D-FT auf 1D zurückzuführen ($O(n^4) \rightarrow O(n^3)$)
- Teile Summe in zwei Teilsummen auf
- Finde Gemeinsamkeiten in den Teilsummen und berechne beide Teilsummen miteinander
- Betrachte die Teilsumme und unterteile rekursiv bis N=1 ($O(n^3) \rightarrow O(n^2 \log n)$)

Separabilität

$$F(u,v) = \frac{1}{N^2}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}(um+vn)\right] =$$

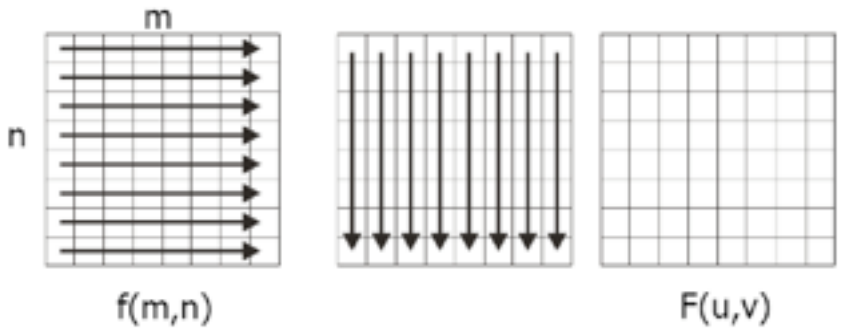
$$= \frac{1}{N^2}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}um\right]\exp\left[-i\frac{2\pi}{N}vn\right] =$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\left(\frac{1}{N}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}vn\right]\right)\exp\left[-i\frac{2\pi}{N}um\right] =$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\exp\left[-i\frac{2\pi}{N}um\right]\left(\frac{1}{N}\sum_{n=0}^{N-1} f(m,n)\exp\left[-i\frac{2\pi}{N}vn\right]\right) =$$

$$= \frac{1}{N}\sum_{m=0}^{N-1}\exp\left[-i\frac{2\pi}{N}um\right]F_u(m)$$

- Vorgehensweise: $F_u(m)$ für alle Spalten m berechnen und dann bei den Zeilen verwenden.



Divide Schritt

- Teile Summe in zwei Teilsummen auf

$$N = 2K, W_N = \exp\left(-i\frac{2\pi}{N}\right)$$

$$F(u) = \frac{1}{N}\sum_{n=0}^{N-1} f(n)(W_N)^{un} = \frac{1}{2K}\sum_{n=0}^{2K-1} f(n)(W_{2K})^{un} =$$

$$= \frac{1}{2}\left(\frac{1}{K}\sum_{n=0}^{K-1} f(2n)(W_{2K})^{2nu} + \frac{1}{K}\sum_{n=0}^{K-1} f(2n+1)(W_{2K})^{(2n+1)u}\right)$$

- Finde Gemeinsamkeiten in den Teilsummen

$$F_{even}(u) = \frac{1}{K}\sum_{n=0}^{K-1} f(2n)(W_{2K})^{2nu}, F_{odd}(u) = \frac{1}{K}\sum_{n=0}^{K-1} f(2n+1)(W_{2K})^{2nu}$$

$$F(u) = \frac{1}{2}\left(F_{even}(u) + F_{odd}(u)(W_{2K})^{u}\right)$$

Ausnutzen der Periodizität

$$(W_K)^{u+N} = (W_K)^u, (W_{2K})^{u+K} = -(W_{2K})^u$$

$$F(u+K) = \frac{1}{2}\left(F_{even}(u+K) + F_{odd}(u+K)(W_{2K})^u\right) =$$

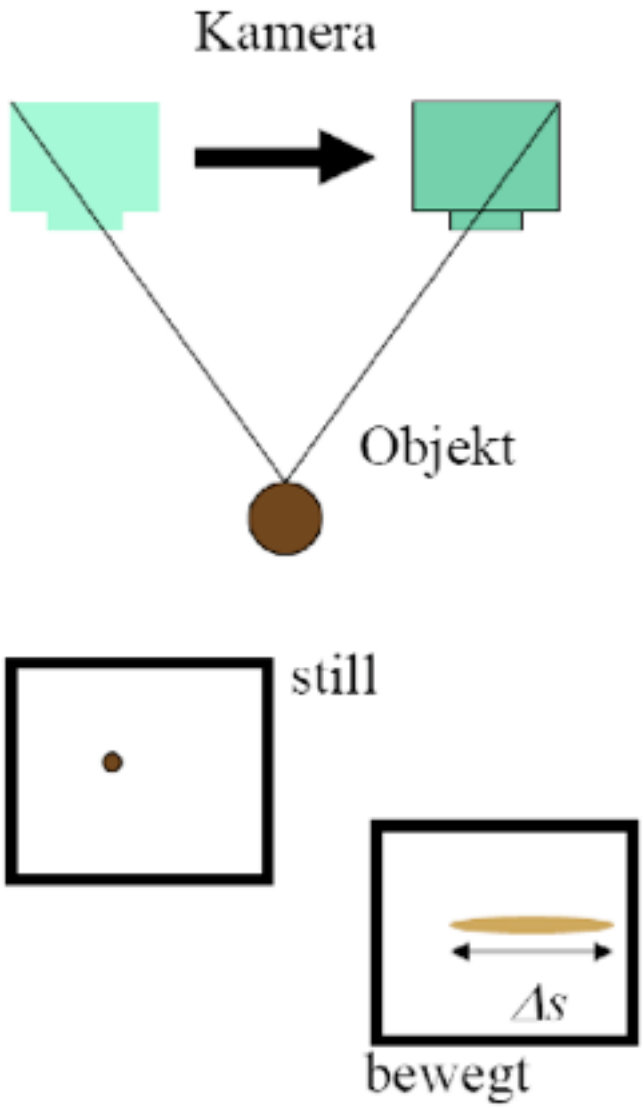
$$= \frac{1}{2}\left(F_{even}(u) - F_{odd}(u)(W_{2K})^u\right)$$

- Also kann man F(u+K) mithilfe F(u) berechnen (einmal $F_{even} + F_{odd}$, einmal $F_{even} - F_{odd}$)
- Betrachte die Teilsumme [0…K-1] und unterteile rekursiv bis K=1 ($O(n^3) \rightarrow O(n^2 \log n)$)

Restauration linearer Bildstörungen

Beispiel I: Bewegungsunschärfe

- Über einen Zeitraum Δ*t* wird ein Objektpunkt *p* auf immer andere Punkte auf dem CCD-Chip abgebildet.
- Bei unbewegter Kamera sei die Bildhelligkeit des abgebildeten Punkts *h*.
- Dann ist sie bei bewegter Kamera *h*/Δ*s*, wobei Δ*s* die zurückgelegte Strecke ist.
- Wenn Δ*s* für alle Punkte gleich ist, dann lässt sich die Veränderung durch eine Faltung beschreiben.



Bewegungsunschärfe

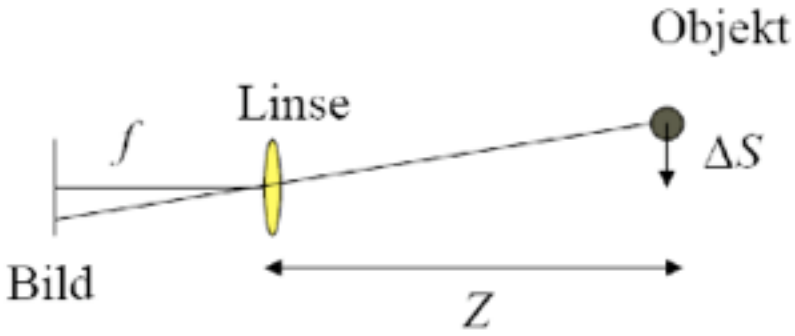
- Faltungskern ist eine Funktion *w* mit

$$w(t \cdot \cos\alpha, t \cdot \sin\alpha) = \begin{cases} \frac{1}{\Delta s} & \left|\frac{t}{2}\right| < \Delta s \\ 0 & sonst \end{cases}$$

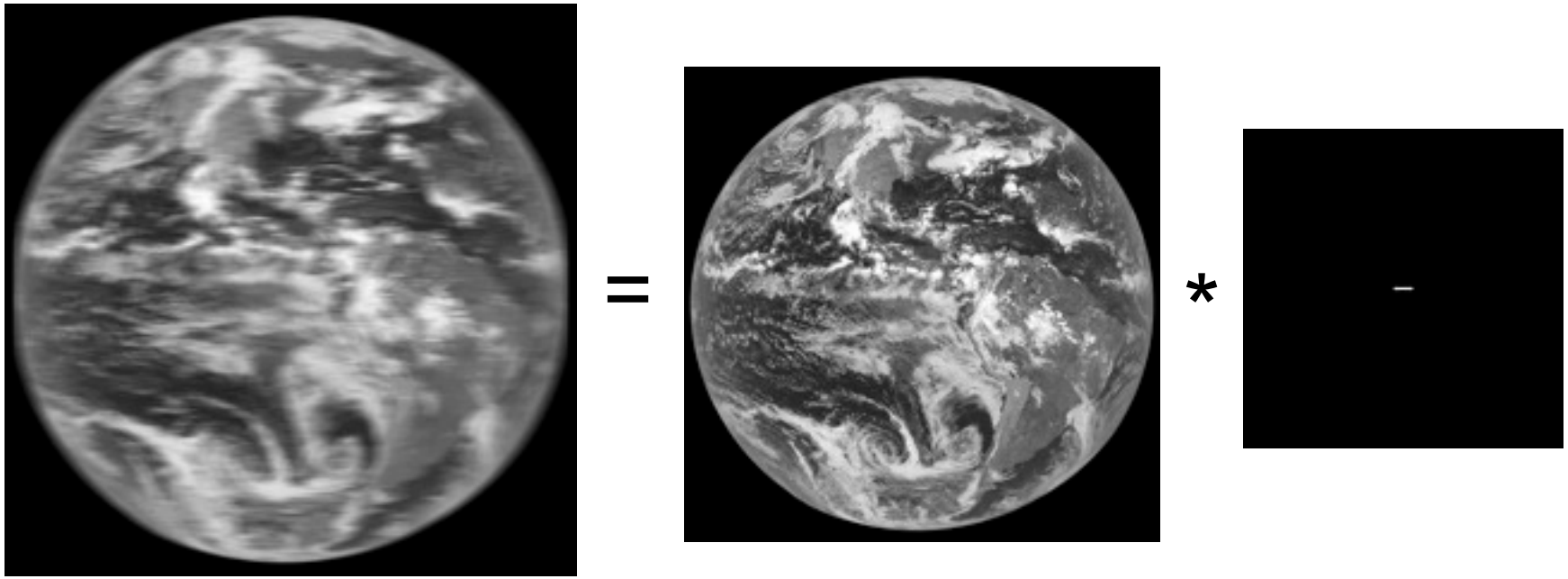
- Der Winkel α gibt die Bewegungsrichtung an.
- Die Strecke Δ*s* gibt die Strecke an, um die sich der Punkt bewegt hat:

$$\Delta s = \frac{f}{Z} \cdot \frac{\Delta S}{p}$$

- *f*: Brennweite
- *Z*: Objektabstand
- Δ*S*: Bewegung in der *X*-*Y*-Ebene
- *P*: Pixelgröße



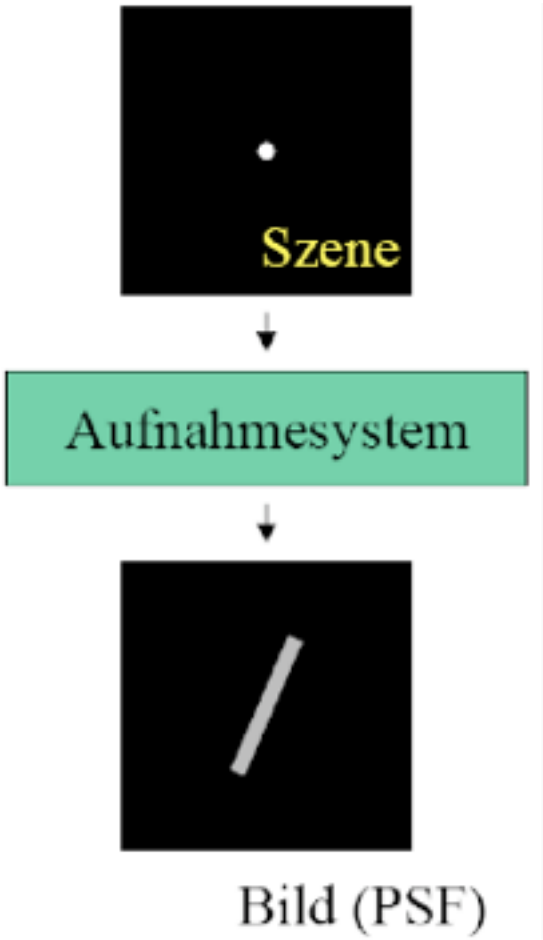
Bewegungsunschärfe



Gestörtes Bild | Original | Faltungskern

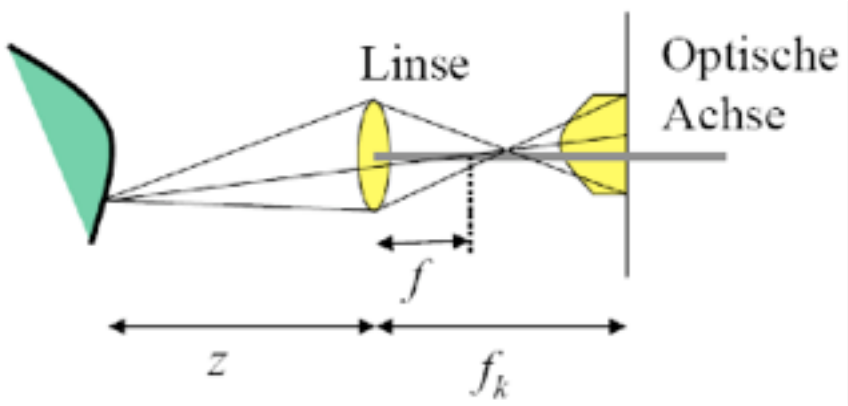
Repräsentation linearer Störungen

- Jede verschiebungsinvariante lineare Operation wird vollständig durch die Faltungsfunktion beschrieben.
- Die Faltungsfunktion beschreibt die Operation für beliebige Bilder
- Die Faltungsfunktion kann als Resultat der Veränderung eines Punkts erzeugt werden
- Punktantwort = Point Spread Function (PSF)



Beispiel II: Fokussierungsunschärfe

- Maß der Unschärfe hängt vom Punktabstand *z*, der Brennweite der Linse *f* und der Kammerkonstante f_k ab.



- Linsengesetz:

$$\frac{1}{z} = \frac{1}{f} + \frac{1}{f_k} \Leftrightarrow f_k = \frac{zf}{f - z}$$

- Größe des Unschärfekreises:

$$\frac{d}{f} = \frac{s}{f_k} \Leftrightarrow s = \frac{f_k d}{f} = \frac{zd}{f - z}$$

- Unschärfe kann durch Aufnahme eines punktförmigen Testobjekts angenähert werden.



Fokussierungsunschärfe





Wie kann die Störung rückgängig gemacht werden?

Bildrestauration

- Ziel: Korrektur des Bildsignals um bekannte und unbekannte Störungen
- Annahme: Störung kann durch einen verschiebungsinvarianten linearen Operator h beschrieben werden.

- $g(x,y)$ → (Störung durch PSF h) → $g'(x,y)$

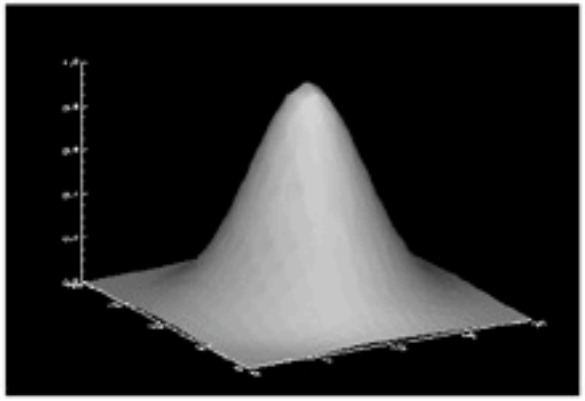
- $g'(x,y) = [h*g]\,(x,y)$

- PSF beschreibt die Störung → **Wie kann die PSF bestimmt werden?**

Gesucht: PSF

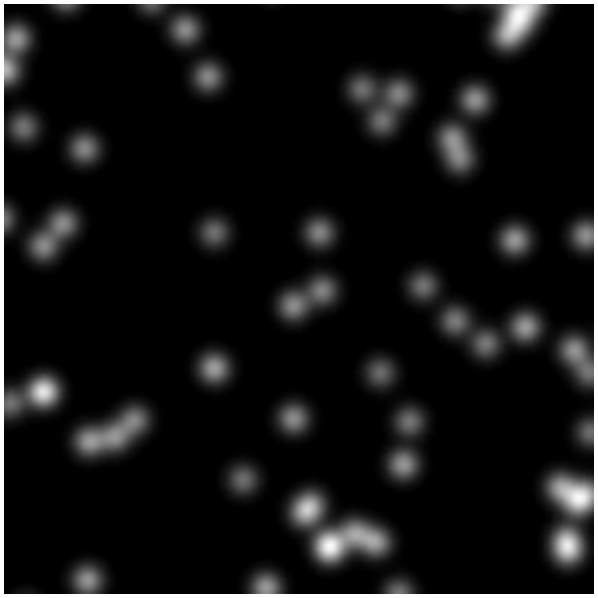






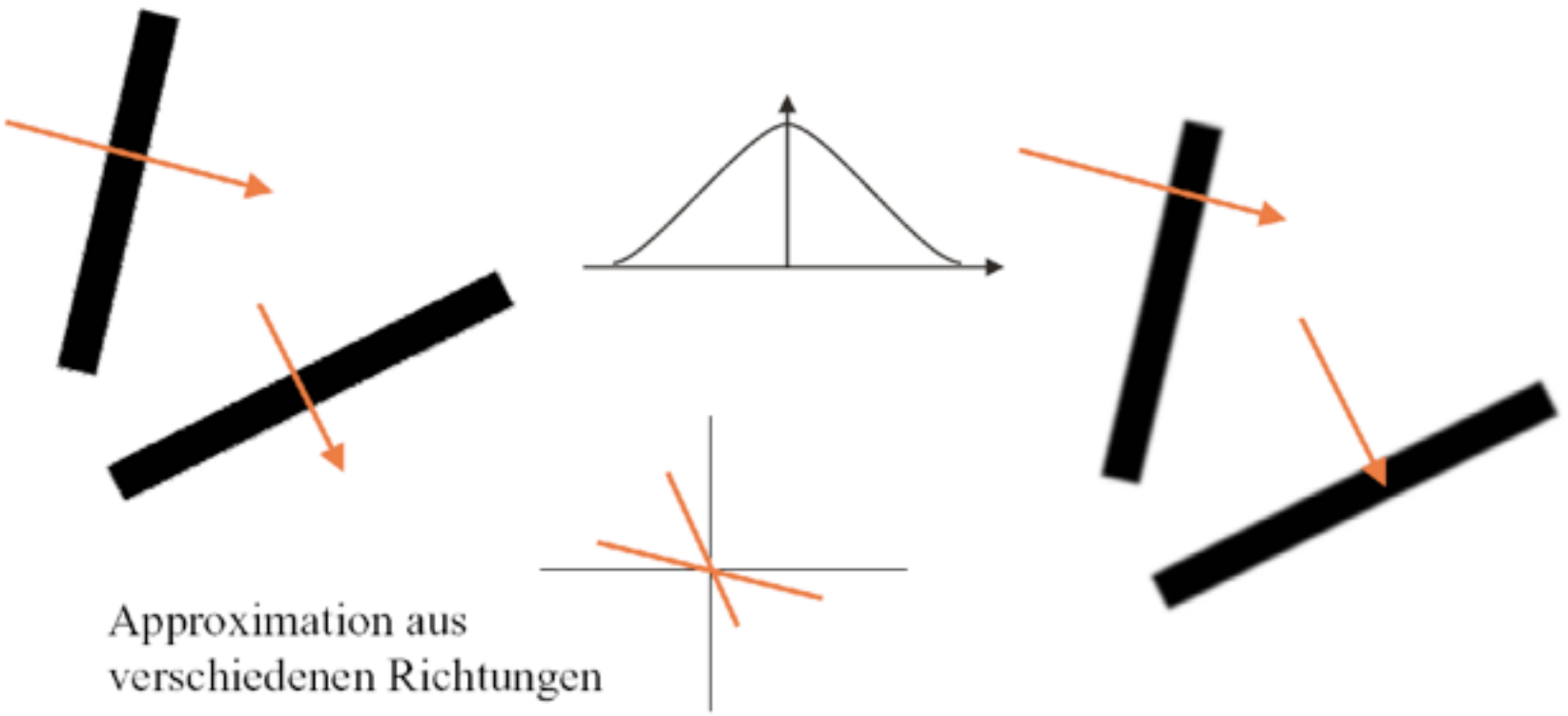
PSF von Testbildern





- Annahme: Störung ist unveränderlich und Testaufnahme ist möglich.
- Durch die Aufnahme eines punktförmigen Objekts kann ein δ-Impuls approximiert werden.
- Aufnahme ist eine Näherung für die PSF.

PSF aus dem aufgenommenen Bild



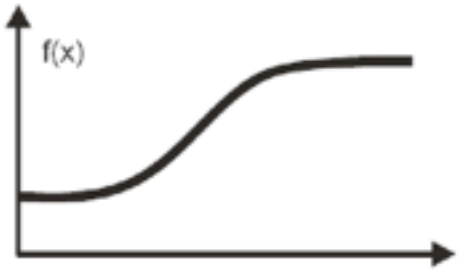
- Testaufnahme ist nicht möglich: Näherungsweise Bestimmung der PSF durch Betrachtung von Punkten oder Linien im gestörten Bild.

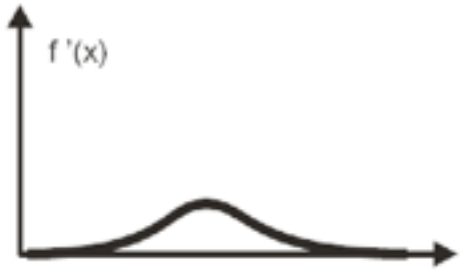
Kanten

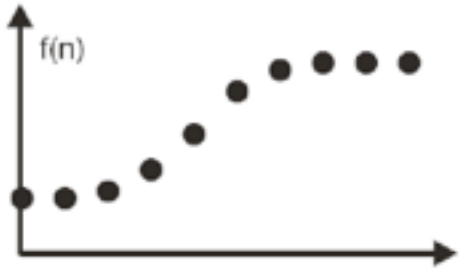
- Die meisten Bilder weisen wenige Linien oder Punkte auf, aber Kanten können in fast jedem Bild gefunden werden.

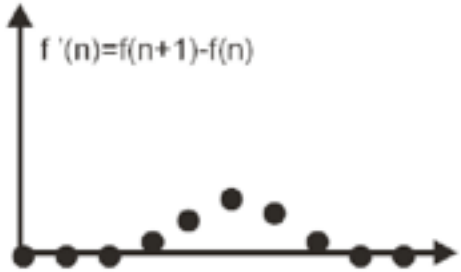


1D-Kanten









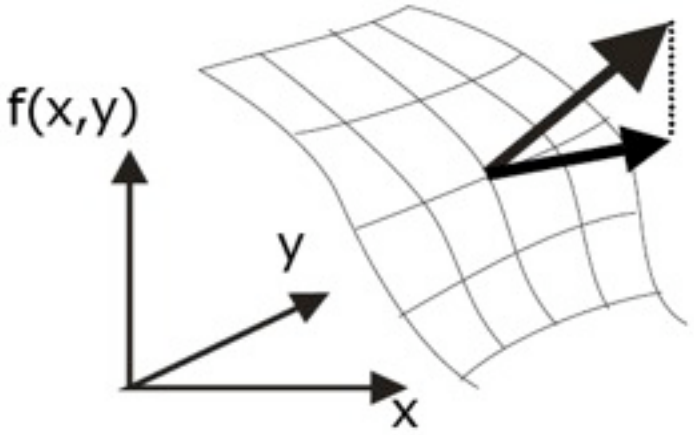
Die Stärke einer Kante hängt von der Steigung der Funktion ab:

Betrag der ersten Ableitung bestimmen.

Für diskrete Funktionen:

Ableitung wird durch Differenz angenähert

Kanten im 2D-Raum: Gradienten



Richtung: Richtung der größten Steigung.

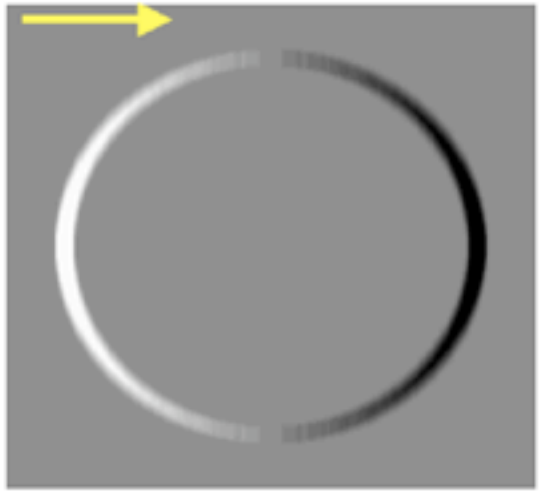
Länge: Stärke der stärksten Steigung.

- Gradient im kontinuierlichen Raum (*x*,*y*): Vektor der partiellen Ableitungen der Bildfunktion in *x*- und *y*-Richtung: (*f*(*x*,*y*)) = (*∂f*/*∂x* *∂f*/*∂y*)
- Approximation des Gradienten: Differential wird durch Differenz approximiert:

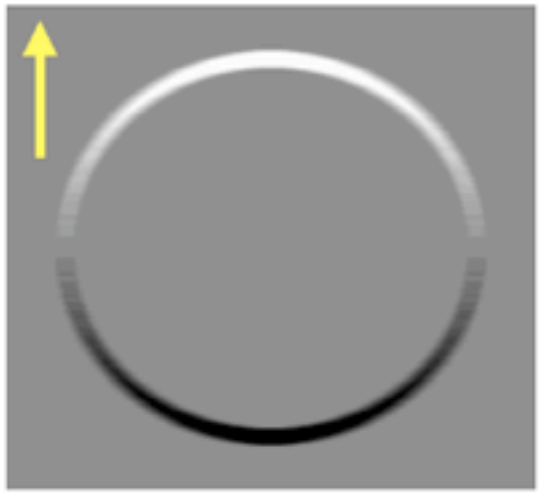
$$\vec{G}(f)(m,n) \approx [G_x(m,n) \quad G_y(m,n)] = [f(m,n)\text{-}f(m\text{-}1,n) \quad f(m,n)\text{-}f(m,n\text{-}1)]$$

Elemente des Gradienten





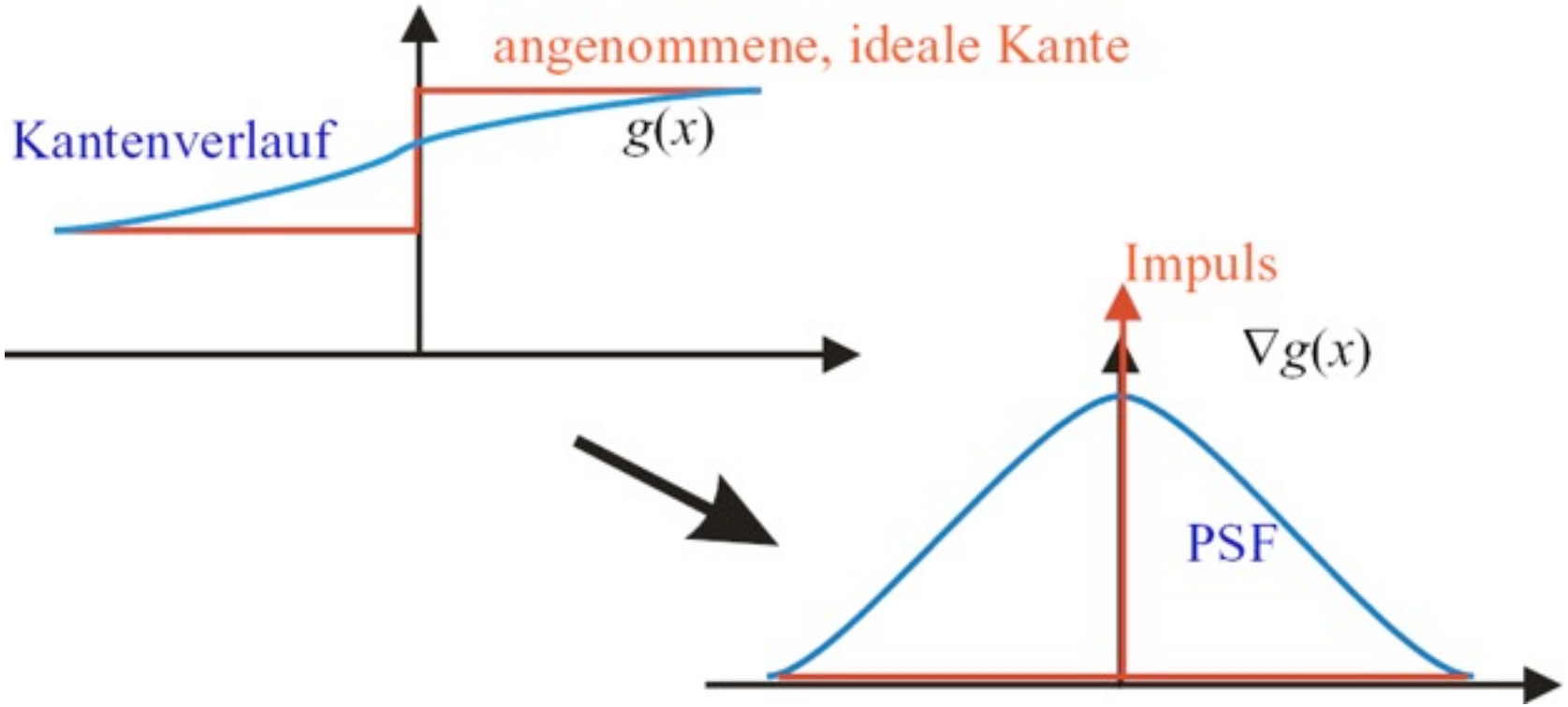






Betrag : $\text{sqrt}(G_x^2+G_y^2)$
Richtung: $\tan^{-1}(G_y/G_x)$

PSF aus Kanten

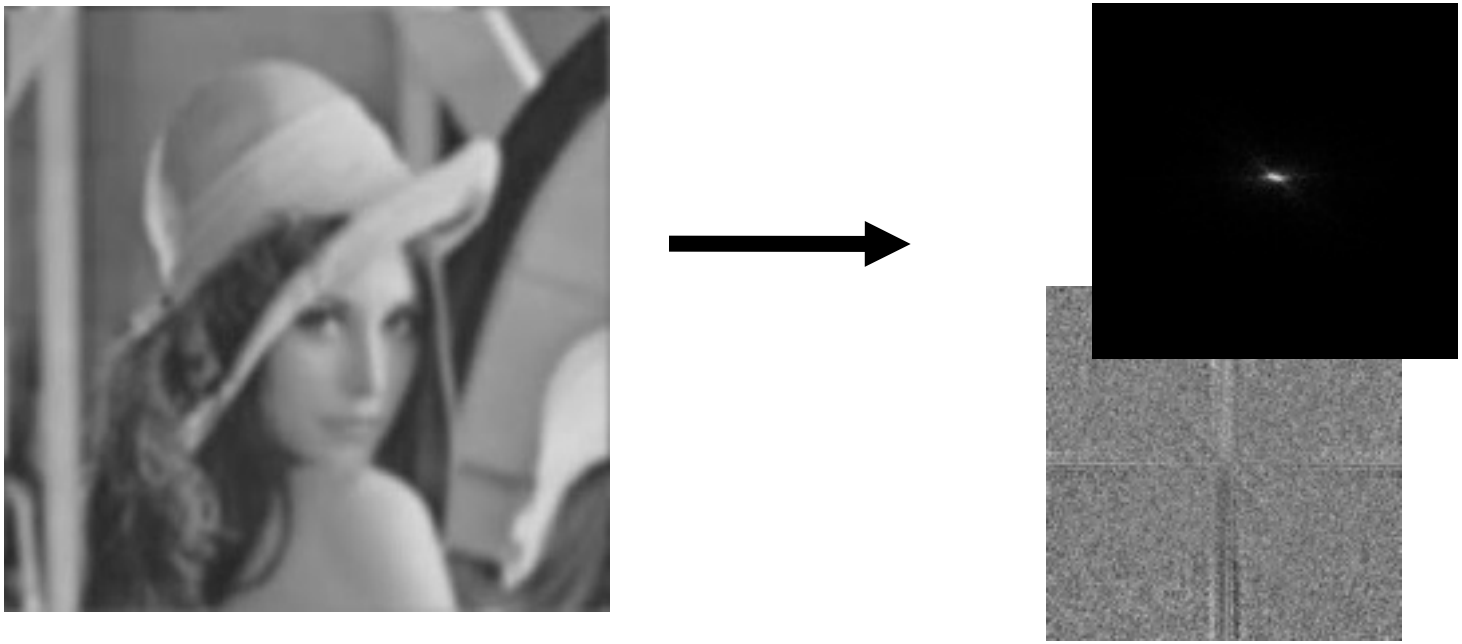


- PSF kann auch aus dem Verlauf einer als ideal angenommenen Kante approximiert werden.

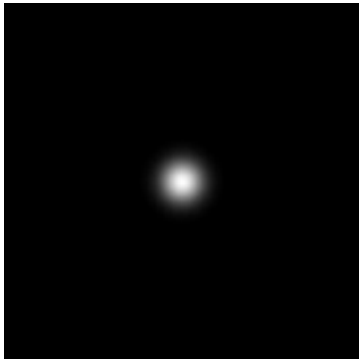
Invertierung der Störung

- Überführung der Repräsentation in den Frequenzraum:
- $G'(u,v) = \mathbf{FT}[g'(m,n)] = \mathbf{FT}[[h*g](m,n)] = H(u,v)\cdot G(u,v)$

- Invertierung:
- $g(m,n) = \mathbf{FT}^{-1}[G'(u,v)/H(u,v)]$ (**Inverse Filterung**)



Inverse Filterung

$FT^{-1}($  /  $)=$ 

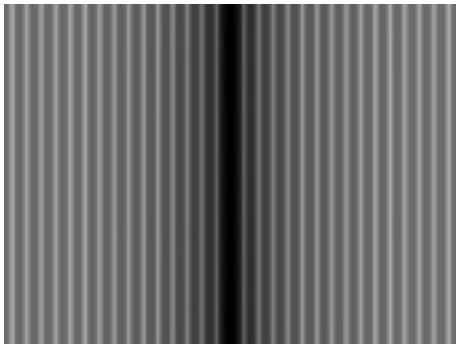
- Theoretisch: vollständige Rückgewinnung der Information aus den gestörten Daten

Bewegungsunschärfe





PSF



FT(PSF)

Bewegungsunschärfe



Resultat der inversen Filterung $\mathbf{FT}^{-1}$ [**FT** (g‘)(u,v)/**FT**(PSF)(u,v)]

Numerische Probleme bei der inversen Filterung

$$g = h * f \Rightarrow f(m,n) = FT^{-1}\left(\frac{G(u,v)}{H(u,v)}\right)$$

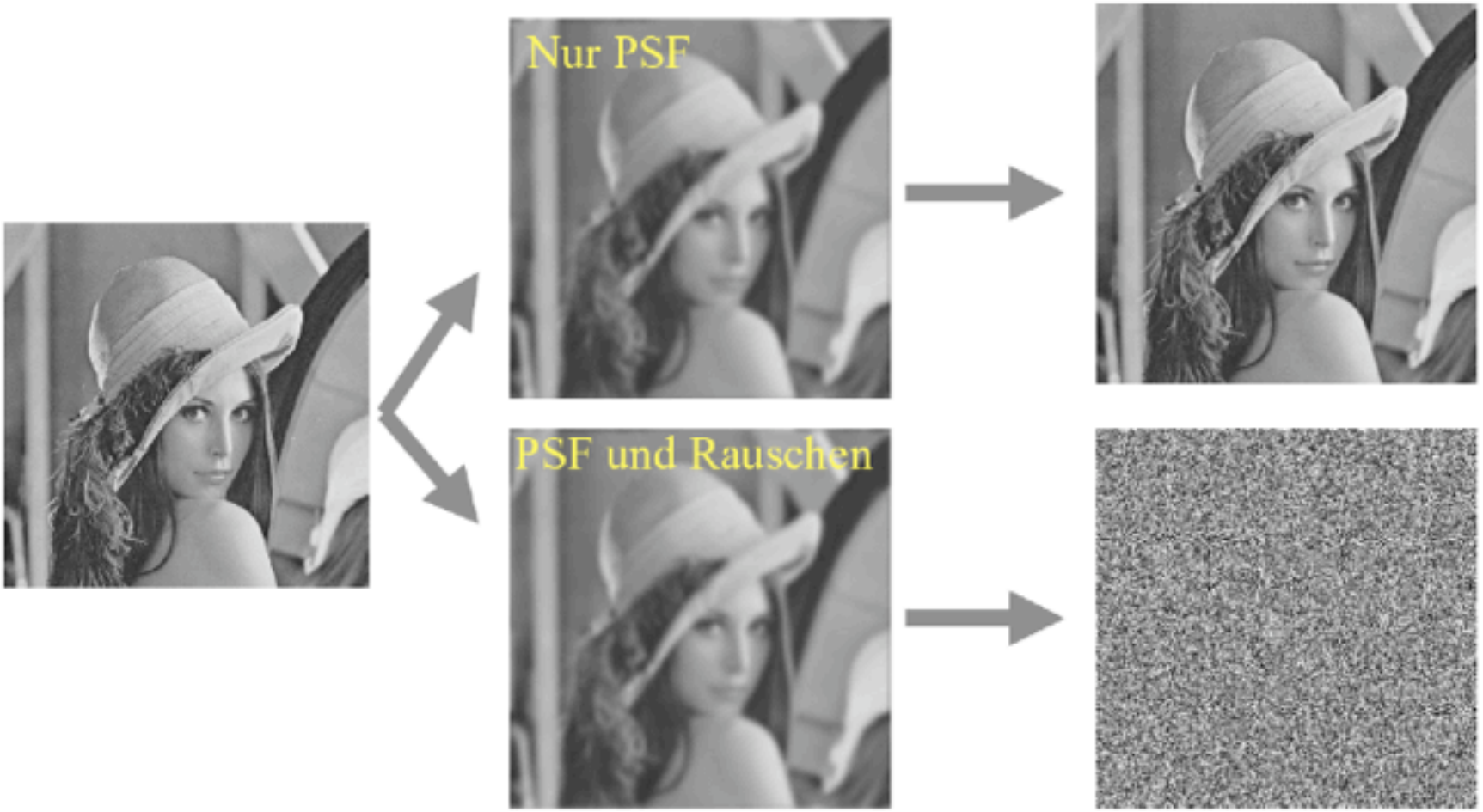
- Problem: Nullstellen von H
  - Treten auf, falls h als Matrix nicht den vollen Rang hat
  - Auch kleine Werte von H sind numerisch schon ein Problem
- Deswegen in der Praxis:

$$F(u,v) = \begin{cases} \frac{G(u,v)}{H(u,v)} & H(u,v) > H_{\min} \\ 0 & sonst \end{cases}$$

Rauschen

- Problem: Inverse Filterung geht von idealen (ungestörten) Daten aus
- In Wirklichkeit enthalten Bilddaten Rauschen
- Dieses Rauschen wird bei der inversen Filterung extrem verstärkt
- Rauschen lässt sich nicht „herausrechnen“, da es nicht wiederholbar ist

Rauschen



Ad-hoc Lösung

- Gewichte die Wirkung der inversen Filterung mit der Amplitude der Störungsfunktion im Verhältnis zur mittleren Amplitude

$$\frac{1}{H(u,v)} \frac{|H(u,v)|}{A_N}$$

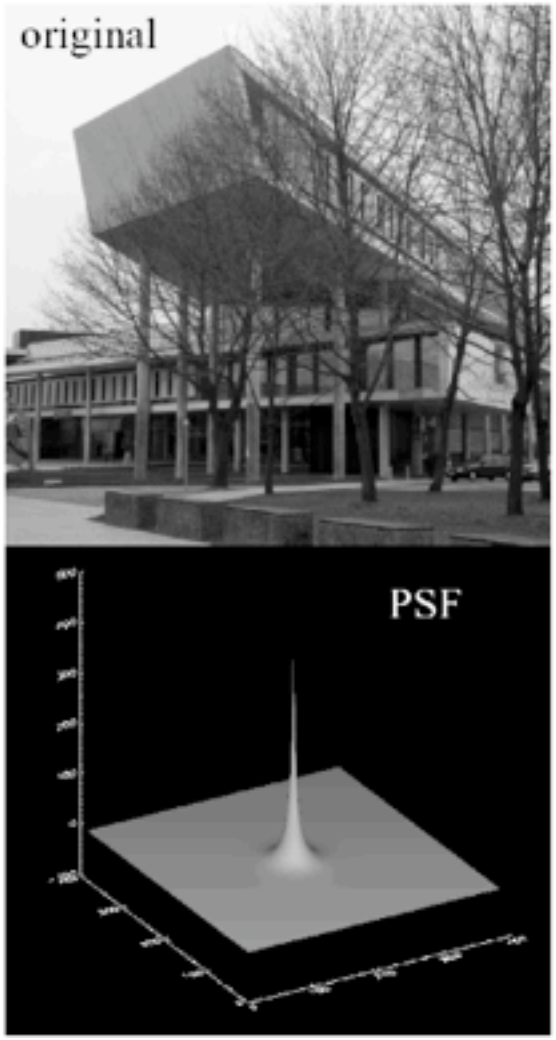
- Problem: Gewichtung nimmt keine Rücksicht auf die Signalstärke von F

Wiener Filter

$$\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+\left(S_\eta(u,v)/S_f(u,v)\right)^{\frac{1}{j}}} \; oder \; \frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+k}$$

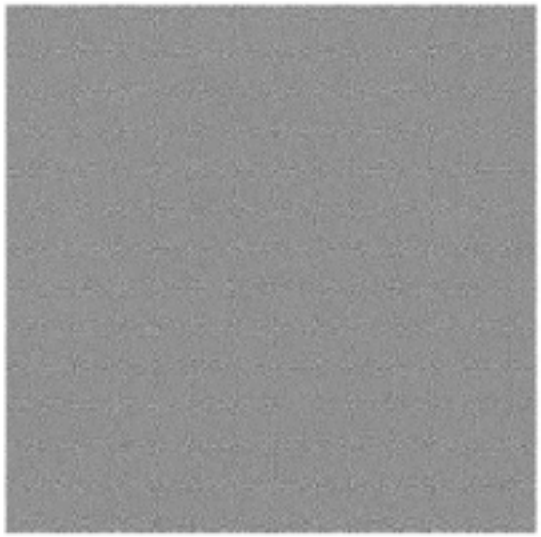
- Wobei S_η und S_f die Spektren des Rauschens bzw. der ungestörten Funktion sind
  - S_η =0 (ungestört) ➔ perfekte inverse Filterung
- Leider ist S_η in der Praxis meist unbekannt
- Lösung: heuristisches Wiener Filter mit k

Beispiel





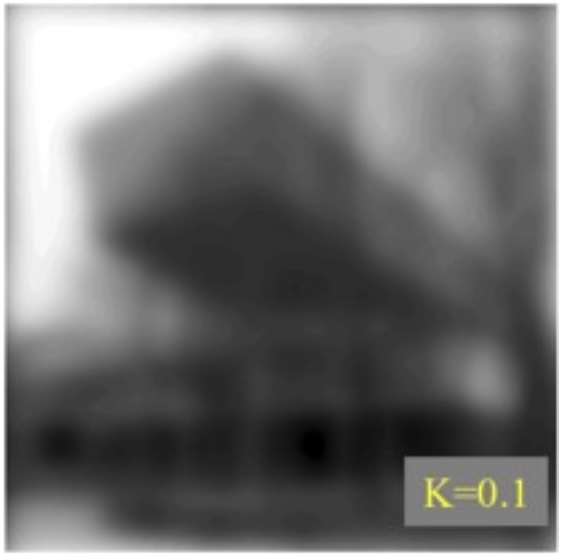
gestörtes Bild

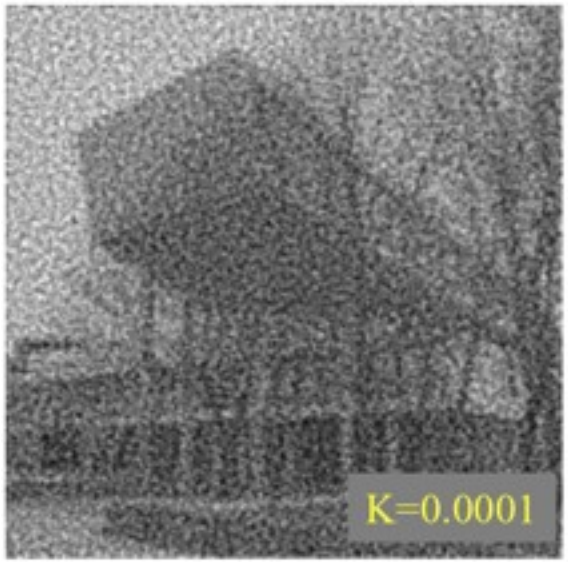


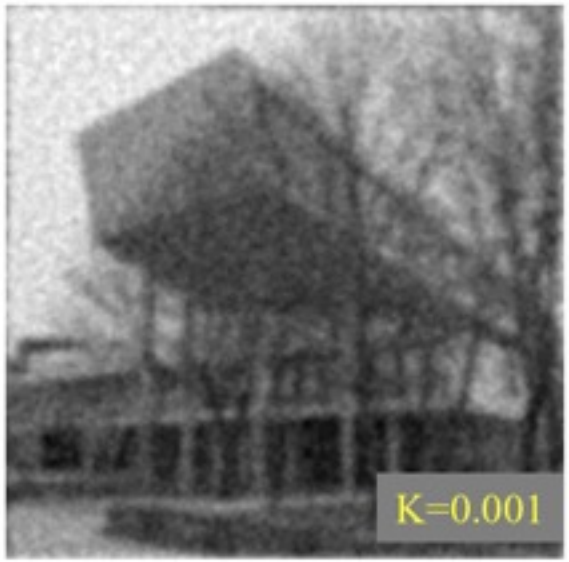
durch Inverse Filterung „restauriert“

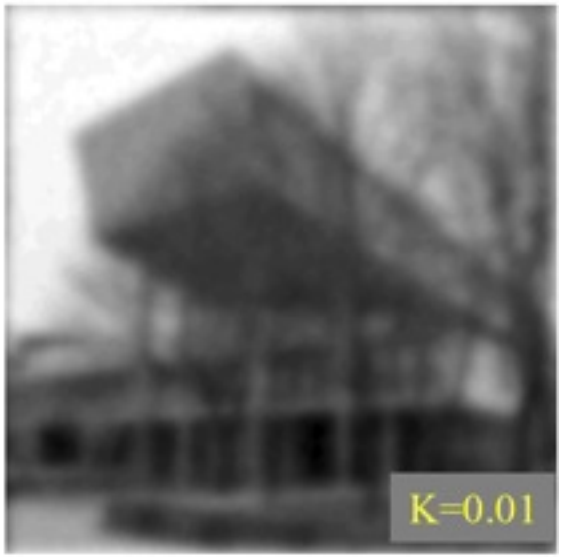
(Heuristisches) Wiener Filter











Transformation und Interpolation

Transformation und Interpolation

- Die Transformationen Translation, Rotation und Skalierung sind auf reellen Zahlen definiert:

$$Rot_\alpha \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}, \quad Tr_{dx,dy} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x+dx \\ y+dy \end{pmatrix}, \quad Sc_s \begin{pmatrix} x \\ y \end{pmatrix} = s \begin{pmatrix} x \\ y \end{pmatrix}$$

- Digitale Bilder haben einen ganzzahligen Definitionsbereich.
- Nach Transformation ist eine Interpolation notwendig.

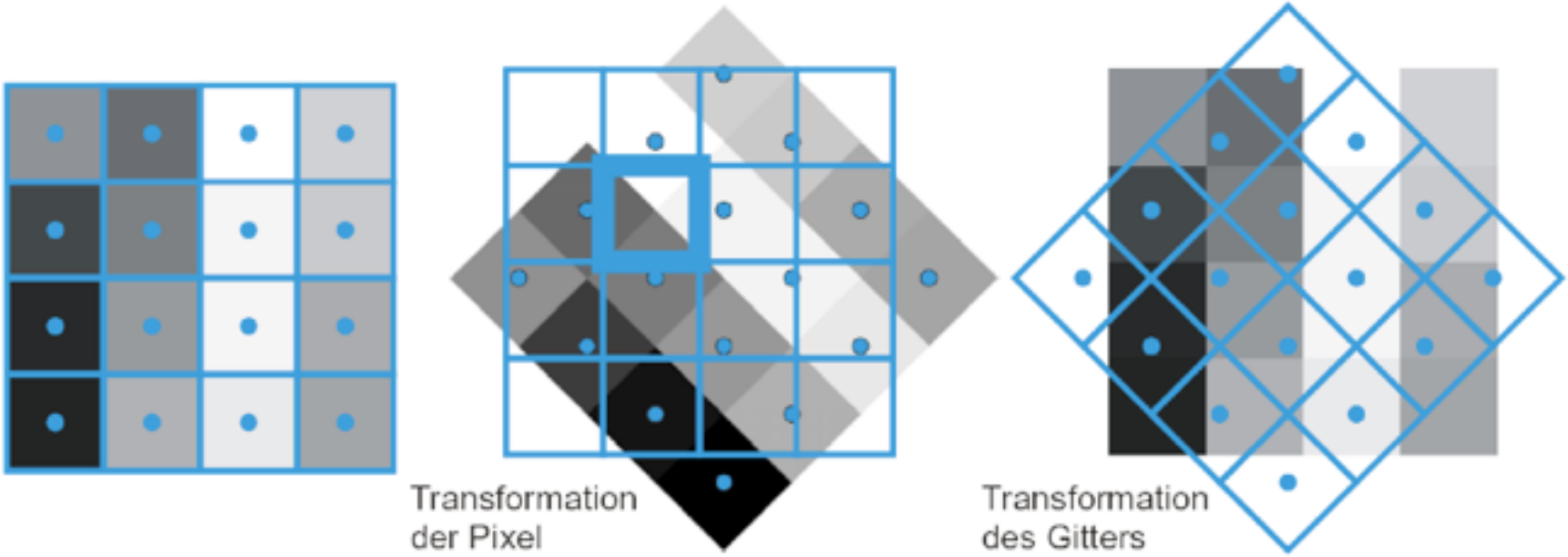
Interpolation

- Konstante Interpolation (Wert des nächsten Nachbarpixels)
- Lineare Interpolation
- Interpolation durch Polynome höheren Grades.
- Interpolation im Frequenzraum.

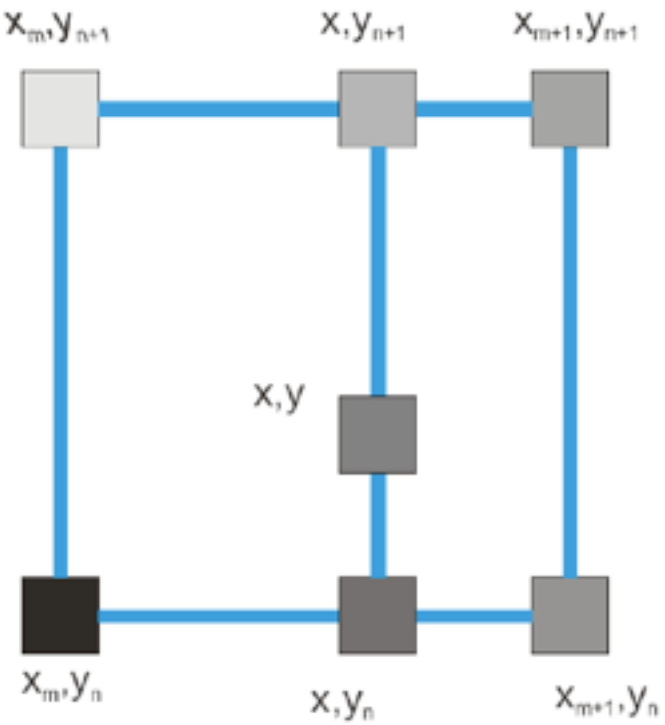




Konstante Interpolation



Bilineare Interpolation



Erster Schritt:

$$g_1(x_m, y) = \frac{y_{n+1} - y}{y_{n+1} - y_n} f(x_m, y_n) + \frac{y - y_n}{y_{n+1} - y_n} f(x_m, y_{n+1}),$$

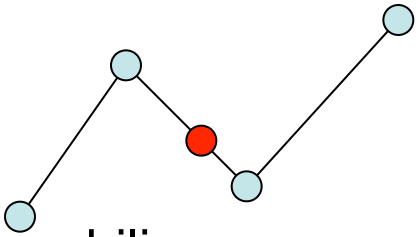
$$g_1(x_{m+1}, y) = \frac{y_{n+1} - y}{y_{n+1} - y_n} f(x_{m+1}, y_n) + \frac{y - y_n}{y_{n+1} - y_n} f(x_{m+1}, y_{n+1}).$$

Zweiter Schritt:

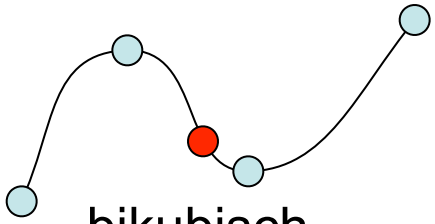
$$g(x, y) = \frac{x_{m+1} - x}{x_{m+1} - x_m} g_1(x_m, y) + \frac{x - x_m}{x_{m+1} - x_m} g_1(x_{m+1}, y).$$

Polynome höheren Grades

- Interpolation der Bildfunktion durch mehr als 2 Stützpunkte
- Polynom n-ten Grades interpoliert n+1 Punkte
- Die Bildfunktion wird besser angenähert, wenn mehr Terme der Taylor-Approximation berücksichtigt werden.
- Ableitungen für die Taylor-Reihe werden durch Differenzen angenähert.
- Grad des Polynoms ist ein Kompromiss zwischen
  - steigender Anzahl berücksichtigter Terme der Taylor-Reihe.
  - steigender Ungenauigkeit der geschätzten Ableitungen.



bilinear



bikubisch

Interpolation im Frequenzraum

- Die Basisfunktionen der Fouriertransformation haben einen reellen Definitionsbereich.
- Ein Funktionswert kann an beliebiger Stelle (x,y) bestimmt werden durch

$$g_F(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) \exp\left(i2\pi \left(\frac{ux}{M} + \frac{vy}{N} \right) \right)$$

- Die Interpolation ist exakt, falls die ursprüngliche Funktion f bandbegrenzt ist (was die meisten Funktionen jedoch nicht sind).

Literatur

- http://www.jhu.edu/~signals/index.html
- http://www.cs.brown.edu/exploratories/

- Klaus D. Tönnies: "Grundlagen der Bildverarbeitung", ISBN 3-8273-7155-4
- http://www.pearson-studium.de/main/main.asp?page=booksites/selectchapter&isbn=3827371554&PSZielgruppe=Student

Computergraphik 1
2. Teil: Bildverarbeitung

Bildverbesserung

Themen jetzt gleich

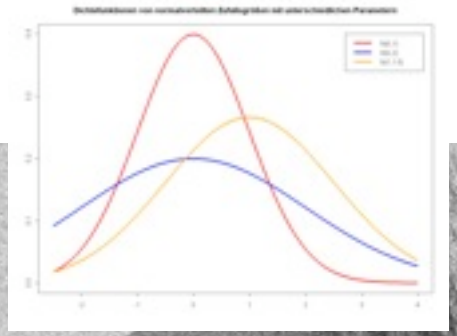
- Rauschen, Entropie
- Bildverbesserung
  - Punktbasiert
  - Flächenbasiert
  - Kantenbasiert

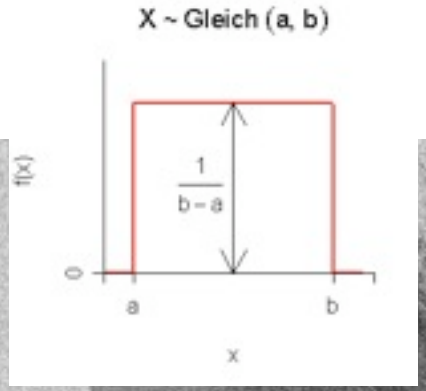
Was ist Rauschen?

- Rauschen $n(m,n)$ ist eine nicht-wiederholbare Veränderung der Bildfunktion.
- Unkorreliertes Rauschen:

  Additiv – $g(m,n)=f(m,n)+n(m,n)$
- Ursache: z.B. Quantenrauschen
- Charakterisierung von n:
  - Beschreibbar über eine Verteilungsfunktion:
    - Gleichverteilung
    - Normalverteilung
  - Erwartungswert E(n)=0

Normalverteilung vs. Gleichverteilung





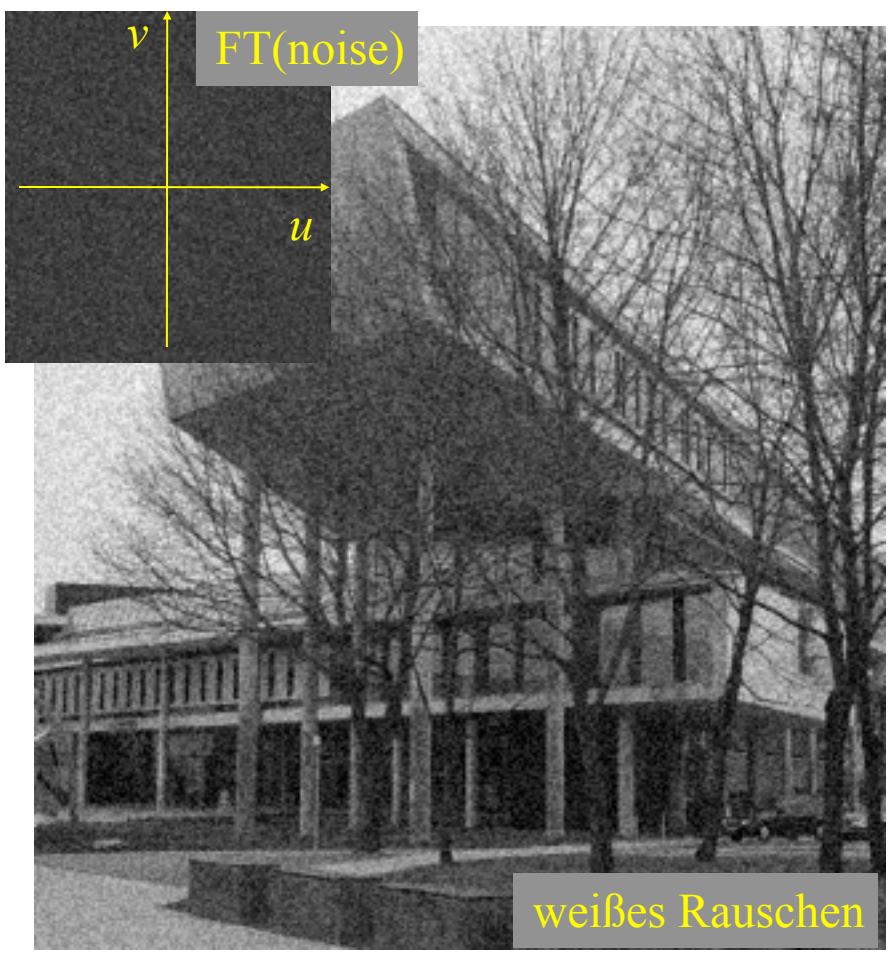
Normalverteilt, SNR=6.5:1

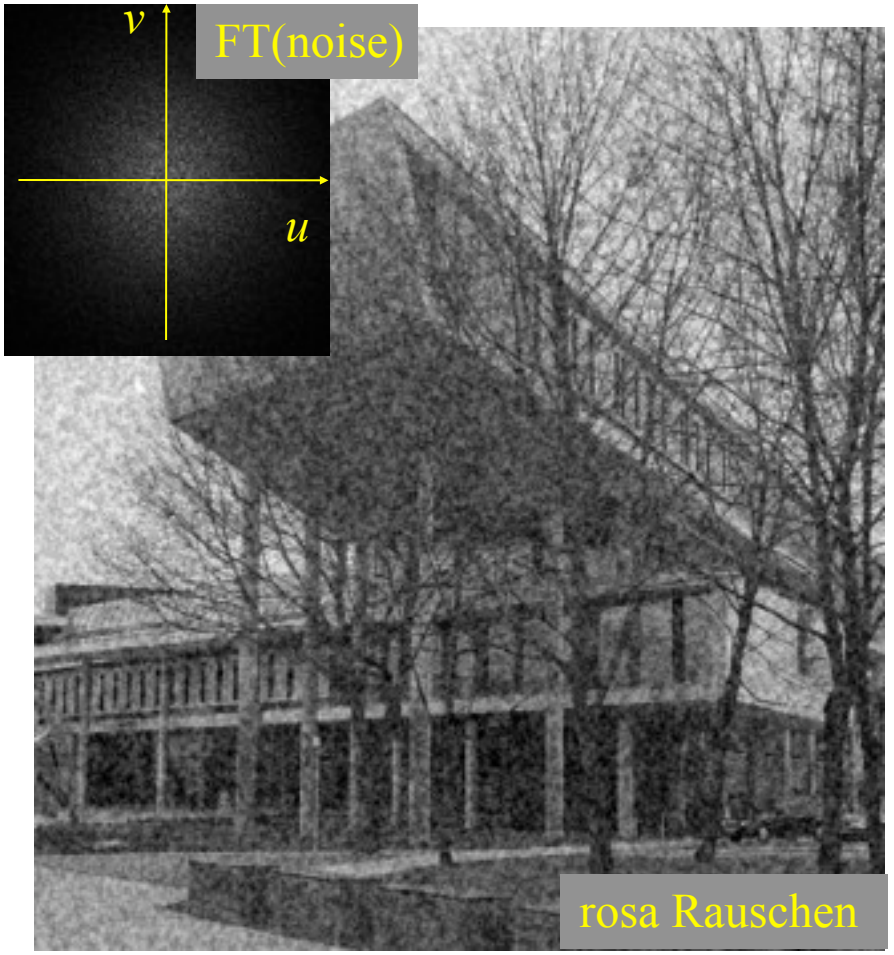
Gleichverteilt, SNR=6.5:1

Beschreibung von Rauschen

- Charakterisierung im Frequenzbereich, z.B.
  - Weißes Rauschen: Alle Frequenzen mit gleicher Amplitude.
  - Rosa Rauschen: Amplitude im niederfrequenten Bereich höher.
  - Farbiges Rauschen: Variierende Amplitude.
- Signal-Rausch-Verhältnis (SNR: Signal-to-Noise-Ratio)
  - $SNR_{max} = max_{m,n}[f(m,n)] / avg[n(m,n)]$
  - $SNR_{\varnothing} = avg[f(m,n)] / avg[n(m,n)]$.
  - SNR kann auch objektabhängig bestimmt werden

Weißes vs. Rosa Rauschen

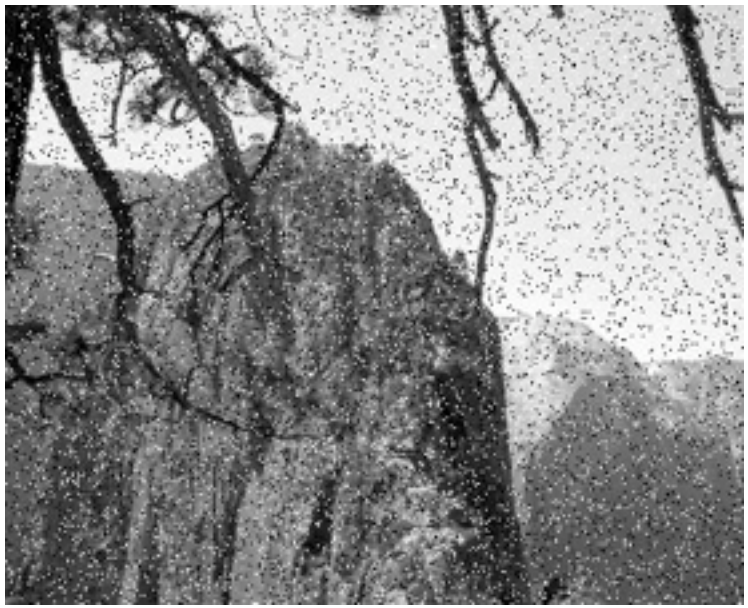




Impulsrauschen

- Einzelne Pixel sind gestört.
- Störung ist maximal (d.h. Pixel ist entweder schwarz oder weiß; Salt-and-Pepper-Noise)

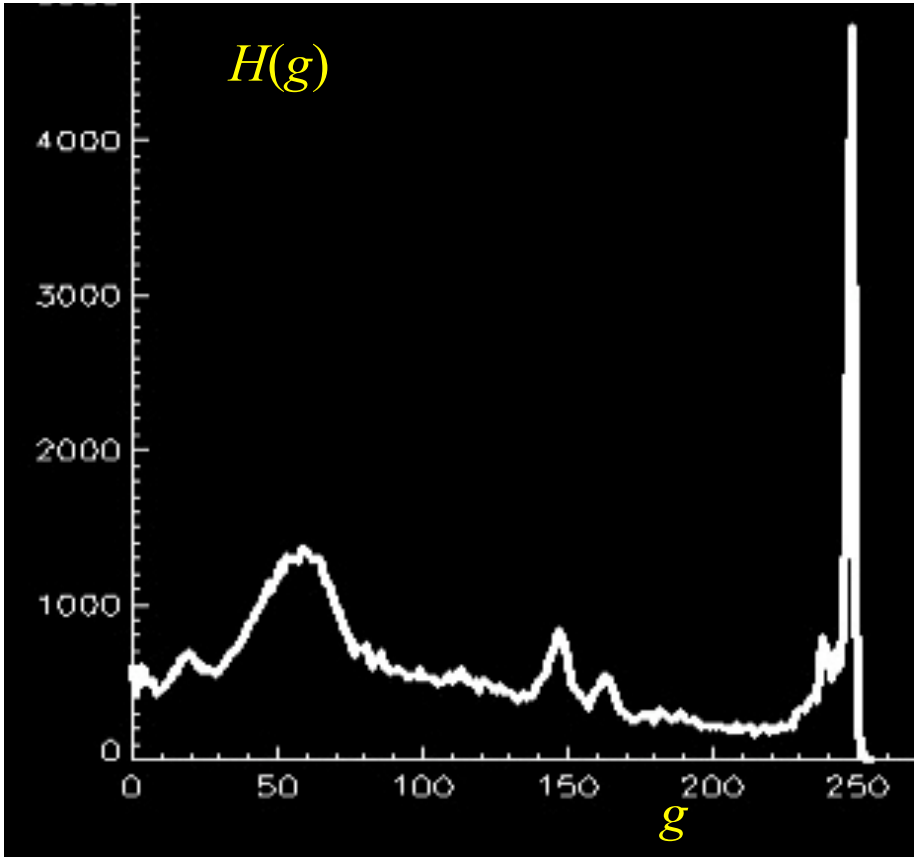




Histogramm

Häufigkeit $H(g)$ der Grauwerte $g=\{0,1,...,N\text{-}1\}$ in einem Bild.

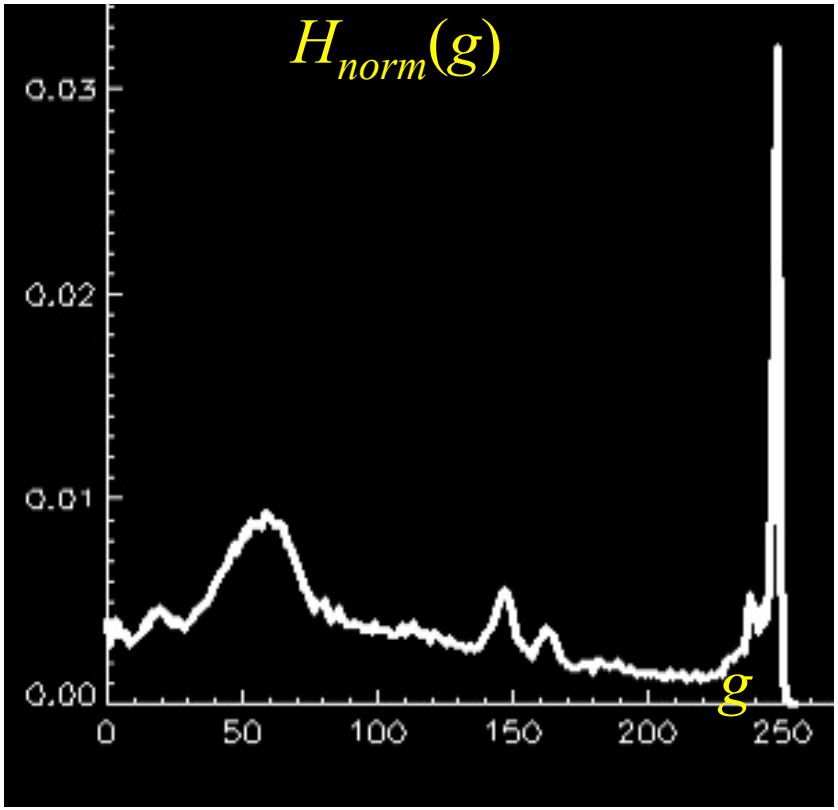




Normiertes Histogramm

- Normierung nach Anzahl der Pixel eines Bildes (Größe *M*x*N*): $H_{norm}(g) = H(g) / (M \cdot N)$
- Ein normiertes Histogramm gibt für jeden Grauwert *g* die Wahrscheinlichkeit an, dass ein beliebiges Pixel diesen Grauwert hat.





Informationsgehalt

Messbare Einheit von Information mit intuitiver Bedeutung.

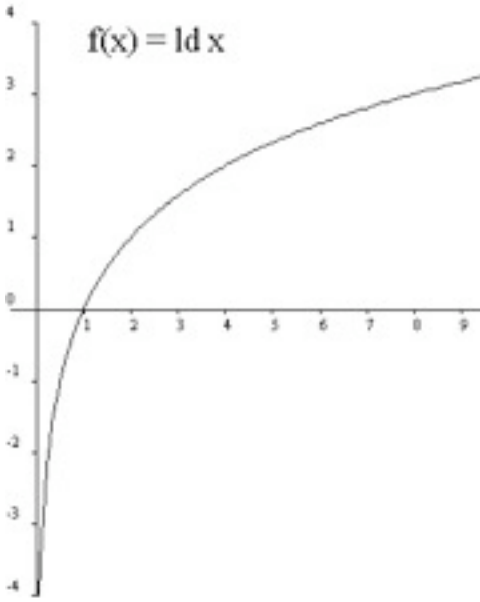
1. **Ansatz:** Information $I(E)$ eines Grauwerts E ist umso höher, je größer die Gesamtanzahl N der verwendeten Grauwerte ist:
   - $I_N(E) = N.$
   - Informationsgehalt ist unabhängig davon, welcher Grauwert aus der Liste $E=\{E_0,E_1,...,E_{N-1}\}$ übermittelt wurde.
   - Für m-wertige Symbole kann eine Informationseinheit für einen Grauwert als $\log_m I_N(E)$ definiert werden.
   - Beispiel:
     - Anzahl der Grauwerte: 256
     - Informationsgehalt jedes Grauwerts: 256
     - Symbol: Bit (2-wertig)
     - Informationseinheit: $\log_2 256 = 8$

Informationsgehalt

- **Nachteil**: Informationsgehalt eines häufig vorkommenden Grauwerts ist genauso groß wie die eines selten vorkommenden Werts.
- Information $I(E)$ eines Pixelwerts E unter Berücksichtigung der Häufigkeit von E:
  - Umgekehrt proportional zur Wahrscheinlichkeit $P(E)$ des Eintreffens.
  - Logarithmus zur Basis n (n – Wertigkeit der Informationseinheit)

$$I(E) = \log_n 1/P(E) = - \log_n P(E)$$

- Zur Repräsentation der Information $I(E)$ werden $I(E)$ Informationseinheiten benötigt.
- Beispiel für Bits:
  - Wahrscheinlichkeit für Eintreffen von E sei 0.5
  - $I(E) = \log_2 2 = 1$.



Informationsgehalt einer Pixelfolge

- Grauwertbereich $\{g_0, g_1, ..., g_{N-1}\}$
- Wahrscheinlichkeiten $\{P(g_0), ..., P(g_{N-1})\}$
- Informationsgehalt einer Folge der Länge k – Wahrscheinlichkeit des Auftretens gewichtet mit Informationsgehalt:

$$-k \times P(g_0) \times \log_2 P(g_0) - k \times P(g_1) \times \log_2 P(g_1) - k \times P(g_2) \times \log_2 P(g_2) ..$$
$$= -k \sum_{i=0}^{N-1} P(g_i) \times \log_2 P(g_i)$$

- Durchschnittlicher Informationsgehalt = **Entropie**:

$$Entropie(P) = -\sum_{i=0}^{N-1} P(g_i) \times \log_2 P(g_i)$$

- Das normierte Histogramm kann als Schätzung für P verwendet werden.

Bildverbesserung

- Verbesserung von Bildeigenschaften zur besseren Wahrnehmbarkeit oder zur Vorbereitung von Analyseschritten.
- Bildeigenschaften:
  - Signal-Rausch-Verhältnis
  - Kontrast
  - Informationsgehalt
- Punktbasierte Methoden
- Flächenbasierte Methoden



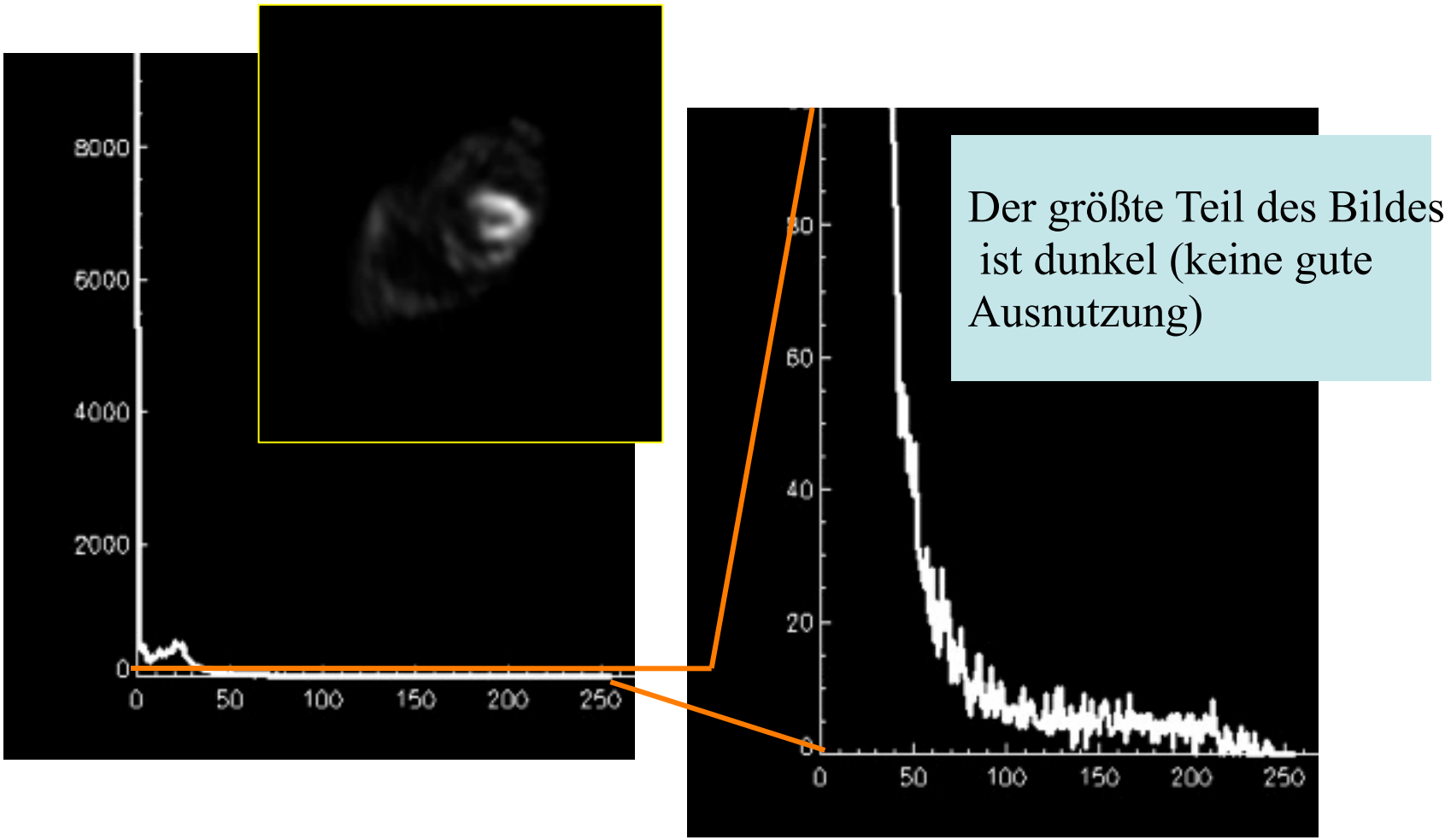


Welches Bild ist besser?
Warum?
Wie ist das messbar?

Pixelbasierte Bildverbesserung

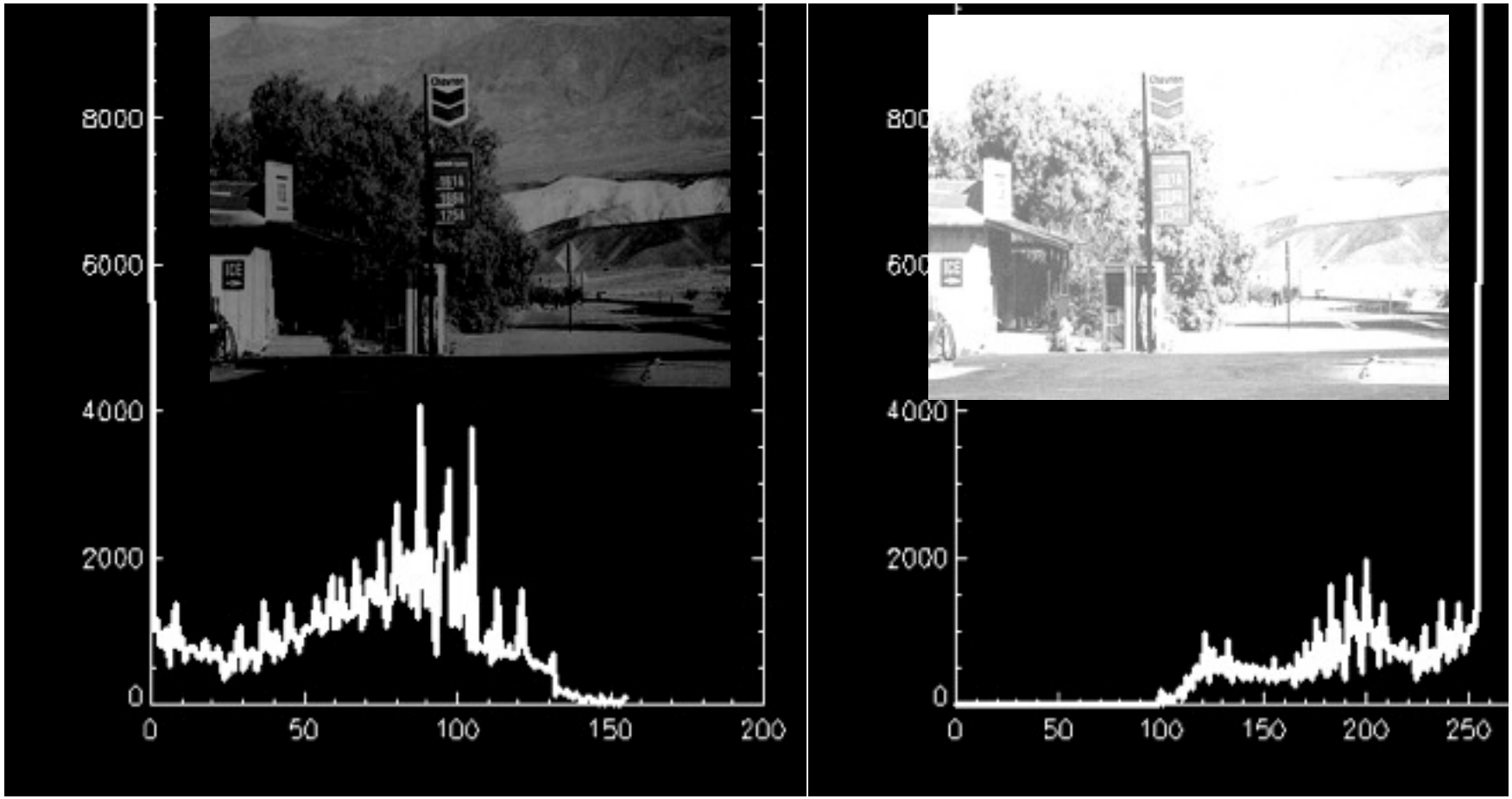
- Abbildung der Grau-/Farbwerte unabhängig von ihrem Ort oder ihrer Zuordnung
  - innerhalb der Grau- bzw. Farbwerte: $g_{neu} = f(g)$ oder $[r_{neu}, g_{neu}, b_{neu}] = [f(r), f(g), f(b)]$
  - von Grauwerten in Farbwerte (Falschfarbdarstellung): $[r_{neu}, g_{neu}, b_{neu}] = [f_1(g), f_2(g), f_3(g)]$
- Qualitätsmerkmal: globaler/lokaler Kontrast, Entropie
- Methoden
  - Monotone Abbildung der Grauwerte
  - Nichtmonotone Grauwertabbildung
  - Falschfarbdarstellung

Nutzung des Grauwertspektrums



Der größte Teil des Bildes ist dunkel (keine gute Ausnutzung)

Unter-/Überbelichtung



Kontrast

- Globaler Kontrast: Größter Grauwertunterschied im Bild

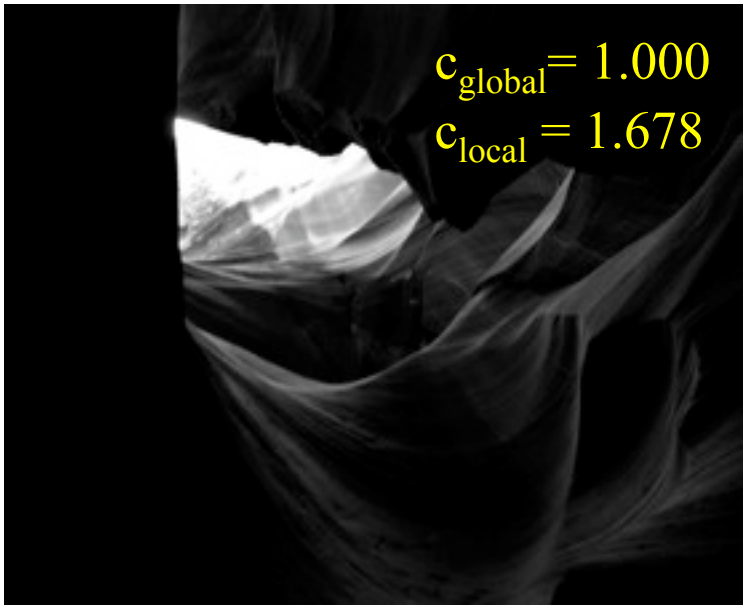
  $c_{global}(f) = [\max_{m.n}(f(m,n))-\min_{m,n}(f(m,n))]/g_{range}$.

  mit g_{range} - Grauwertbereich

- Lokaler Kontrast: z.B. durchschnittlicher Grauwertunterschied zwischen benachbarten Pixeln

  $c_{local}(f) = 1/MN \sum_m \sum_n |f(m,n)-f_{nb}(m,n)|$

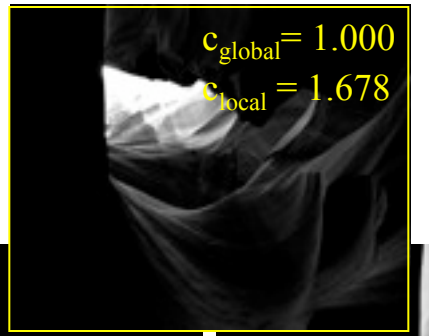
  mit $f_{nb}(m,n)$ – durchschnittlicher Grauwert in der Umgebung von (m,n).



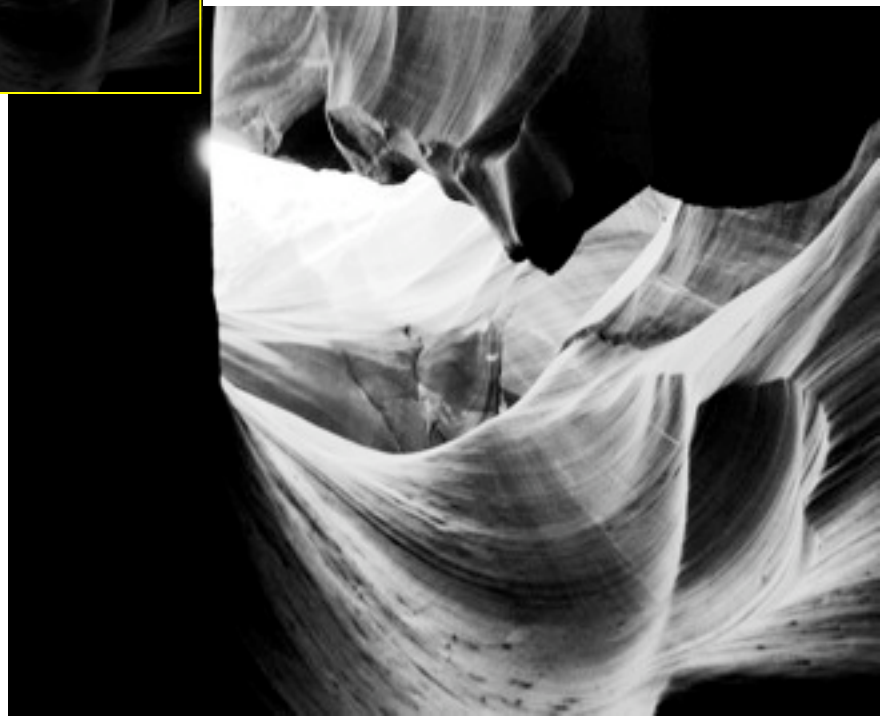
Globaler / Lokaler Kontrast

$c_{global} = 0.500$
$c_{local} = 0.839$





$c_{global} = 1.000$
$c_{local} = 4.580$

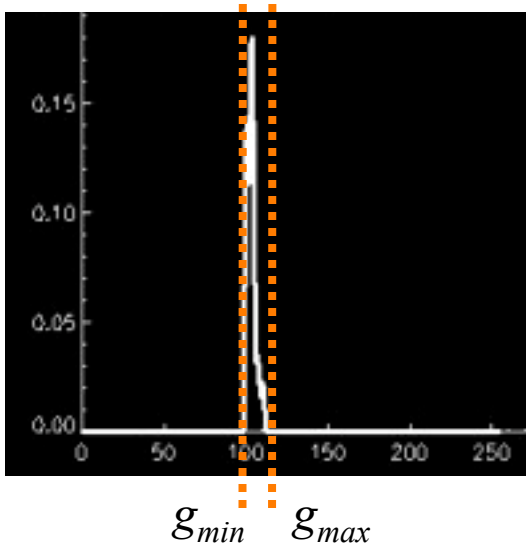


Maximierung des globalen Kontrasts

- Kontrastumfang g_{max}-g_{min} im Verhältnis zum maximalen Wertebereich w_{min}...w_{max} (z.B. 0...255) ist Skalierungsfaktor.

- Transferfunktion

$$g'(g) = (g - g_{\min}) \times \frac{w_{\max} - w_{\min}}{g_{\max} - g_{\min}}$$



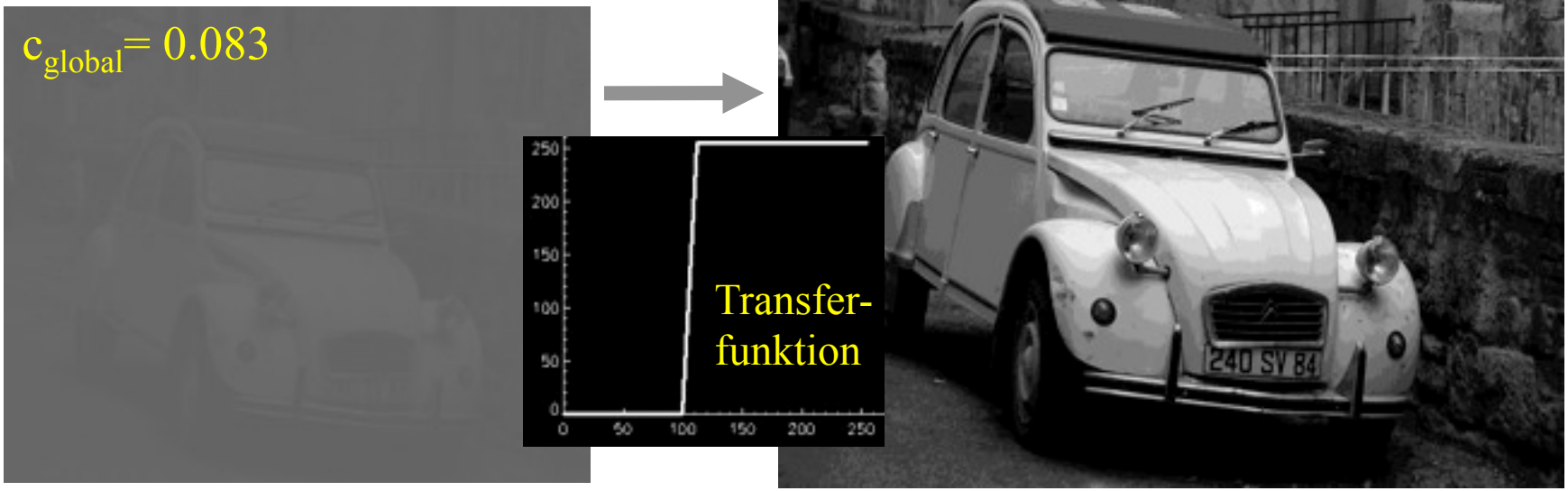
Histogramm



Maximierung des globalen Kontrasts

$$g'(g) = (g - g_{\min}) \times \frac{w_{\max} - w_{\min}}{g_{\max} - g_{\min}}, \quad g_{\min} = 100,\ g_{\max} = 112,\ w_{\min} = 0,\ w_{\max} = 255$$

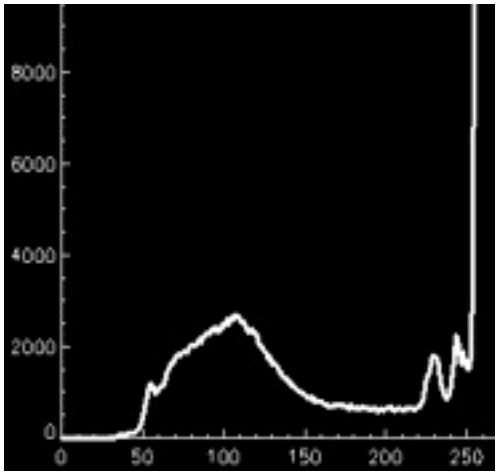
$$g'(g) = (g - 100) \times \frac{255}{12}$$



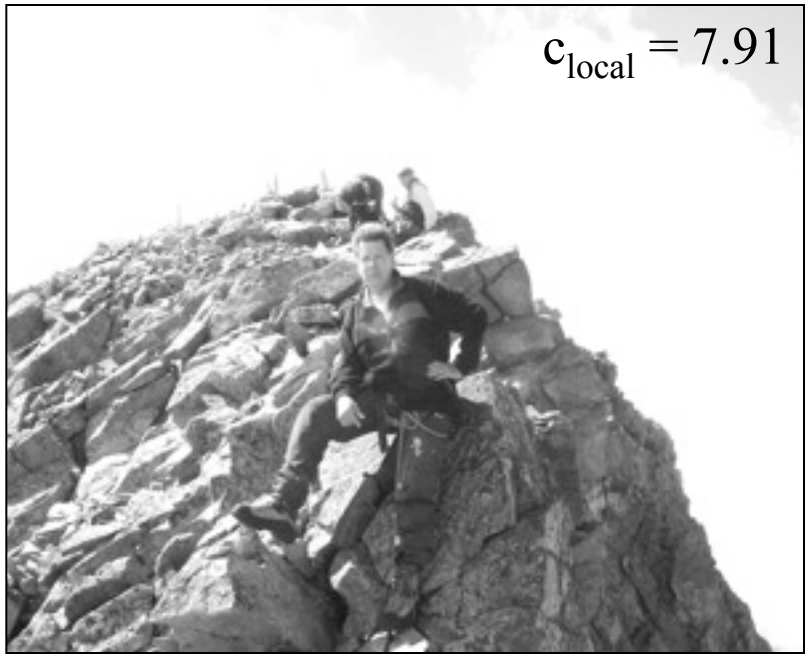
Verbesserung des lokalen Kontrasts

- Bild ist zu hell (zu dunkel), aber Grauwertbereich ist nahezu ausgenutzt.
- Nichtlineare, monotone Transferfunktion, z.B. Gammakorrektur:

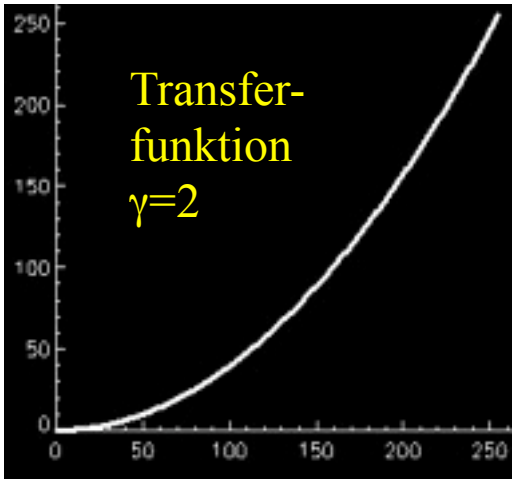
$$g'(g) = w_{\max} \times \left(\frac{g}{w_{\max}} \right)^{\gamma}$$



Histogramm



Verbesserung des lokalen Kontrasts

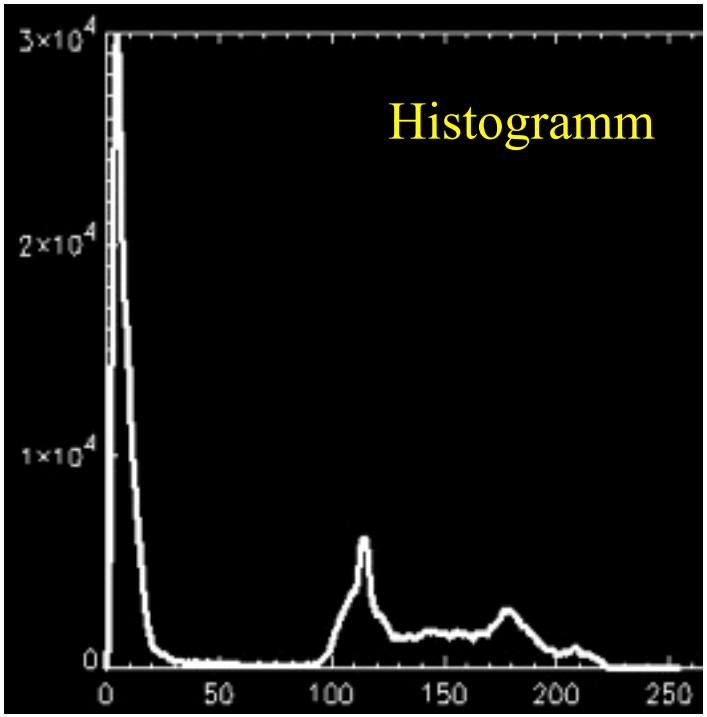






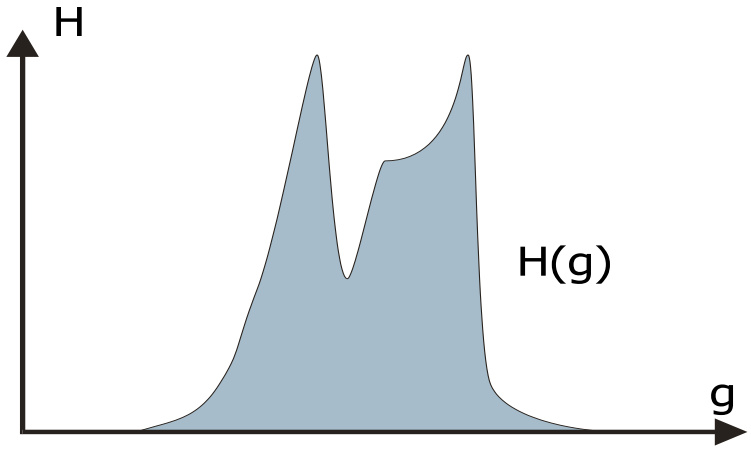
Maximierung des Informationsgehalts

- Gibt es eine „optimale“ Korrektur?
- Optimal = maximaler Informationsgehalt





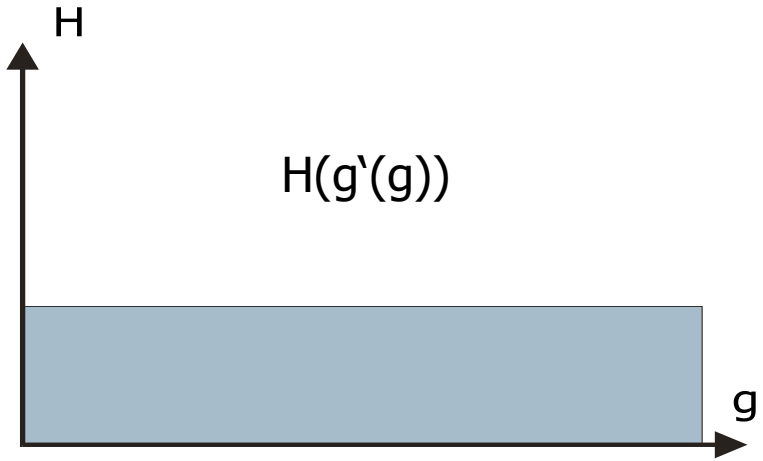
Maximaler Informationsgehalt



Entropie ist maximal, falls $P(g_i)$=const für

i=0,N-1

gesucht: Histogrammtransformation $g'(g)$ zur Maximierung der Entropie



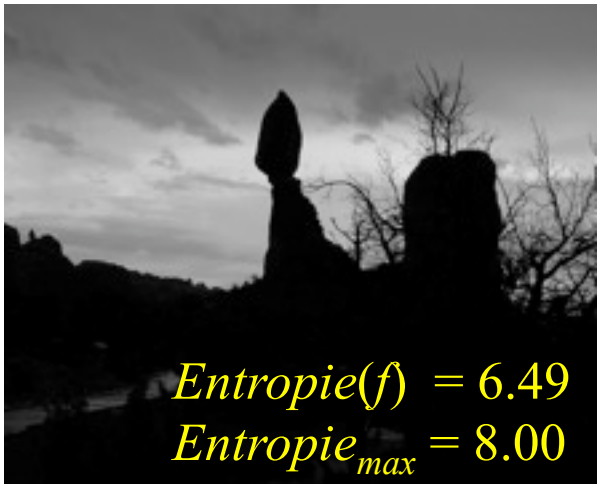
Annahme:

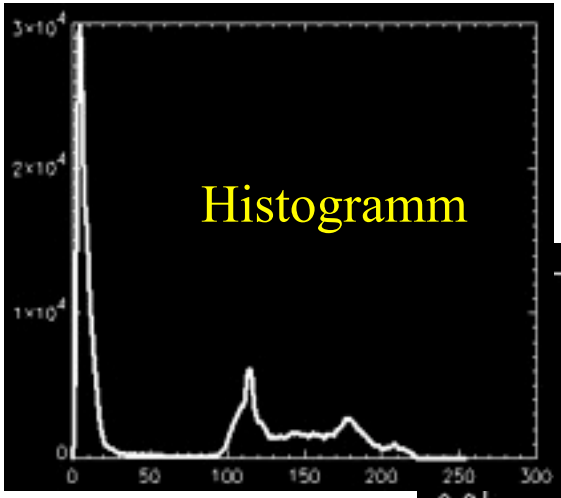
$H(g)$ ist normiert und kontinuierlich, d.h., $\int H(g)=1$.

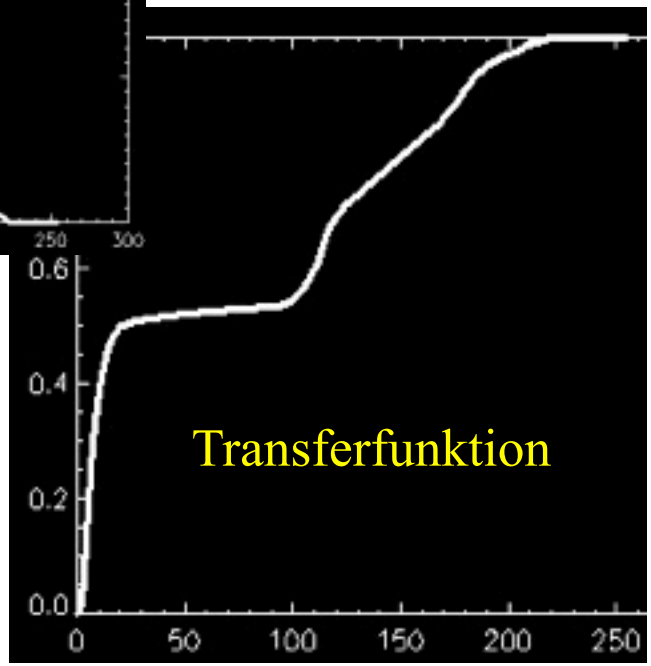
Dann existiert die folgende Transferfunktion g':

$$g'(g) = \int_{0...g} H(w)\, dw$$

Beispiel







Aber: was ist, falls $g'(g)\cdot(N\text{-}1)$ keine ganze Zahl ist?

Histogrammlinearisierung

Transferfunktion für ein diskretes Histogramm:

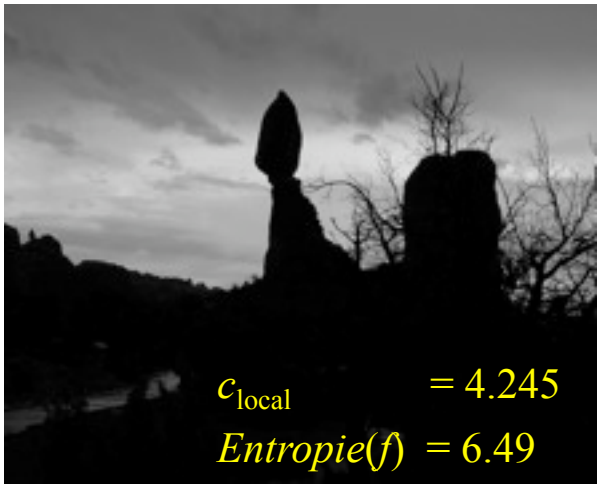
$$E[H(g)] = \lceil N_g \cdot \sum_{w=0\ldots g} H(w) \rceil - 1, \text{ mit: } N_g \text{ - Anzahl der Grauwerte.}$$

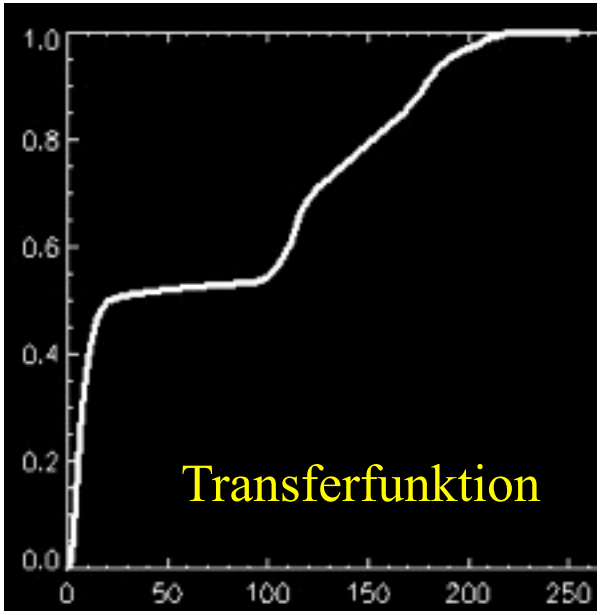
Beispiel:

Grauwert	0	1	2	3	4	5	6	7
Häufigkeit	50	150	350	250	100	60	30	10
H(g)	0.05	0.15	0.35	0.25	0.10	0.06	0.03	0.01
kumulativ	0.05	0.20	0.55	0.80	0.90	0.96	0.99	1.00
Grauwert	0.4	1.6	4.4	6.4	7.2	7.68	7.92	8.00
aufgerundet	1	2	5	7	8	8	8	8
$\lceil\ \rceil$ **-1**	0	1	4	6	7	7	7	7

Keine Linearisierung, sondern von der Häufigkeit abhängige Spreizung.

Beispiel







Warum wurde die Entropie **kleiner**?

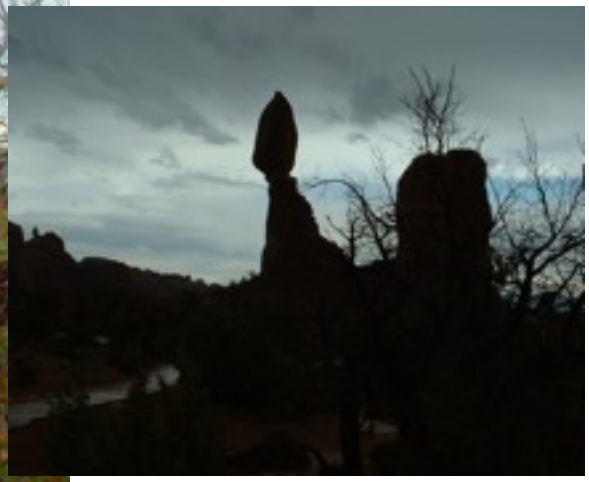
Beispiel



Das geht auch in Farbe.

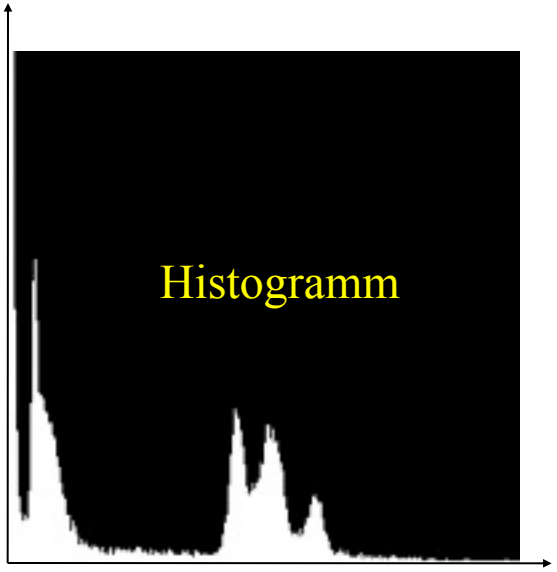
Farbkanäle werden unabhängig voneinander behandelt.

(Ist das eine gute Idee?)

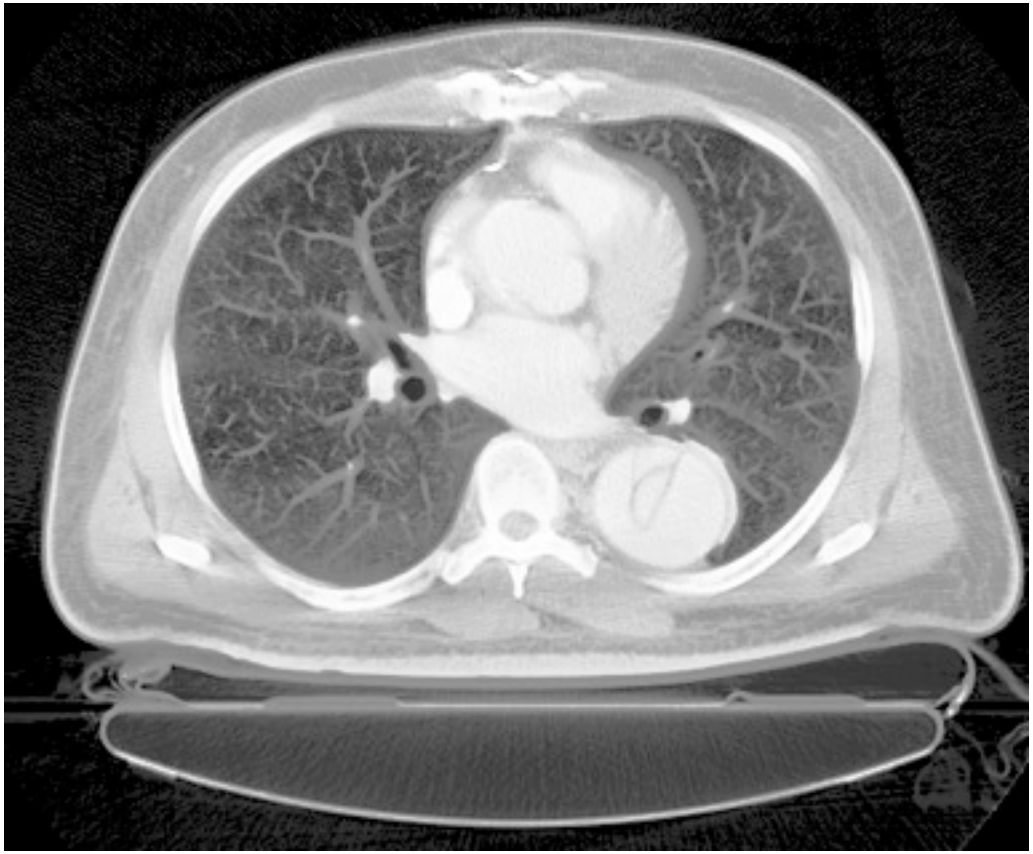


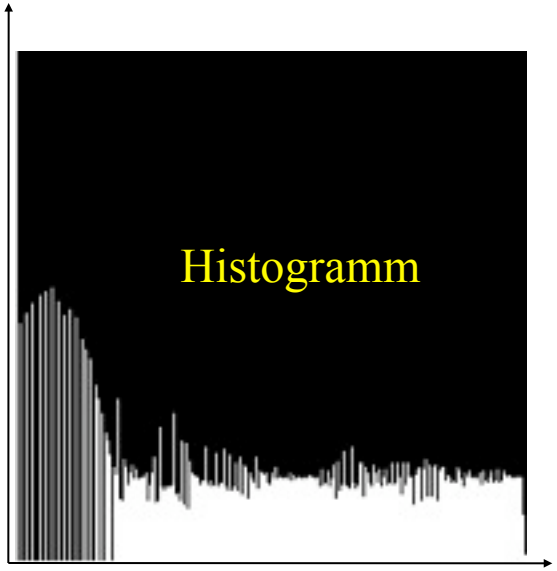
Histogrammlinearisierung



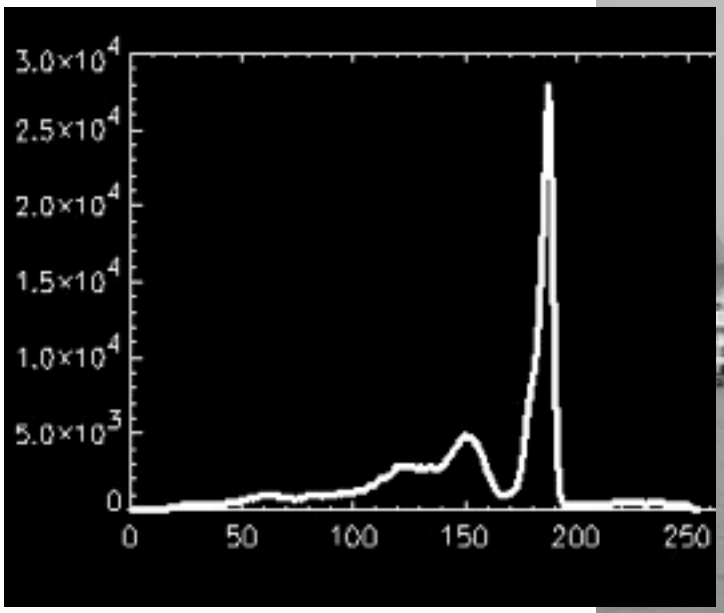


Histogrammlinearisierung





Problem





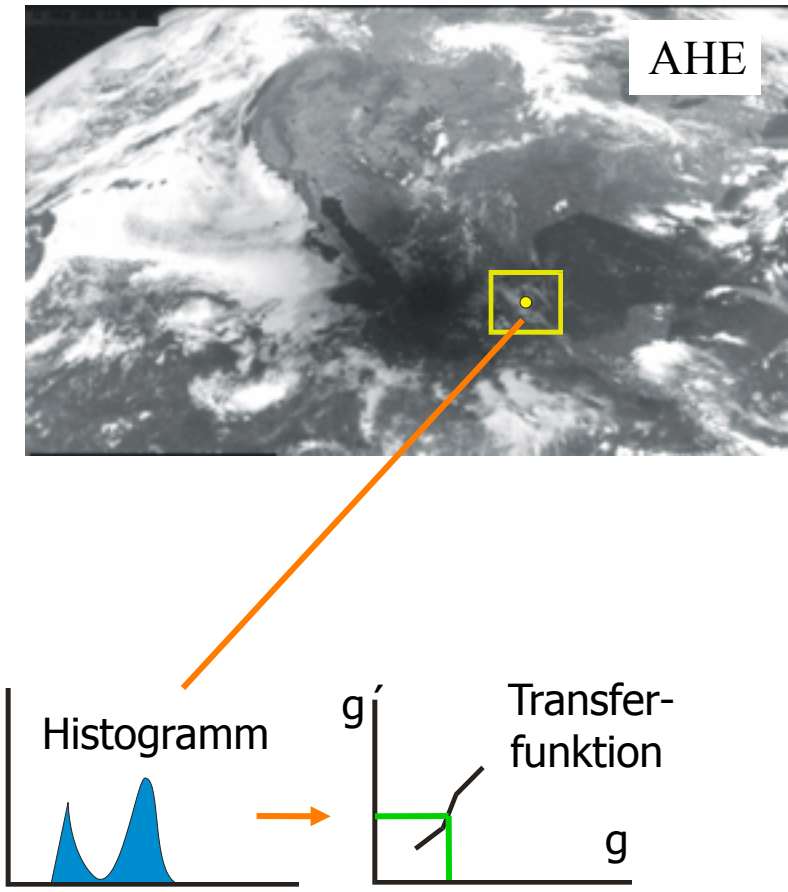
Problem



Das „Unwichtige“ wurde verstärkt, das Wichtige abgeschwächt!

Histogrammlinearisierung - Varianten

- Adaptive Histogram Equalisation (AHE)
  - Histogramm wird an jedem Punkt für eine vorgegebene Umgebung erstellt.
  - Linearisierung nach diesem Histogramm
  - Nur der Grauwert des betreffenden Punkts wird modifiziert
- Contrast Limited Adaptive Histogram Equalisation (CLAHE):
  - wie AHE, aber Kontrastverstärkung nur bis zu einem gewissen Maximum.
  - verhindert die bei AHE vorkommende Kontrastverstärkung im Bildhintergrund.



Adaptive Histogrammlinearisierung





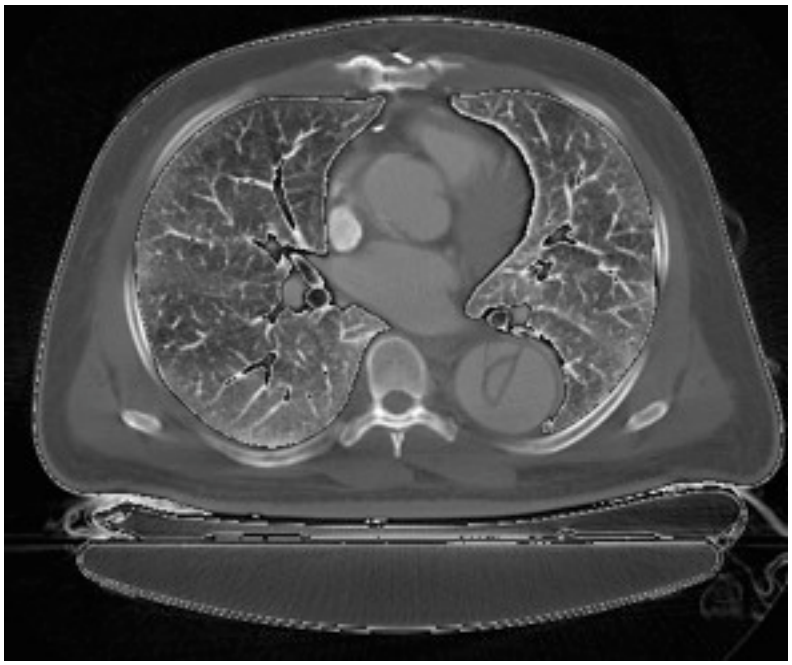
Kontrastlimitierte AHE



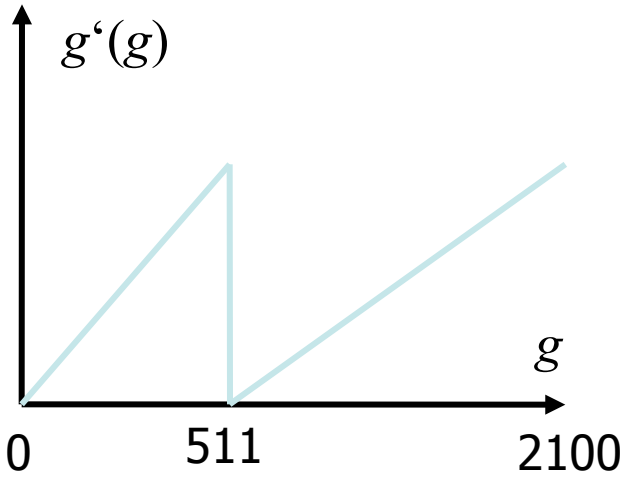




Nichtmonotone Grauwertabbildung

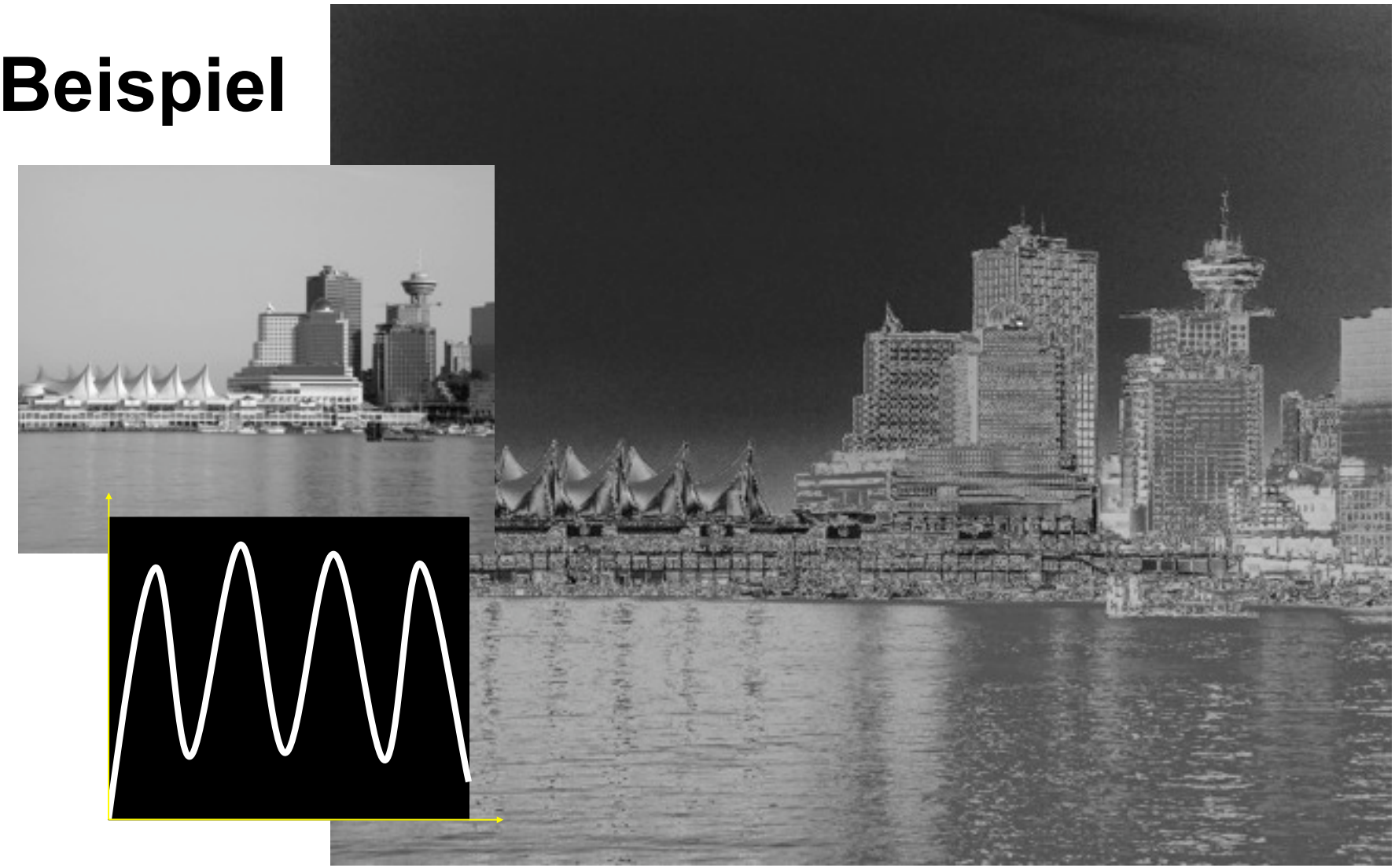


Zwei Grauwertfenster in einem Bild.



- Erzeugt künstliche Kanten.
- Grenzen von Maxima der Transferfunktion nicht immer erkennbar.

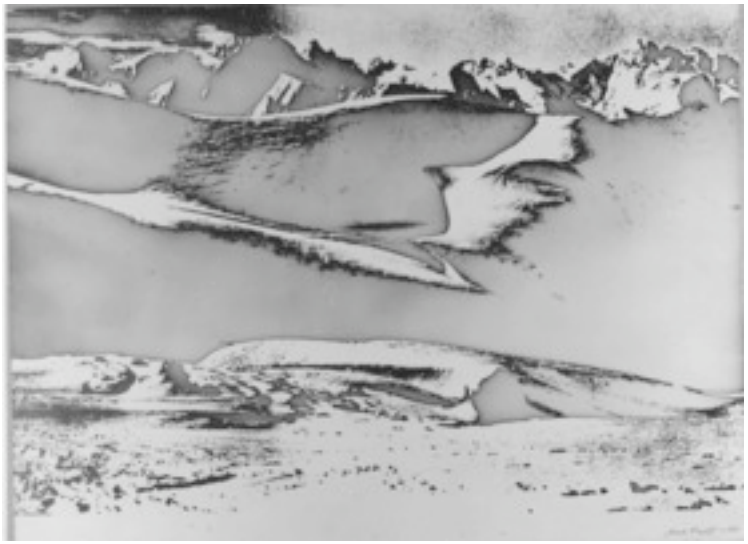
Beispiel



Eine Zuordnung zwischen Helligkeit und ursprünglichem Grauwert ist nicht mehr herstellbar.

Analoge Variante: Solarisation

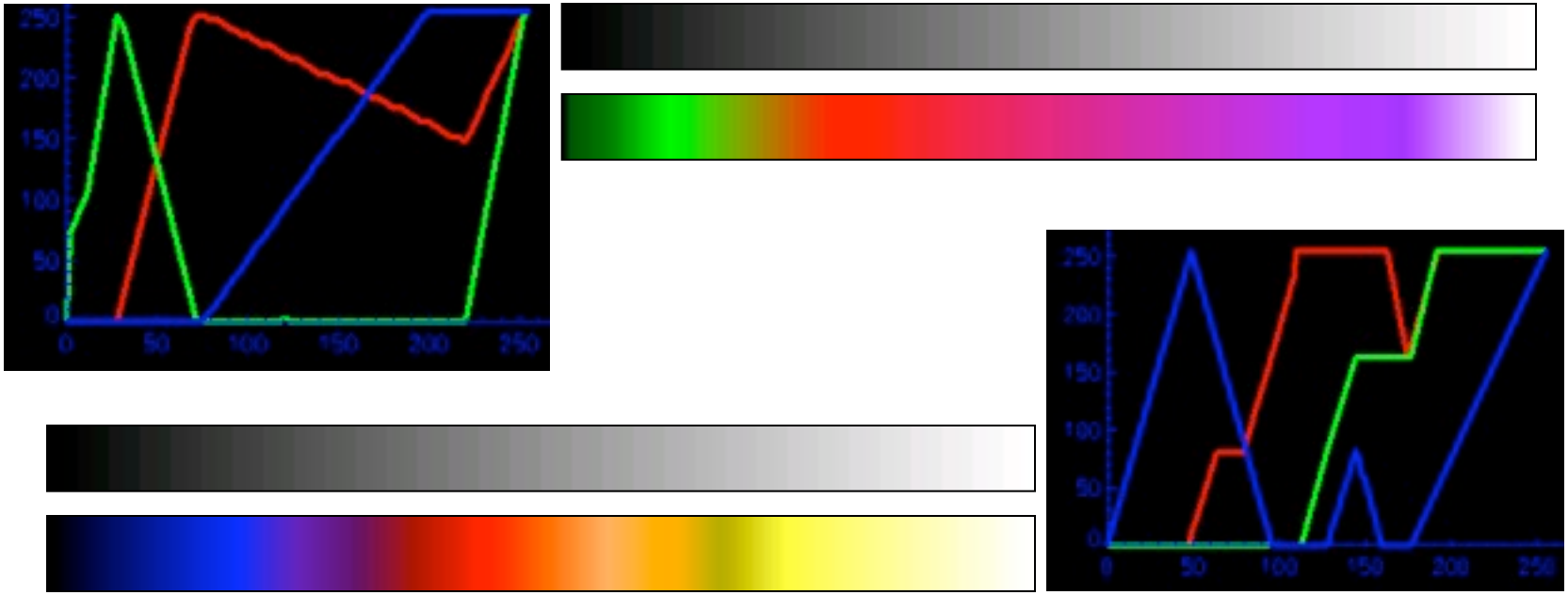




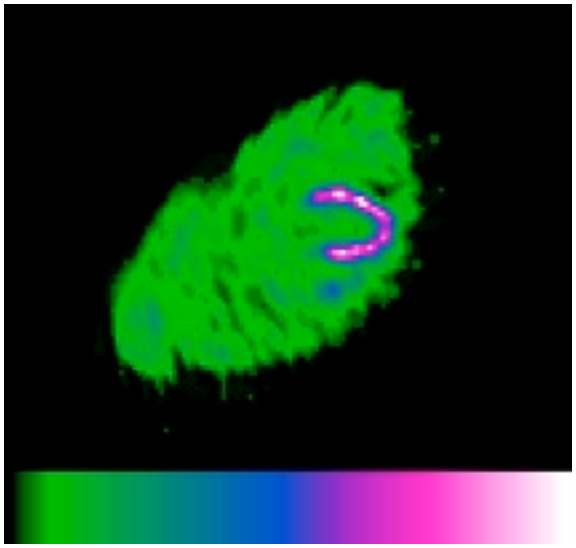
- Lange bekannter chemischer Effekt bei der Negativ- oder Positiventwicklung
- Entsteht durch zus. Belichtung während der Entwicklung
  - z.B.: jemand betritt unangemeldet die Dunkelkammer
- Bewirkt nichtmonotone Schwärzungskurve
- Schwer zu steuern in der analogen Photographie
- Fotos oben von 1938, 1950 http://www.bsz-bw.de/eu/hoepffner-schoffers/

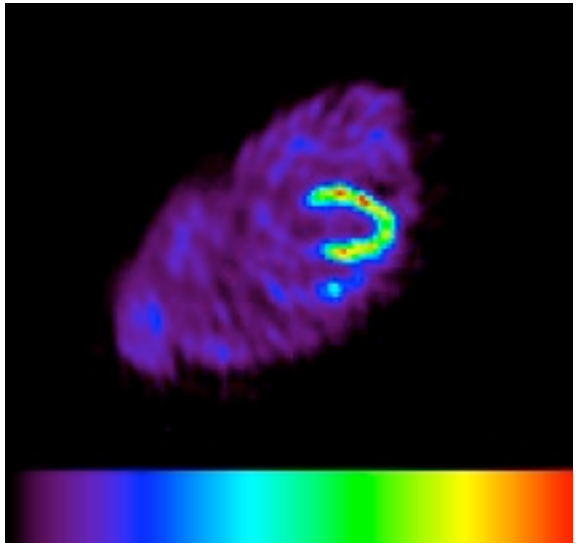
Farbe zur Kontrastverstärkung

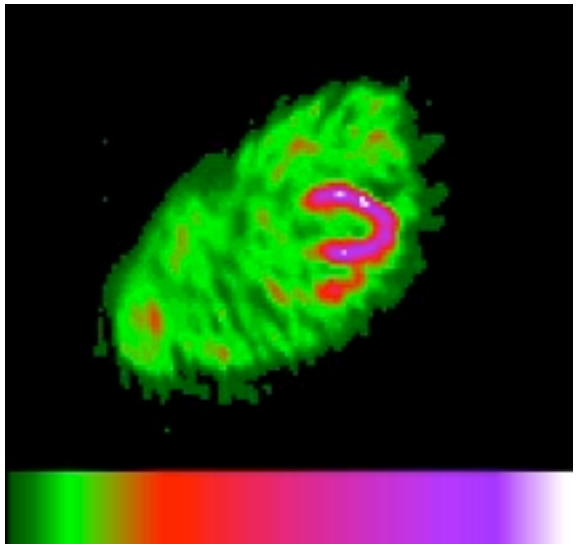
- Es können wesentlich mehr Farb- als Grauwerte unterschieden werden.
- Kontrastverstärkung durch drei nicht-lineare, nicht-monotone Abbildungsfunktionen der Grauwerte: *redi*(*g*), *greeni*(*g*) *bluei*(*g*)

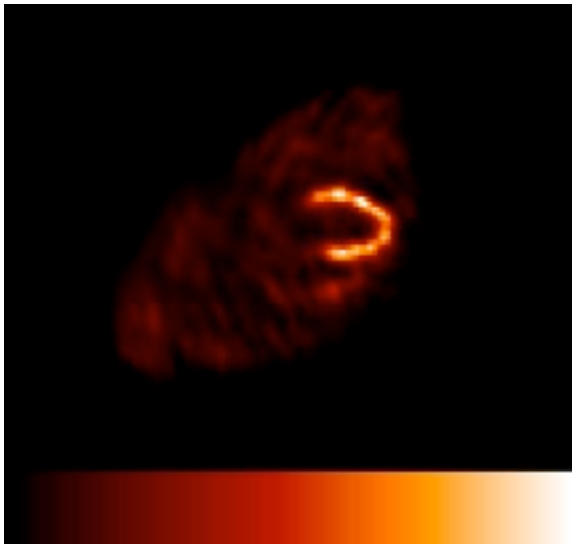


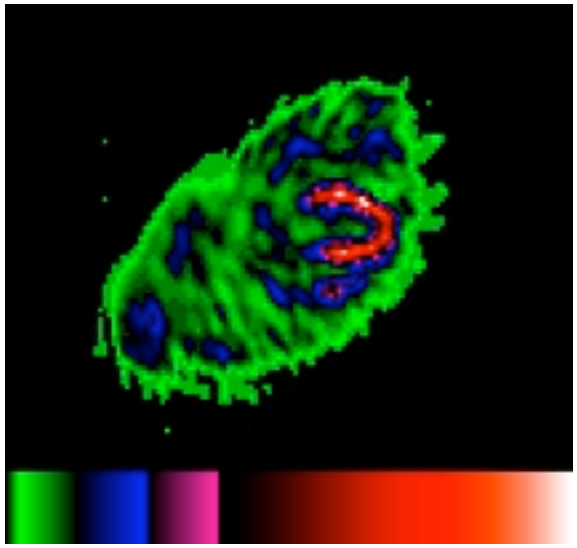
Beispiel







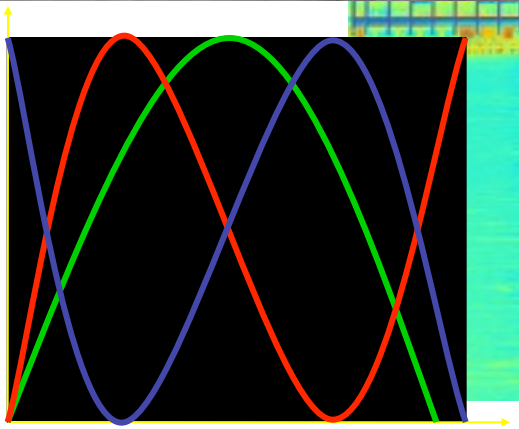




Achtung:
Nichtlineare Transformationen erzeugen künstliche Kanten.

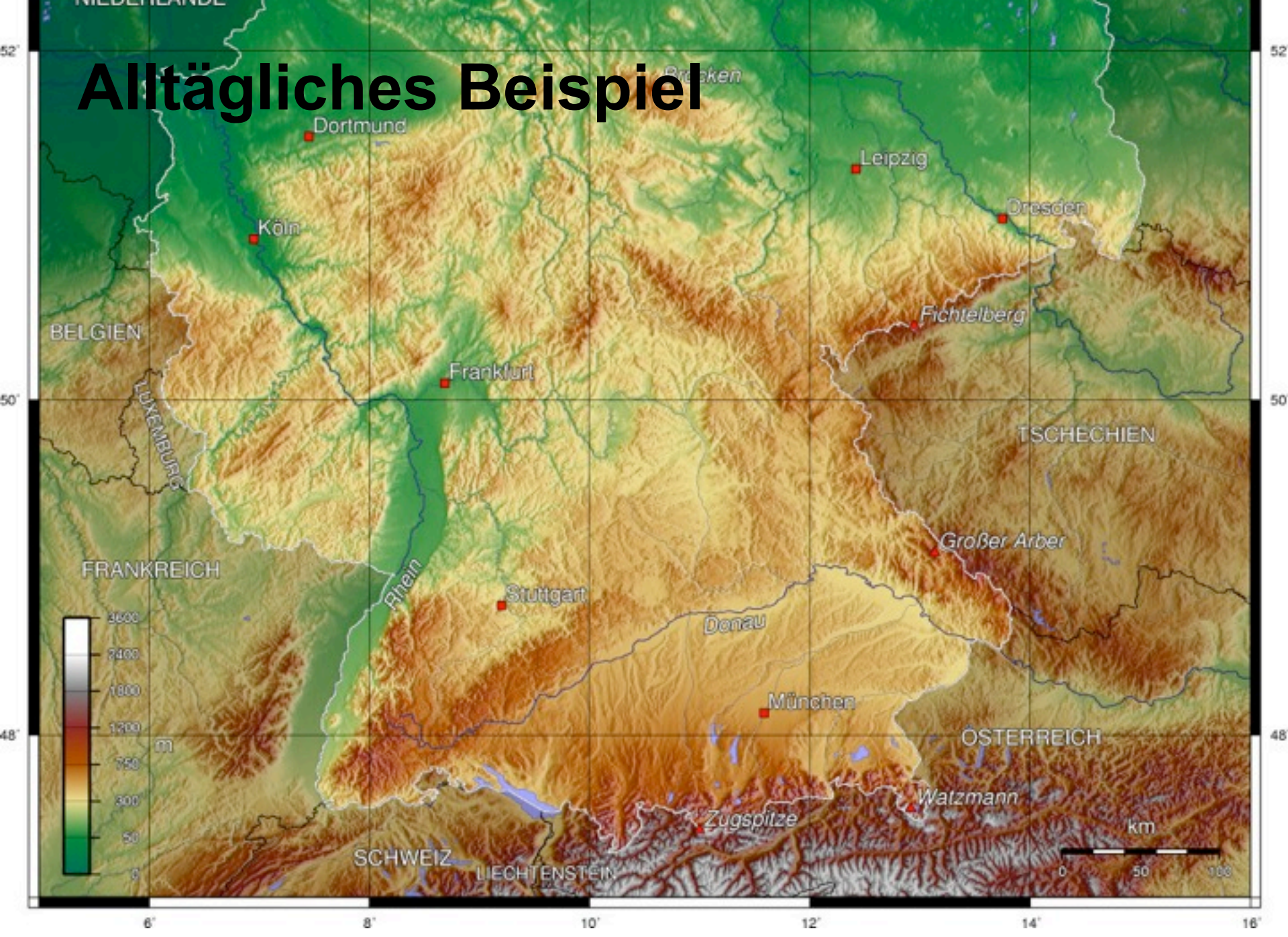
Beispiel





Ein Zusammenhang zwischen Farbe und Grauwert ist nicht mehr erkennbar.

Alltägliches Beispiel



Was sollten Sie bis hierhin gelernt haben?

- Punktbasierte Verfahren werden über eine Transferfunktion zwischen Grauwerten (Farbwerten) definiert.
- Grauwerttransformationen
  - monoton: linear, γ-Korrektur, Histogrammlinearisierung
  - Nicht monoton: Stufentransformation, Falschfarbdarstellung.
- Erfolg kann an kontrastbasierten Maßzahlen ermittelt werden.
- Objektabhängige Bildverbesserung erfordert Zusatzwissen.

Flächenbasierte Bildverbesserung

- Rauschen kann durch Integration einer Signalfolge mit (nahezu) konstantem Signal reduziert werden.
- Konstante Signalfolge:
  - Integration über eine zeitliche Folge.
  - Integration über eine homogene Fläche.
- Lineare verschiebungsinvariante Operatoren
  - Konvolutionsmethoden
  - Filterung im Frequenzraum

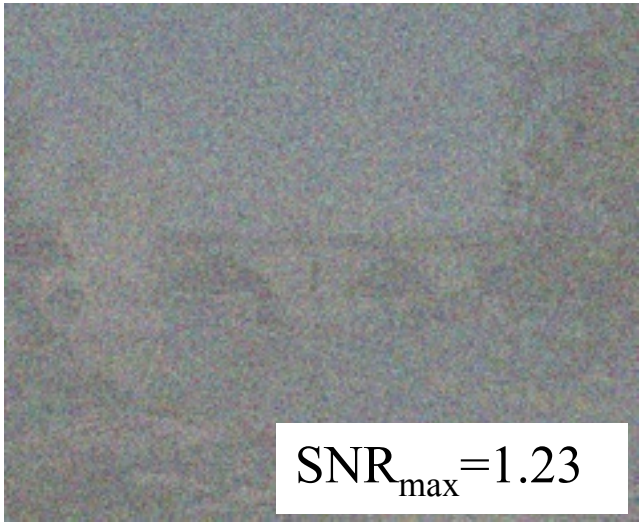
Zeitliche Folge

- Annahmen
  - Aufnahme mehrerer Bilder g_i, i=1,I über einen gegebenen Zeitraum.
  - Bild verändert sich über den Zeitraum nicht (keine Bewegung, keine Beleuchtungsänderung).
  - Erwartungswert E des Rauschens n ist 0.
- Näherung an die unverrauschte Funktion f:
  - $E\{g(m,n)\} = E\{f(m,n)\} + E\{n(m,n)\}$
    $= E\{ f(m,n) \} + 0 = f($
  - Abschätzung von $E\{g(m,n)\}$ durch Integration über die Bilder.



Beispiel

- Einzelne Aufnahme mit normalverteiltem Rauschen (SNR≈1.2).
- Addition von 10 bzw. 50 Aufnahmen.







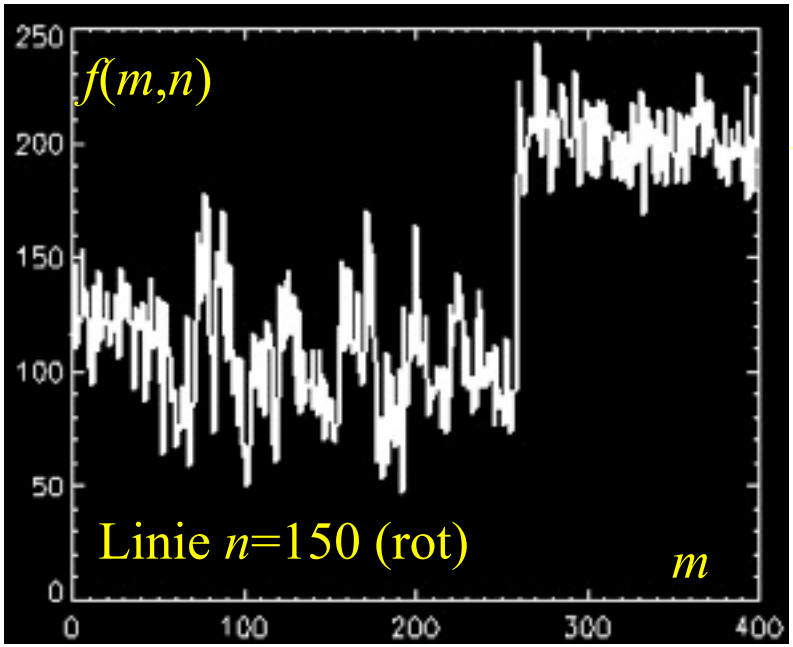
Integration über die Fläche

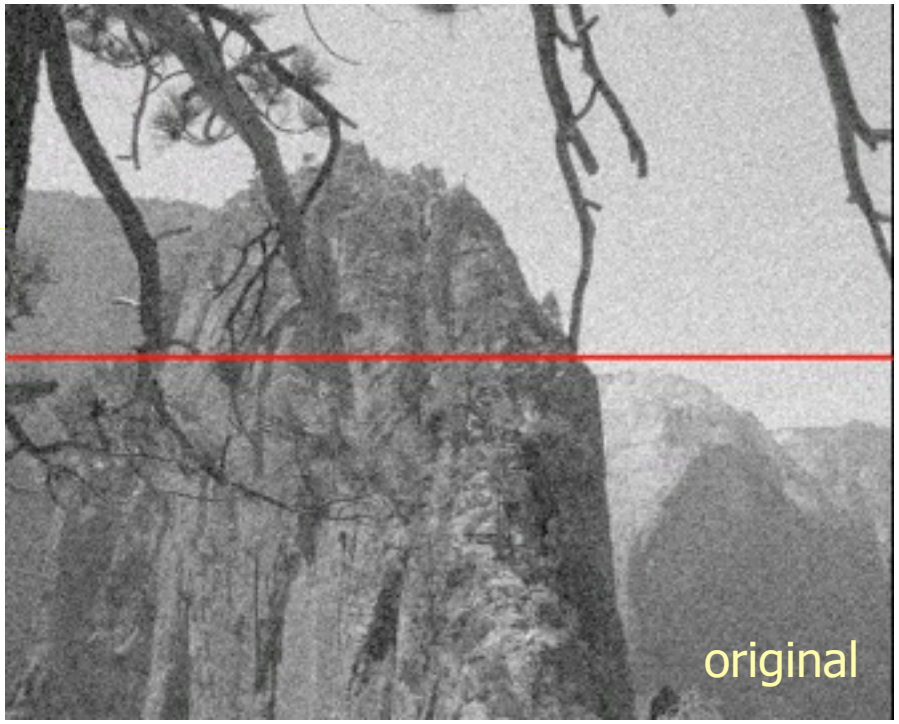
- Falls für eine Reihe von Bildpunkten (p_0,...,p_n) gilt, dass $f(p_i)$=const, dann kann Rauschen n mit $E\{n\}$=0 durch Addition der gemessenen Funktionswerte $g(p_i)$ reduziert werden.
- Annahmen:
  - Bild besteht aus homogenen Bereichen.
  - Benachbarte Punkte haben den gleichen Grauwert.
- Rauschunterdrückung:
  - Mittelwertbildung über vorgegebene Nachbarschaft.



Mittelwertbildung durch Konvolution

Konvolutionskern: Gleichmäßige Gewichtung der Pixel in einer gegebenen Nachbarschaft

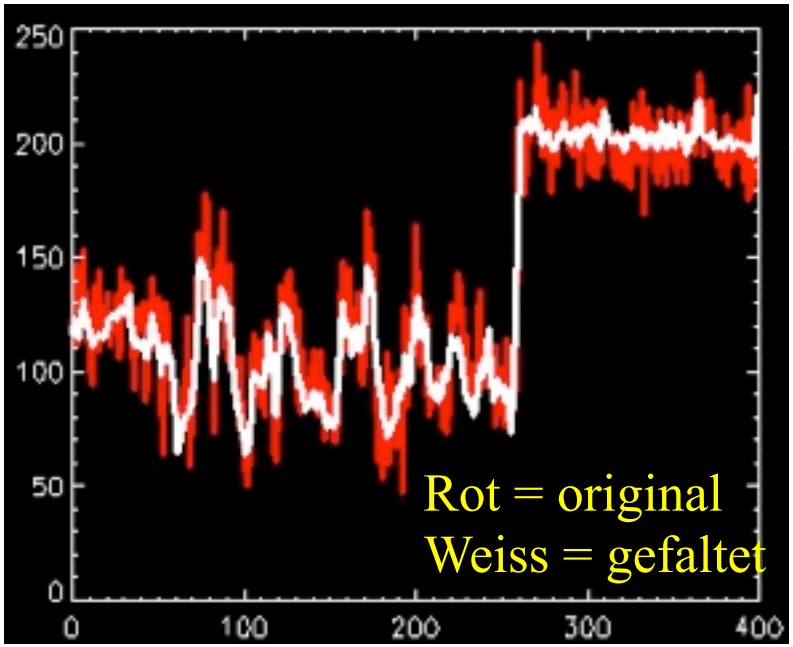


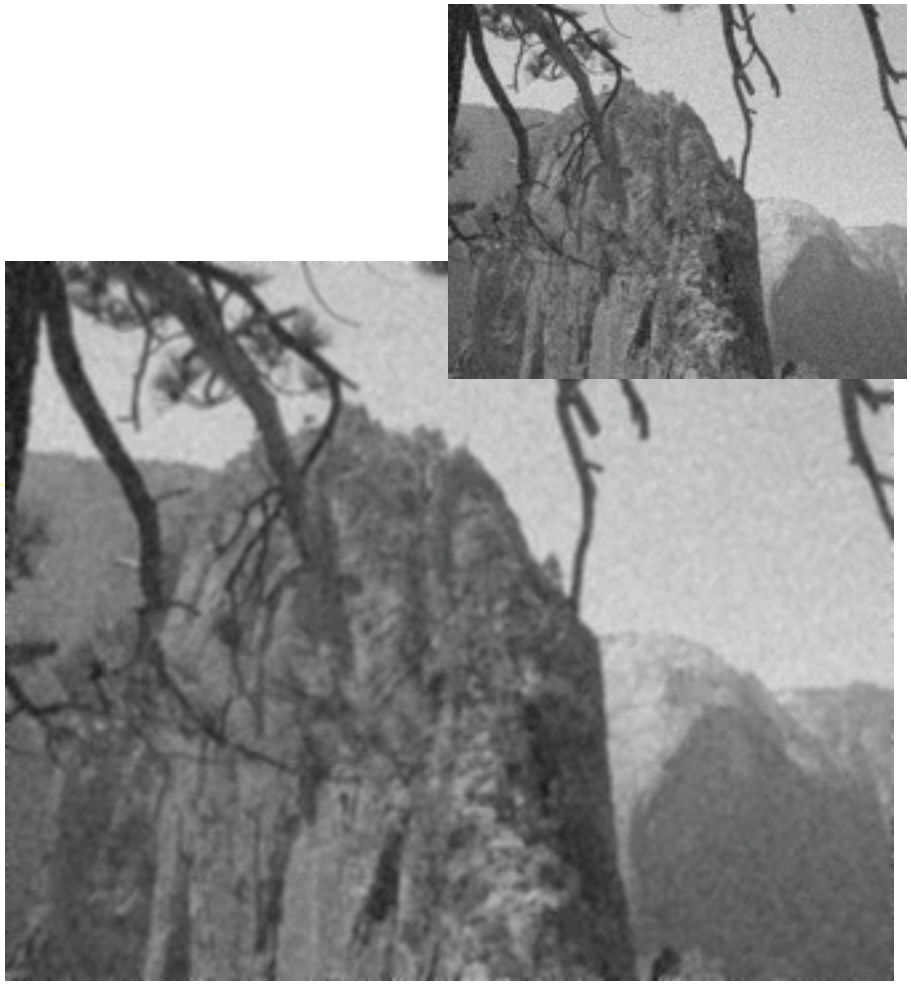


3x3 Boxcar-Filter

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

Filterkern

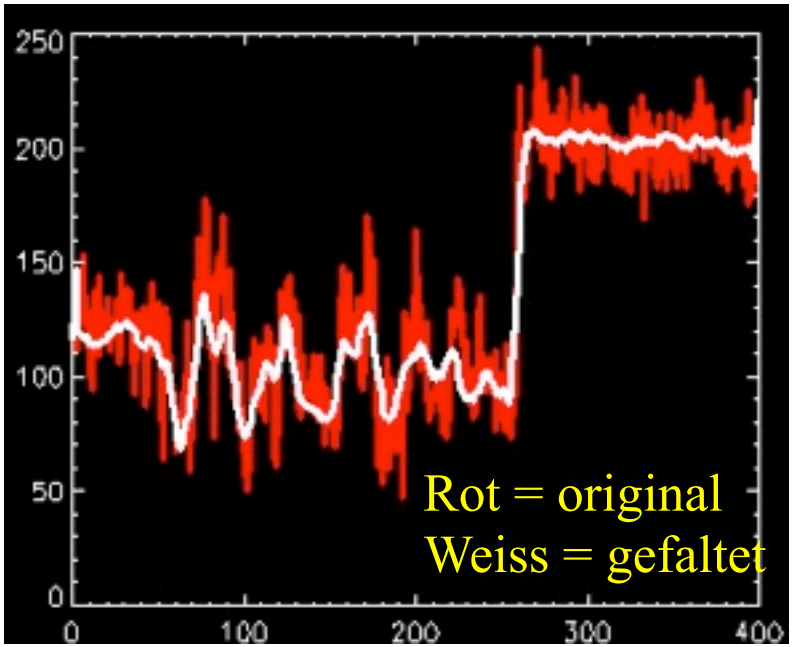


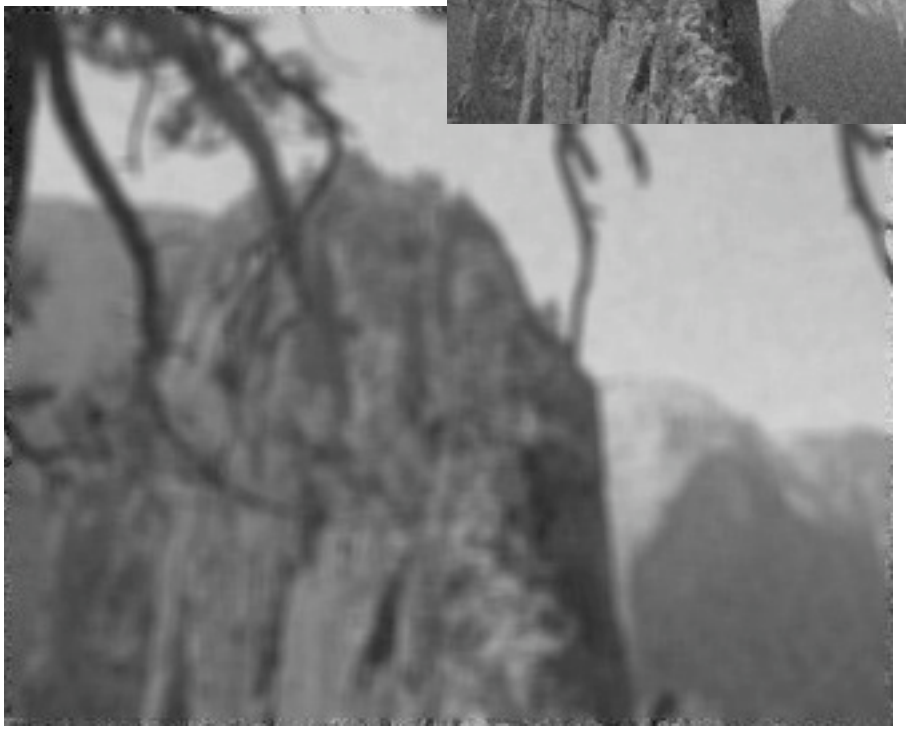


7x7 Boxcar-Filter

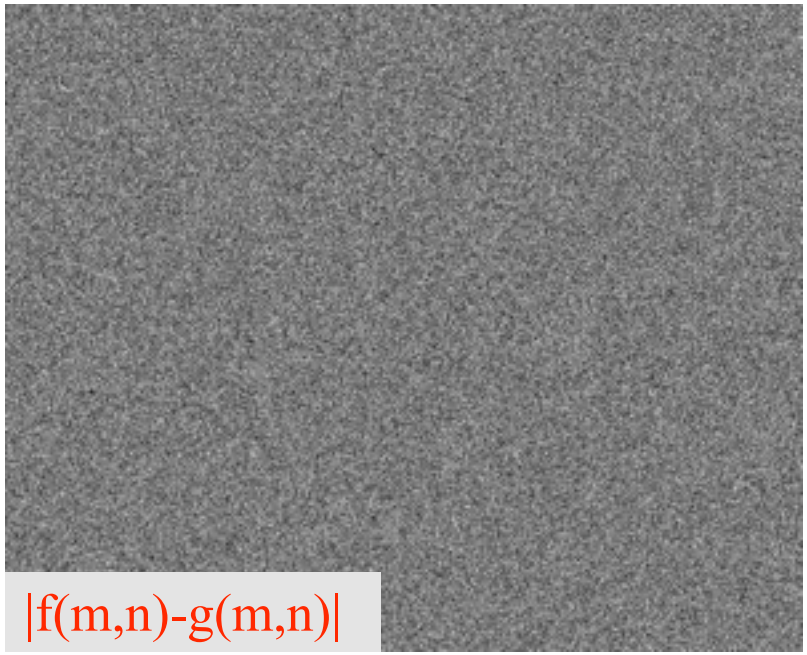
Beobachtung: Kanten werden degradiert.

Grund: Annahme konstanter Funktionswerte ist nicht wahr.

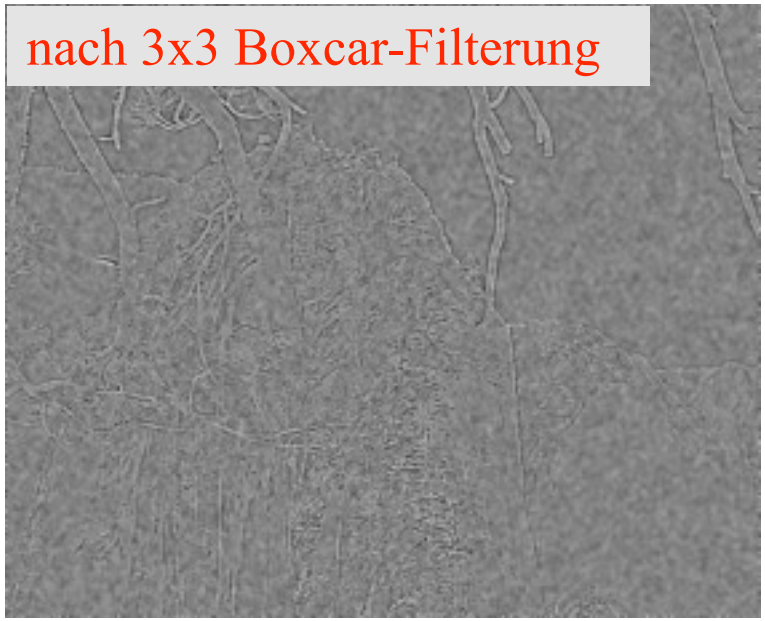




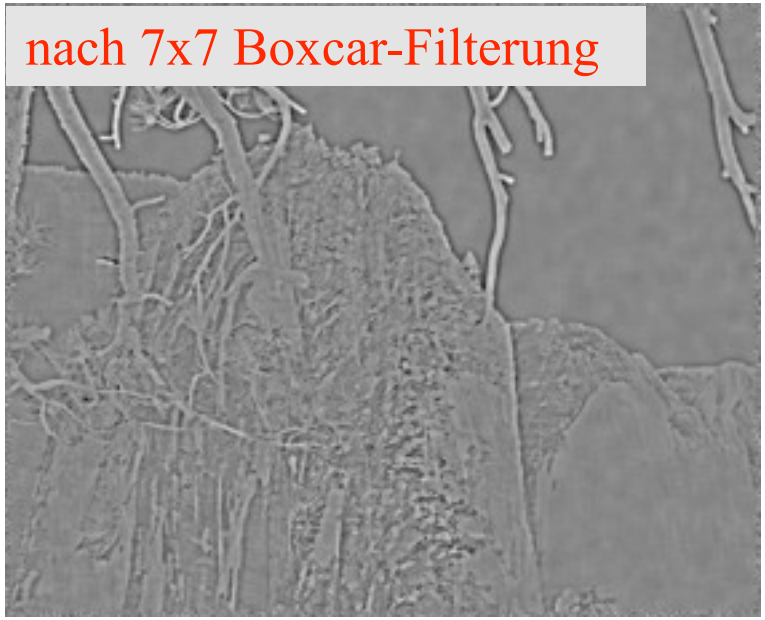
Verhalten an Kanten



|f(m,n)-g(m,n)|



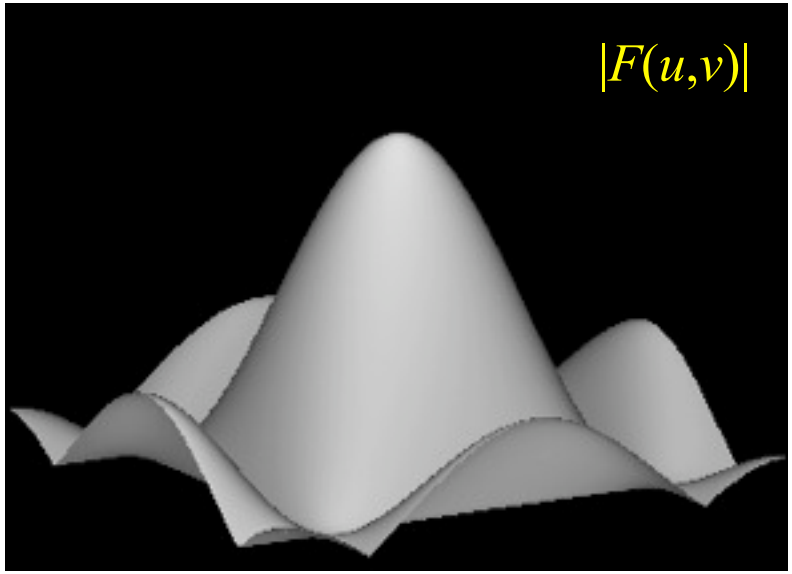
nach 3x3 Boxcar-Filterung

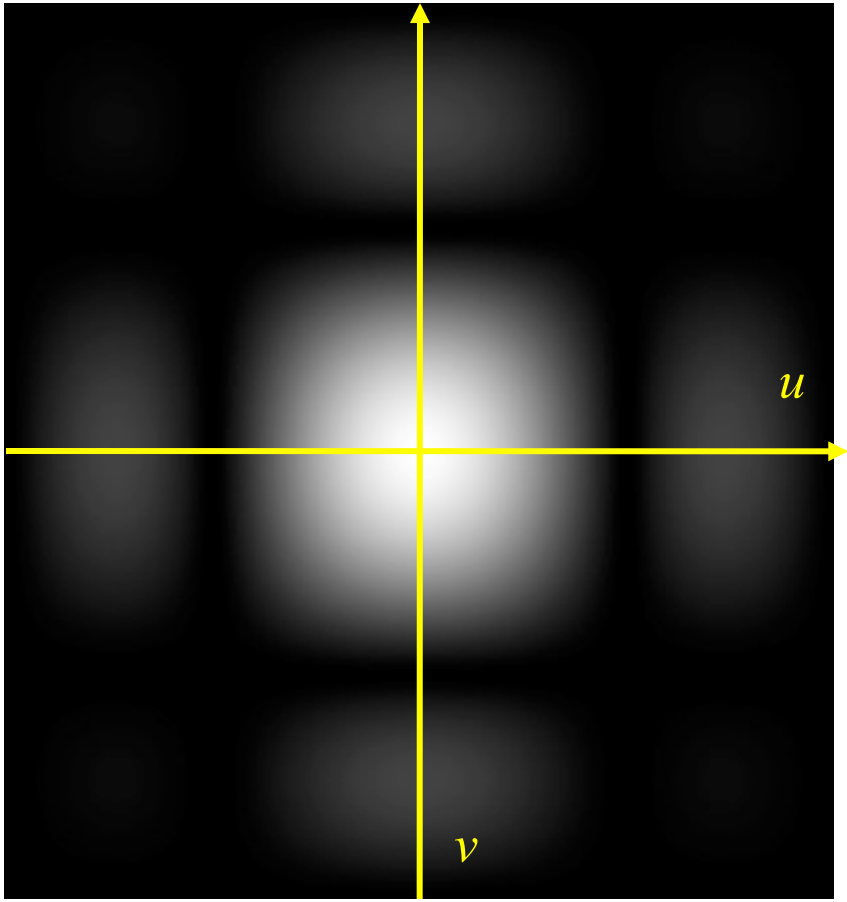


nach 7x7 Boxcar-Filterung

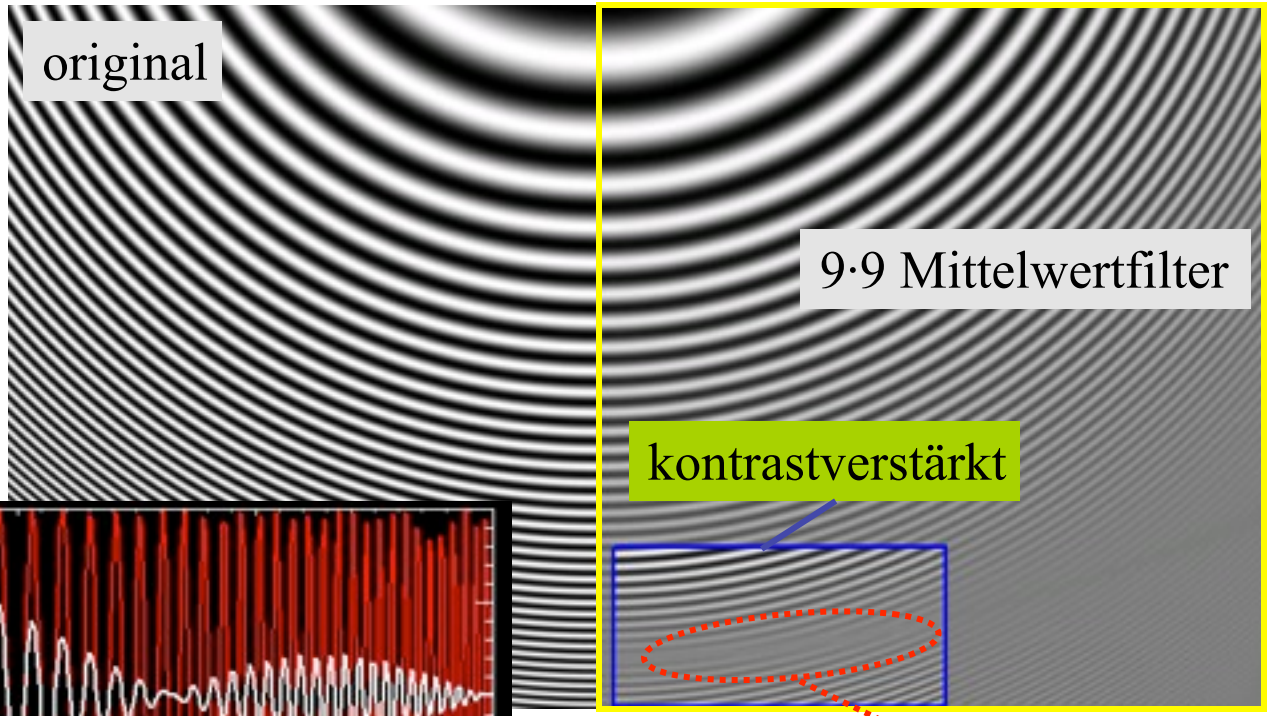
Richtungsabhängigkeit des Mittelwertfilters

Transferfunktion (Repräsentation im Frequenzraum).

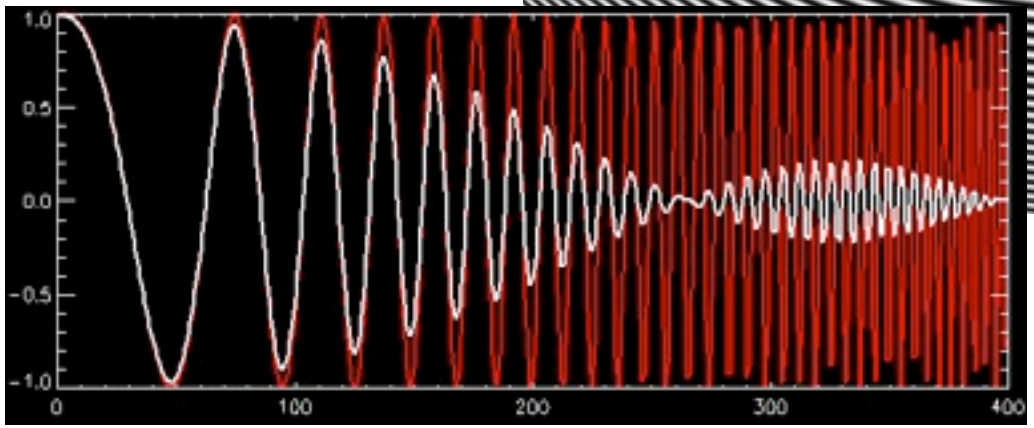




Auswirkungen



Bildzeile
rot: vor der Filterung
weiß: nach Filterung

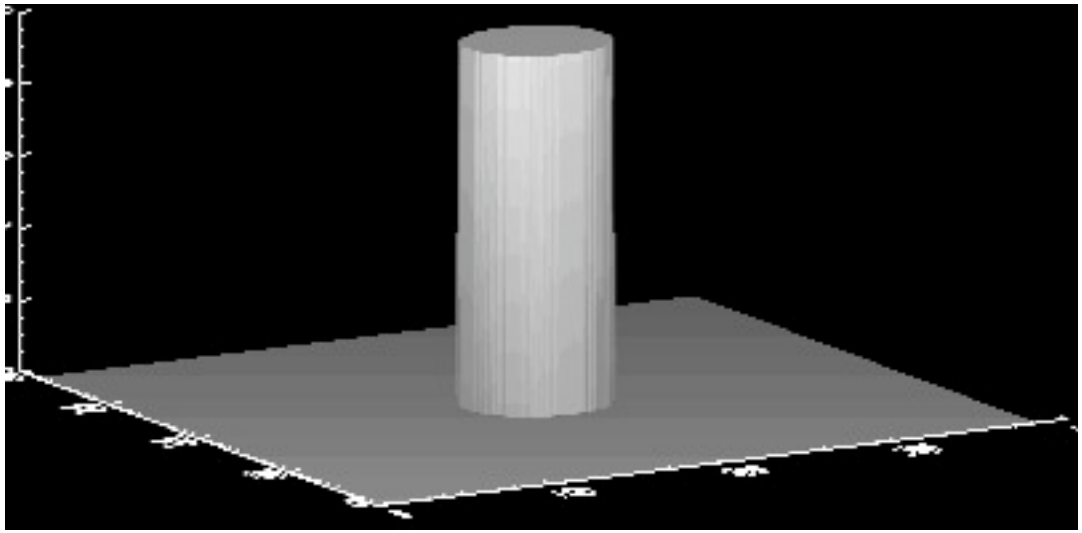


Frequenzraumfilterung

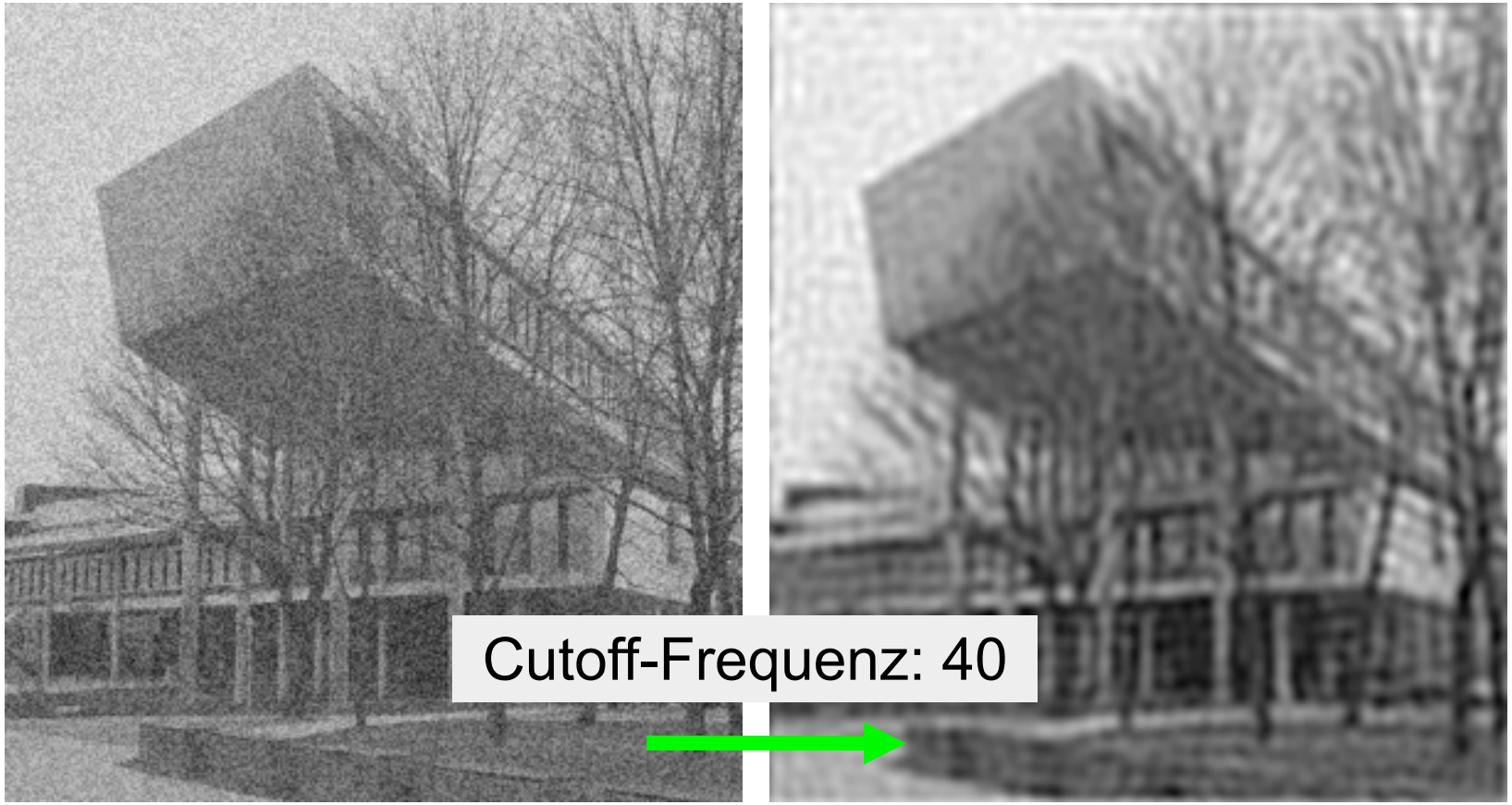
- Filter im Frequenzraum so entwickeln, dass die Artefakte nicht auftauchen können.
- Ideales Tiefpassfilter

$$H_{F_{\max}}(u,v) = \begin{cases} 1 & \text{, falls } u^2 + v^2 \le F_{\max}^2 \\ 0 & \text{, sonst.} \end{cases}$$

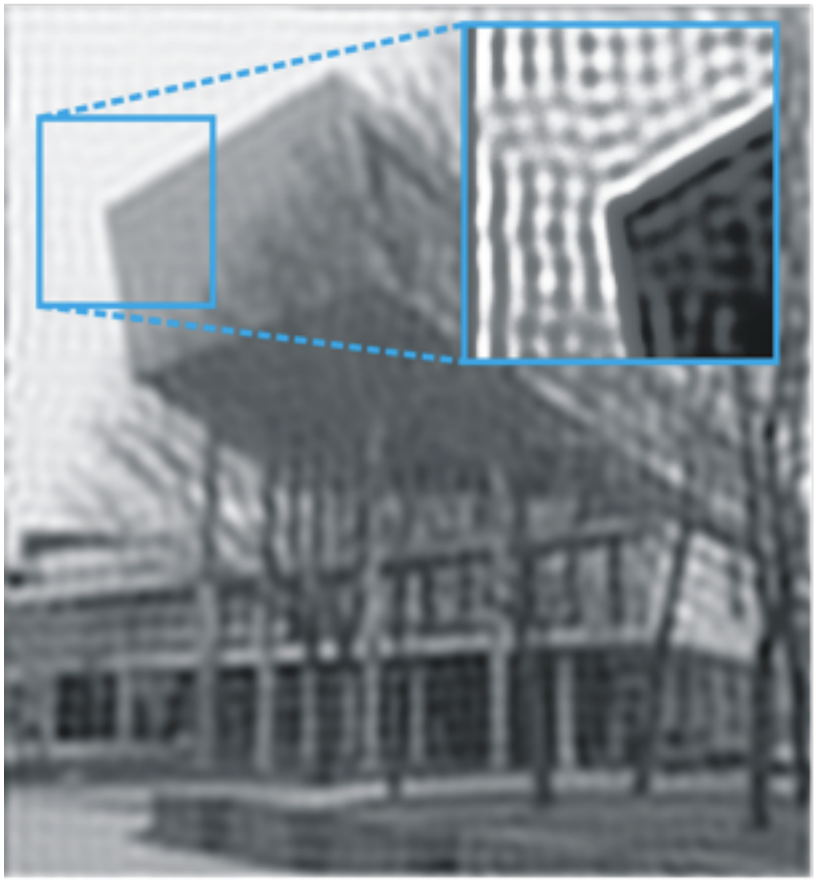
F_{max} – Cut-Off-Frequenz

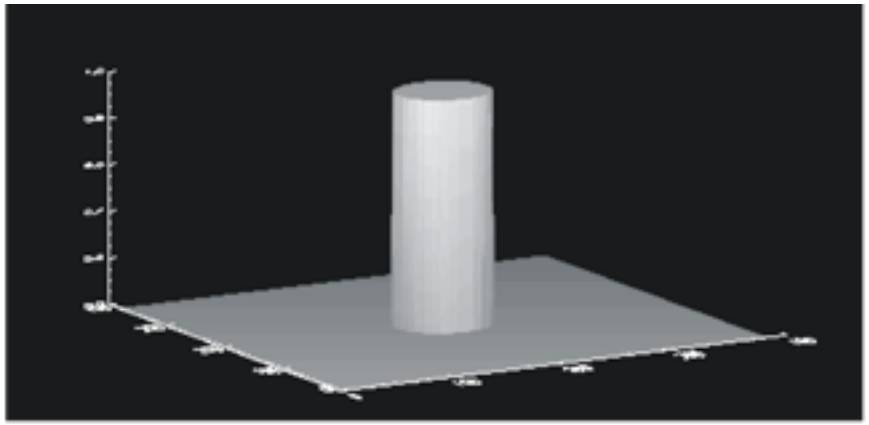


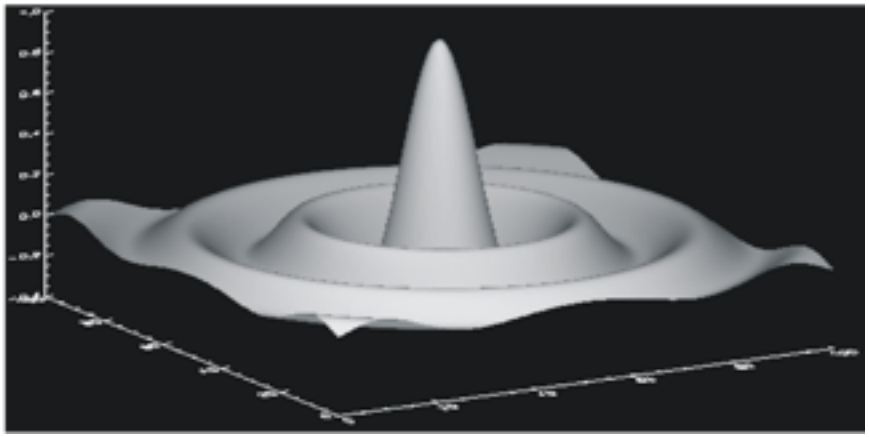
Tiefpassfilter zur Rauschunterdrückung



Ringing-Artefakt



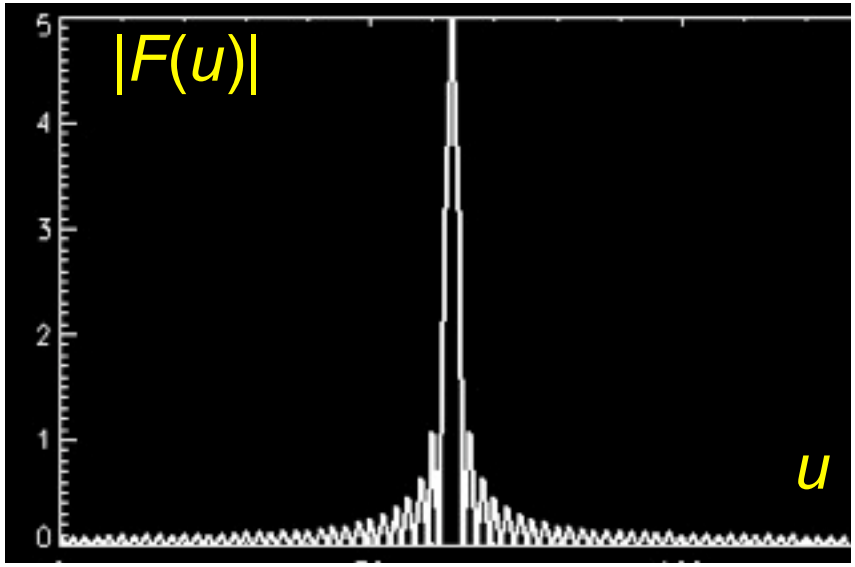




Ringing-Artefakt



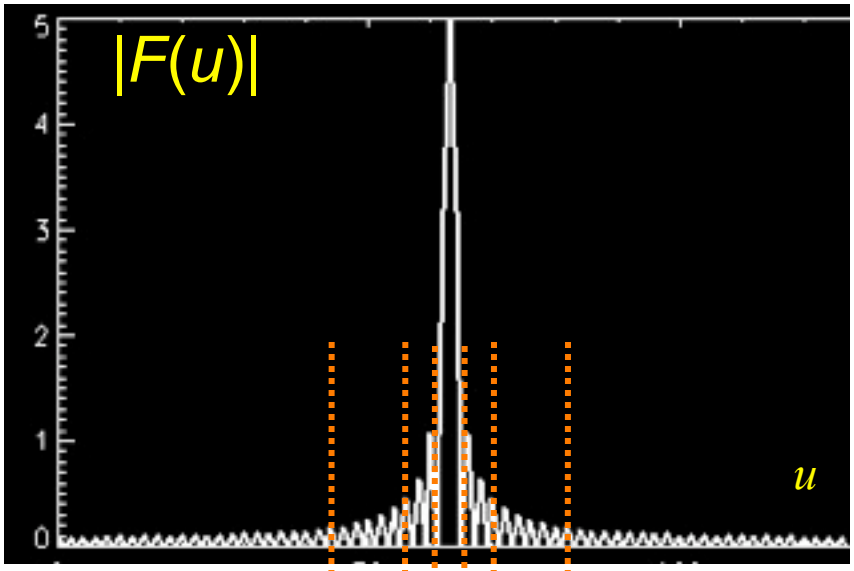
Bildzeile

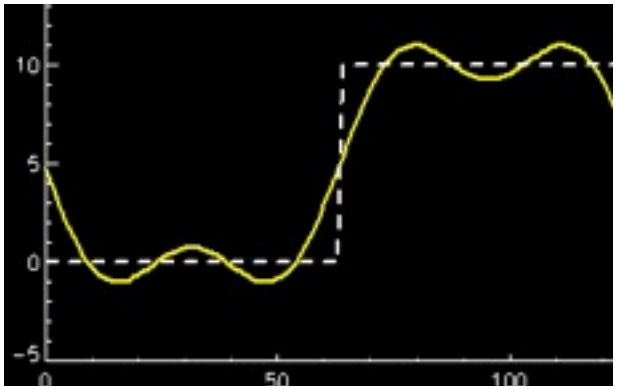


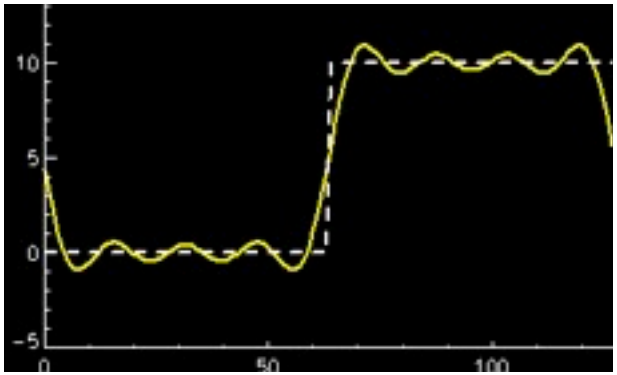
Fourier-transformierte Zeile

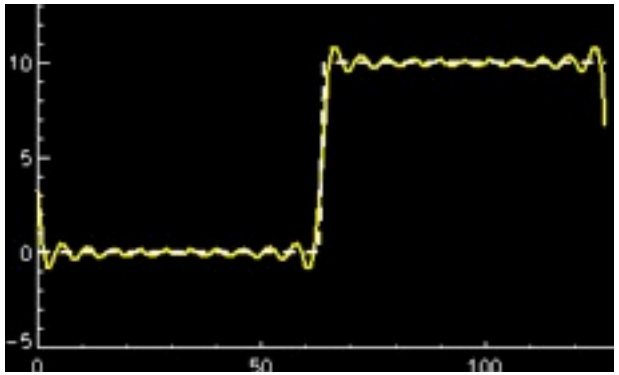
Das Ringing-Artefakt entsteht, weil scharfe Kanten durch Wellen **aller** Frequenzen beschrieben werden.

Filterung









Fourier-transformierte Zeile

Butterworth-Filter

Frequenzen werden nicht gelöscht, sondern nur abgeschwächt.

Tiefpass-Filter:

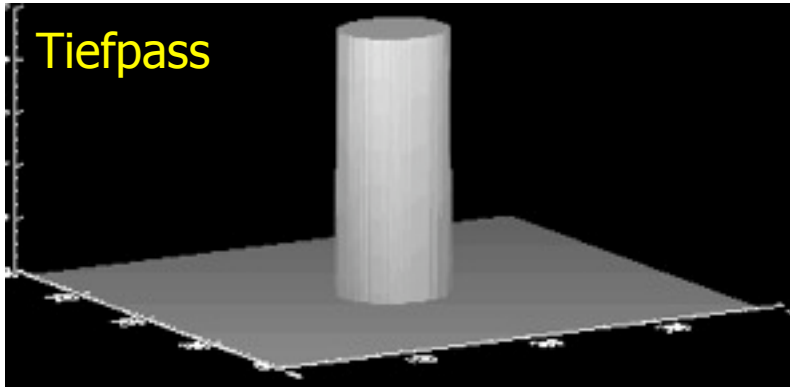
$$H(u,v) = \frac{1}{1 + \left(D(u,v)/D_0\right)^{2n}}$$

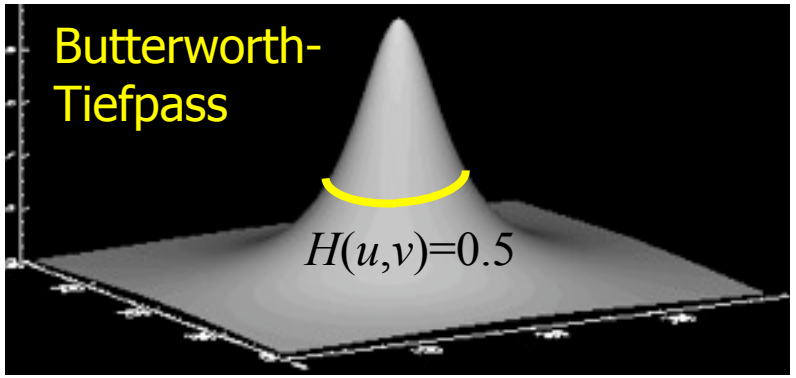
Hochpass-Filter:

$$H(u,v) = \frac{1}{1 + \left(D_0/D(u,v)\right)^{2n}}$$

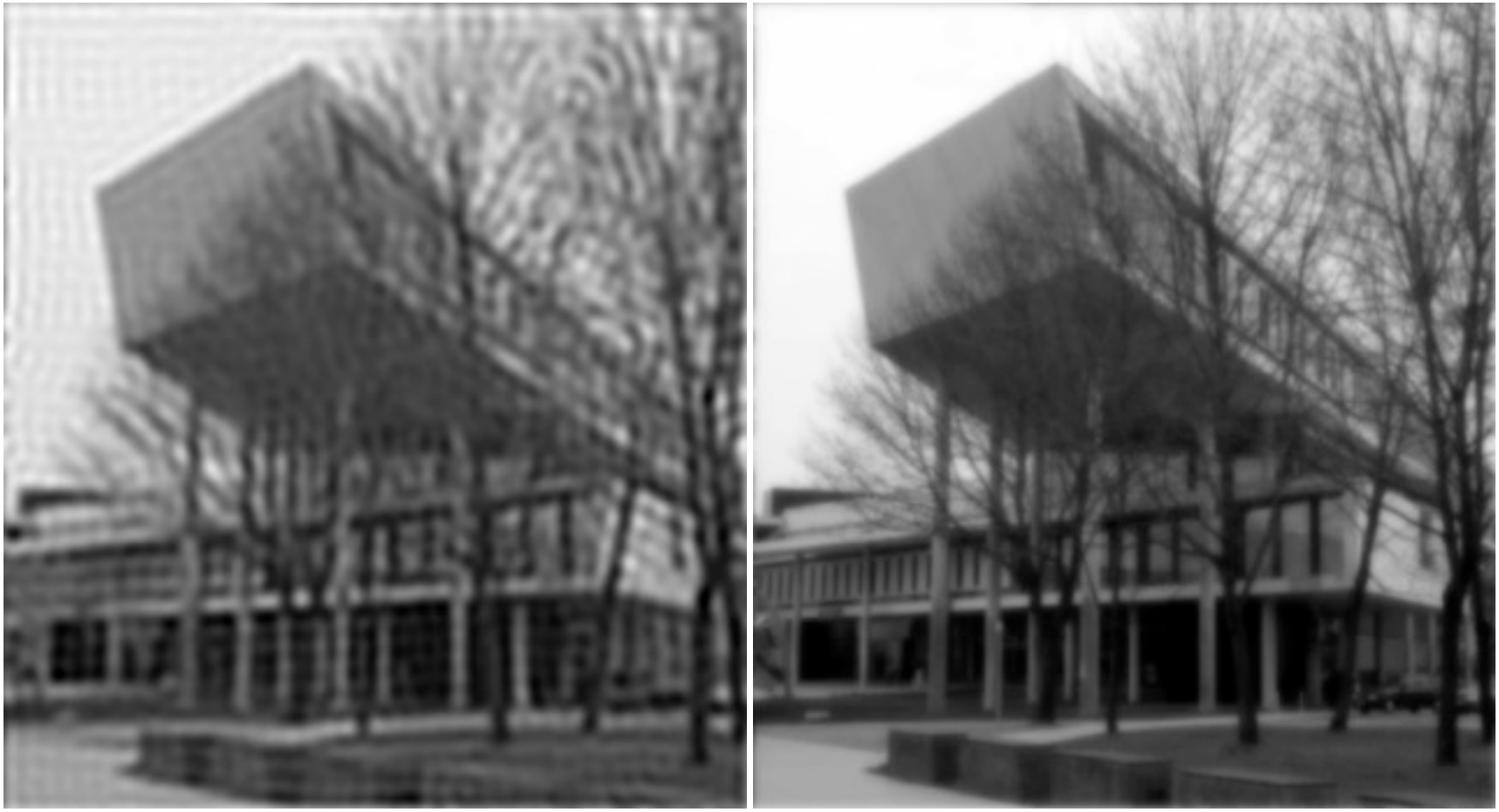
D_0: Cutoff-Frequenz,

$D(u,v)$: Frequenz, d.h. Abstand vom Ursprung





Butterworth vs. Einfacher Tiefpass



Binomialfilter

Eindimensionaler Binomialfilter $B^p = [1\ 1]*[1\ 1]*...*[1\ 1]$ (p-mal):

$B^0 = 1^{-1} \cdot [1]$

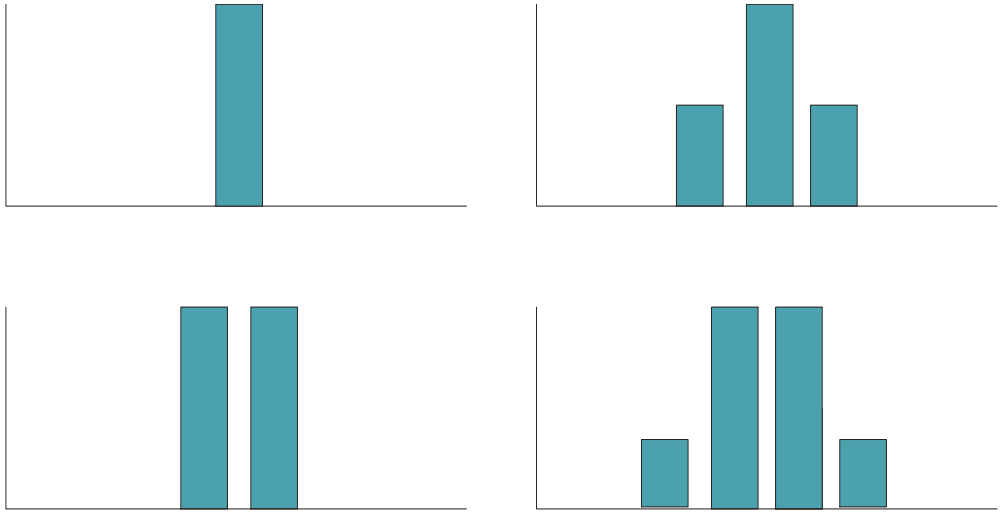
$B^1 = 2^{-1} \cdot [1\ 1]$

$B^2 = 4^{-1} \cdot [1\ 2\ 1]$

$B^3 = 8^{-1} \cdot [1\ 3\ 3\ 1]$

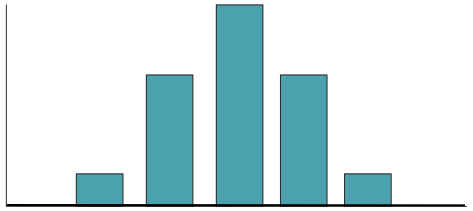
$B^4 = 16^{-1} \cdot [1\ 4\ 6\ 4\ 1]$

...



Zweidimensionaler Binomialfilter $\mathbf{B}^p = B^p * (B^p)^T$:

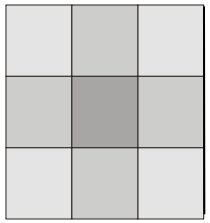
$$\mathbf{B}^2 = 4^{-1} \cdot [1\ 2\ 1]^T \cdot 4^{-1} \cdot [1\ 2\ 1] = 16^{-1} \cdot \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$



Zweidimensionale Binomialfilter

$$\mathbf{B^2} = 1/16 \cdot [1\ 2\ 1]^T \cdot [1\ 2\ 1] = 1/16 \cdot$$

1	2	1
2	4	2
1	2	1

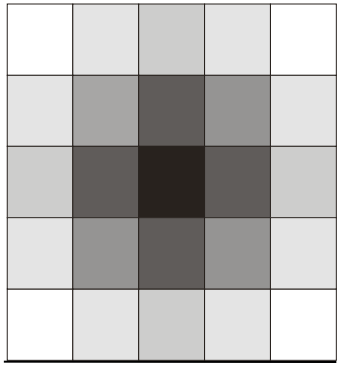


$$\mathbf{B^3} = 1/64 \cdot [1\ 3\ 3\ 1]^T \cdot [1\ 3\ 3\ 1] = 1/64 \cdot$$

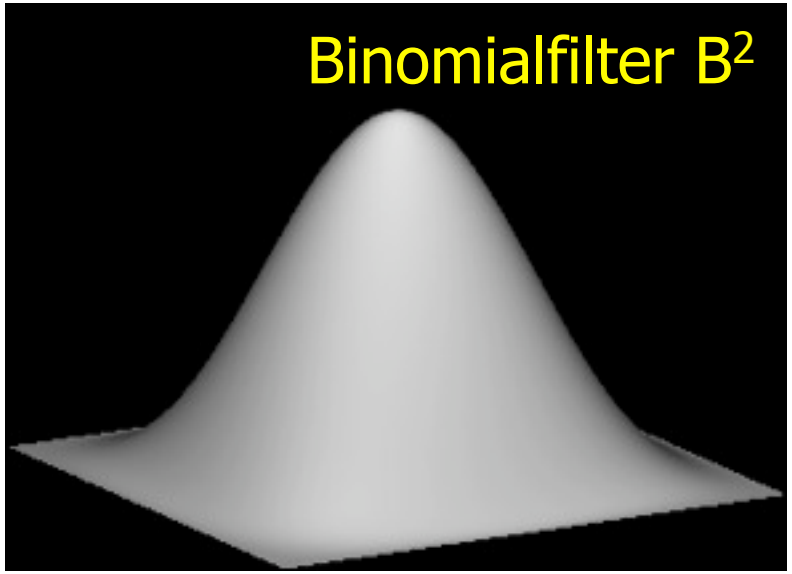
1	3	3	1
3	9	9	3
3	9	9	3
1	3	3	1

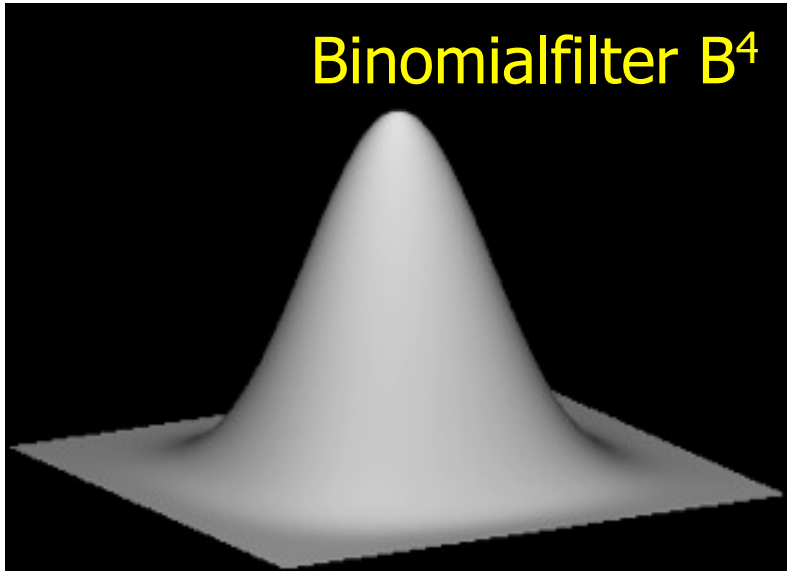
$$\mathbf{B^4} = 1/256 \cdot$$

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



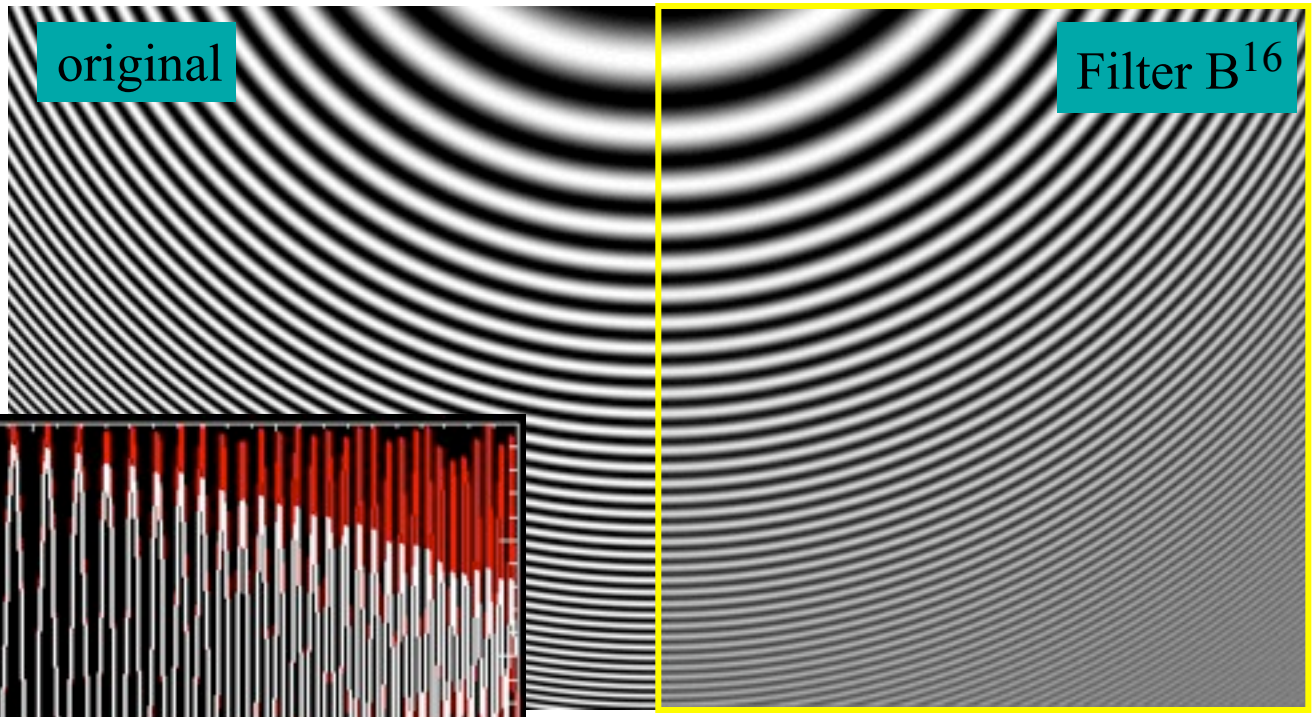
Transferfunktion des Binomialfilters



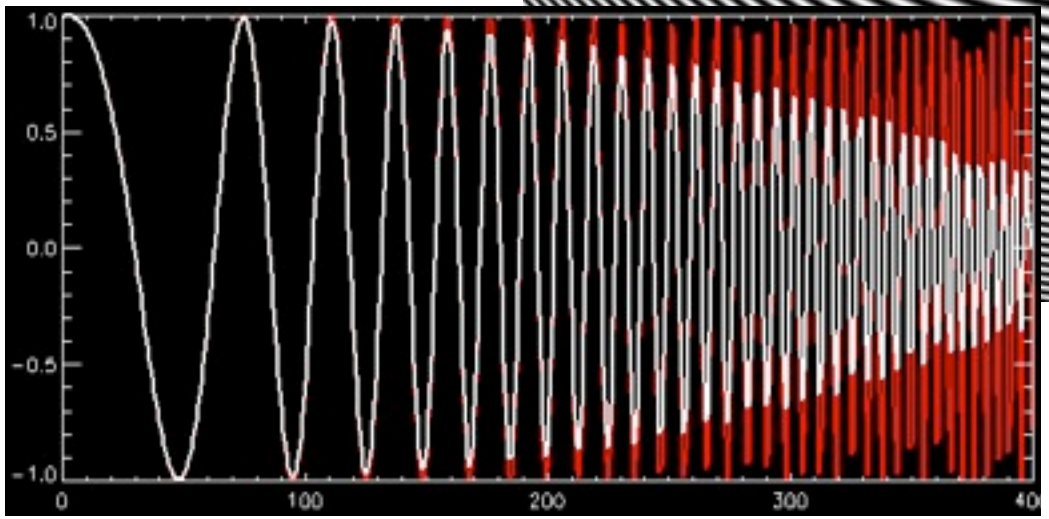


Weniger Artefakte an Kanten sind zu erwarten.

Filterresultate des Binomialfilters



Bildzeile
rot: vor der Filterung
weiß: nach Filterung



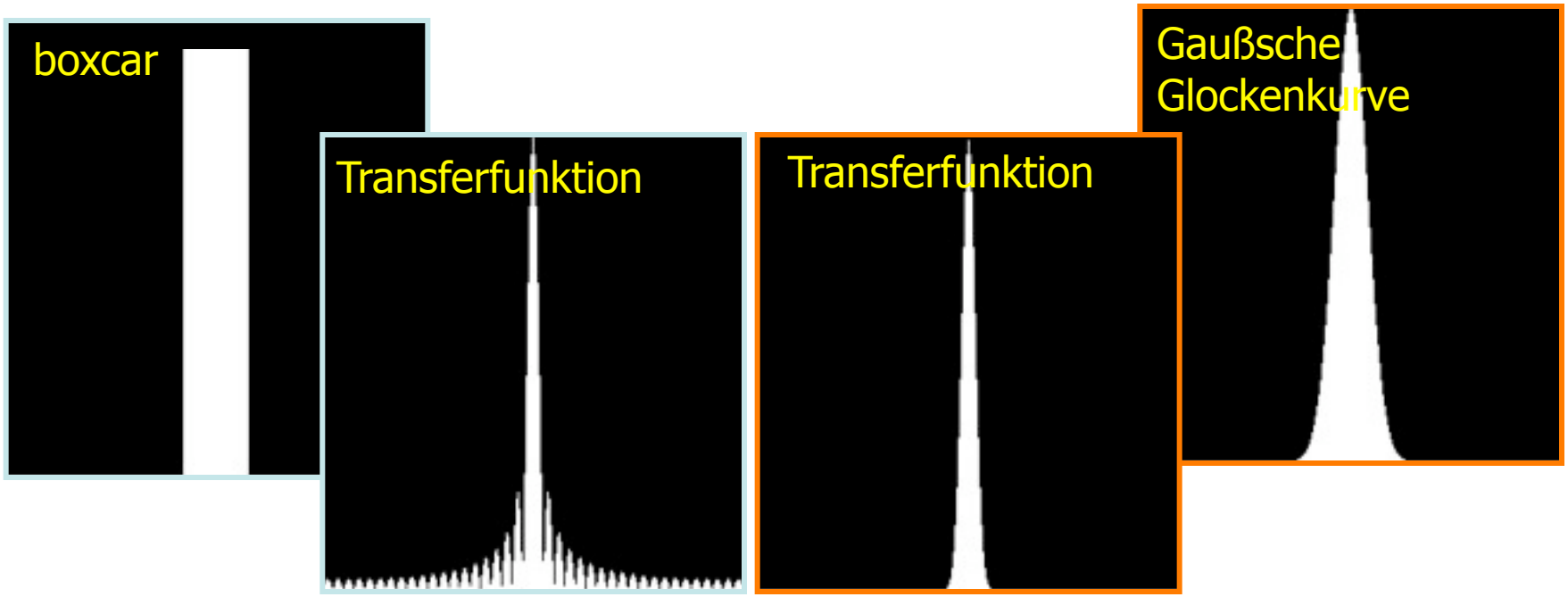
Butterworth-Filter / Binomialfilter

- Ideales Tiefpassfilter: kompakter Träger im Frequenzraum, aber artefakt-verursachende Ortsraumrepräsentation
- ► Butterworth-Filter: kontrolliert monoton fallende Funktion im Frequenzraum, deren Ortsraumrepräsentation ebenfalls monoton fällt.

- Mittelwertfilter: kompakter Träger im Ortsraum, aber artefakt-verursachende Frequenzraumrepräsentation
- ► Binomial-Filter: monoton fallende Funktion mit kompaktem Träger im Ortsraum, deren Frequenzraumrepräsentation monoton fällt.

Binomialfilter und Gaußfunktion

- Für immer größere Filterkerne nähert sich das Binomialfilter der Gaußschen Glockenkurve an.
- Der Betrag der Transferfunktion einer solchen Funktion ist wieder eine Gaußsche Glockenkurve.



$$\text{gauss}(x,y) = [\sigma \cdot \sqrt{2\pi}]^{-1} \cdot \exp\left[-(x^2+y^2)/(2\sigma^2)\right]$$

Filterung mit 2D Gaußfilter



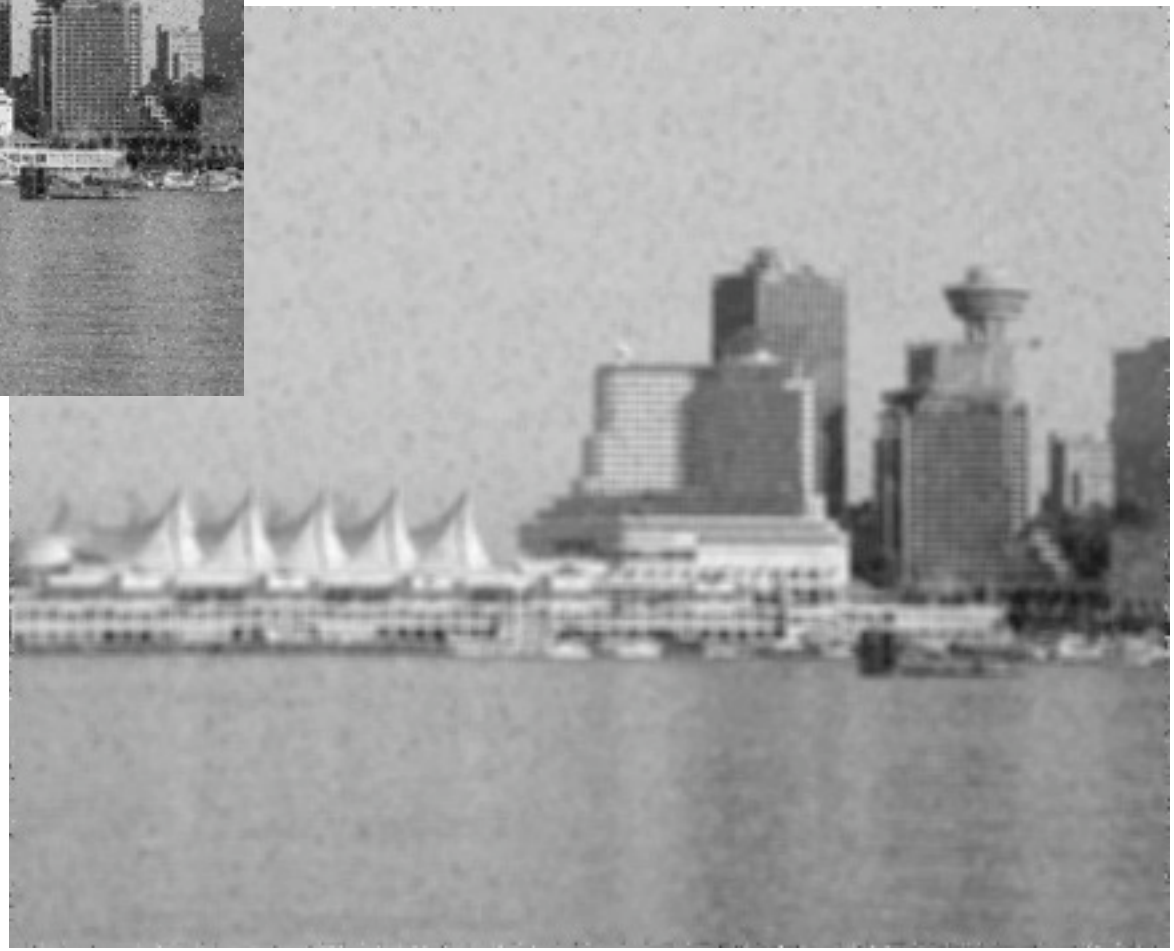
Die Gaußfunktion ist separabel, so dass die Filterung durch zwei 1D Konvolutionen erfolgen kann.





Grenzen





Sogenanntes Impulsrauschen (Salt & Pepper Noise) kann nicht entfernt werden.

Was sollten Sie gerade gelernt haben?

- Rauschunterdrückung durch Schätzung des Erwartungswerts der Bildfunktion
- Schätzung des Erwartungswerts = zeitliche oder räumliche Integration
- Filter im Orts- und Frequenzraum
- Artefakte bei Orts-/Frequenzraumfiltern

Literatur

- Klaus D. Tönnies: "Grundlagen der Bildverarbeitung", ISBN 3-8273-7155-4
- http://www.pearson-studium.de/main/main.asp?page=booksites/selectchapter&isbn=3827371554&PSZielgruppe=Student

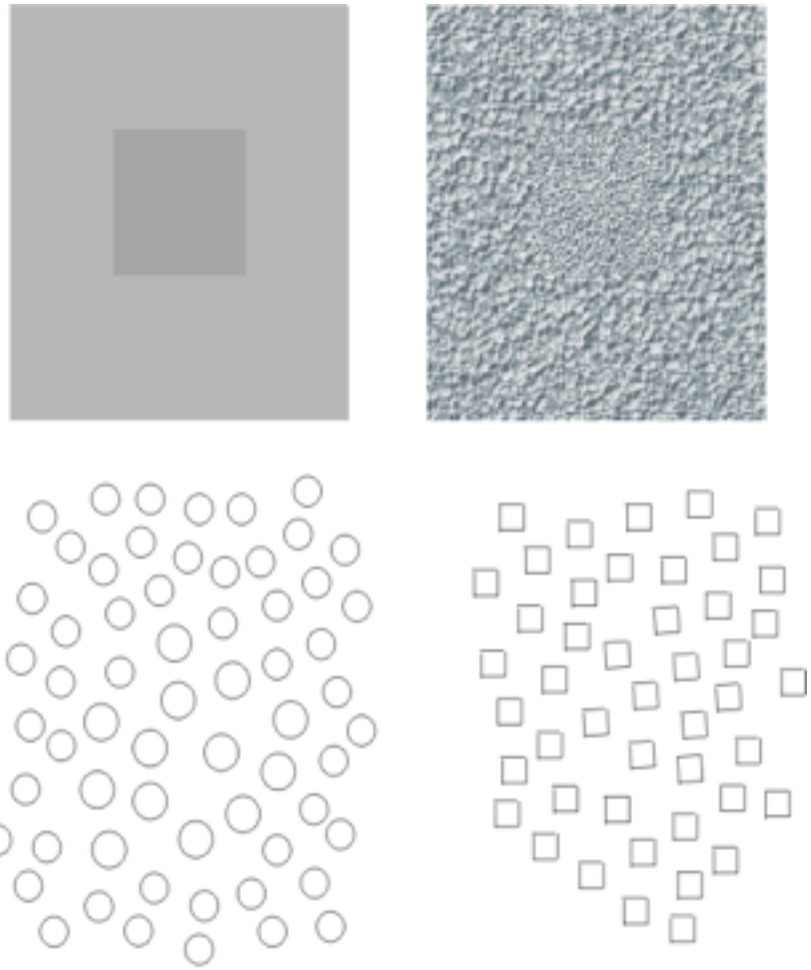
Computergraphik 1
2. Teil: Bildverarbeitung

Bildverbesserung 2

Themen jetzt

- Lineare Bildverbesserung
  - Kantenoperatoren
  - Kantenattribute
  - Kanten und Rauschen
- Nichtlineare Filter
  - Rangordnungsfilter
  - Diffusionsfilter
- Segmentierung
  - Schwellwertbasiert
  - Shadingkorrektur

Was sind Kanten?



Diskontinuitäten zwischen Regionen.

Problem:

Die Region selbst und damit ihr Homogenitätskriterium muss bekannt sein, um Diskontinuität definieren zu können.

Homogenität einer Funktion:

- Grauwerte
- Textur
- Form

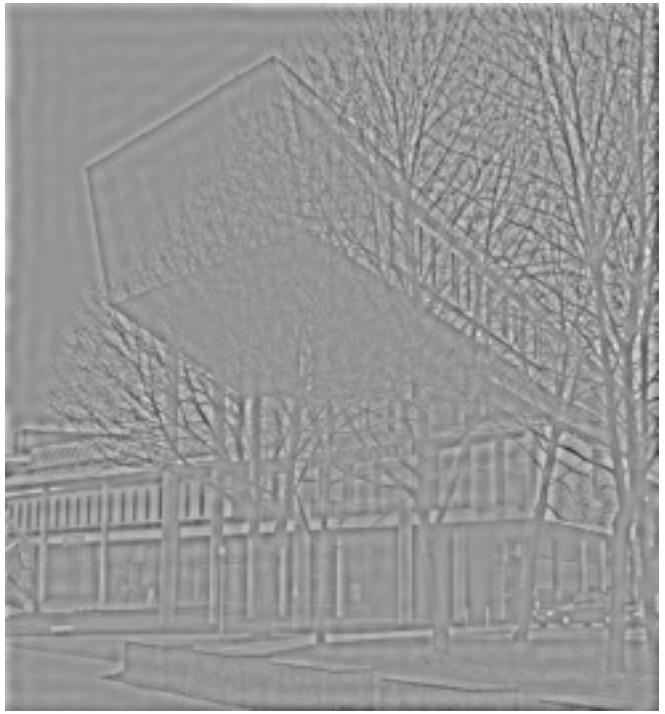
Kantenhervorhebung durch Frequenzraumfilterung

- Kanten weisen mehr hochfrequente Anteile auf als homogene Gebiete

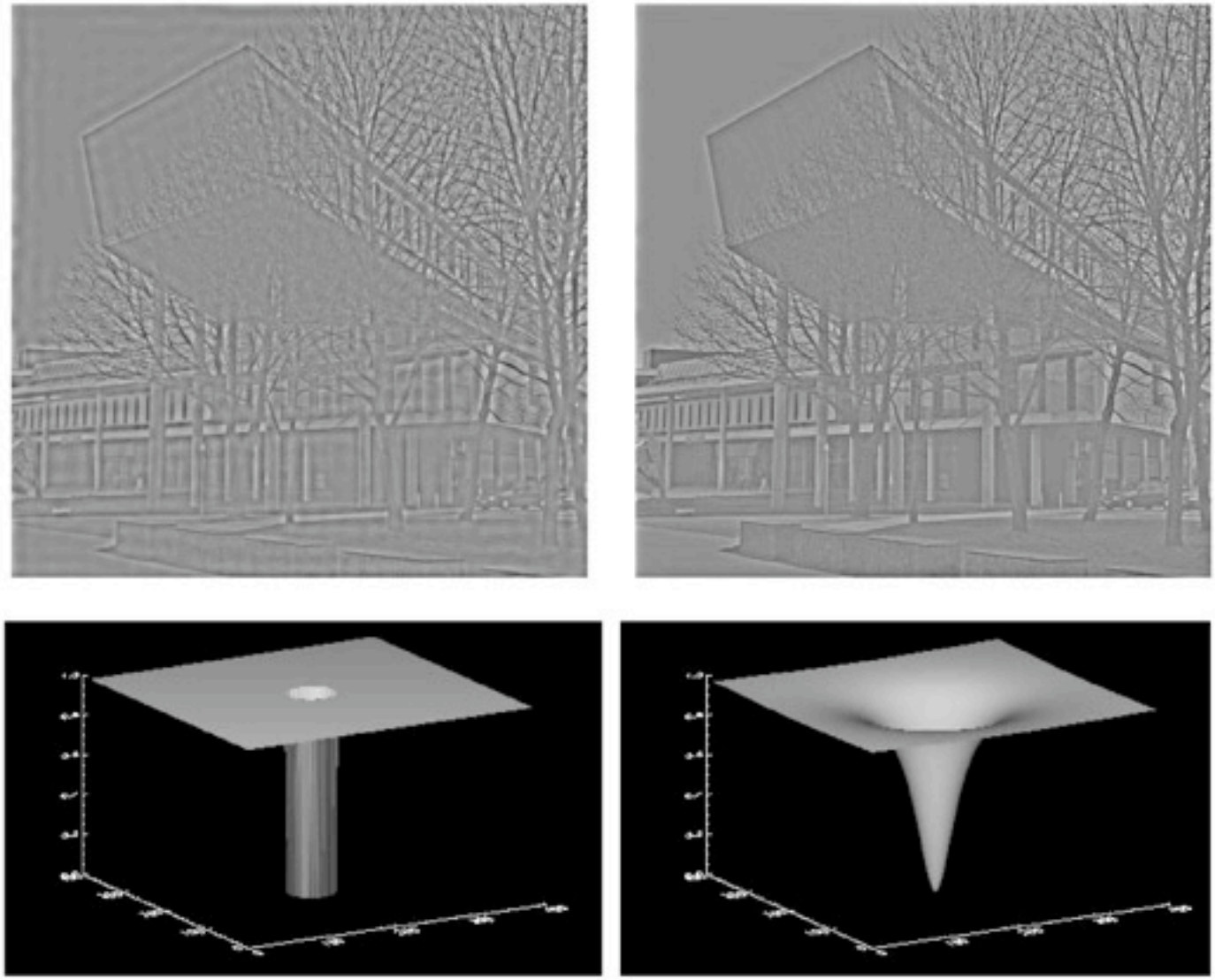
► Hochpassfilterung

$$H_{F_{\max}}(u,v) = \begin{cases} 1 & , \text{falls } u^2 + v^2 \geq F_{\max}^2 \\ 0 & , \text{sonst.} \end{cases}$$





Filterung mit Idealem Hochpass- (links) und mit Butterworth-Hochpassfilter (rechts)



Butterworth-Hochpassfilter

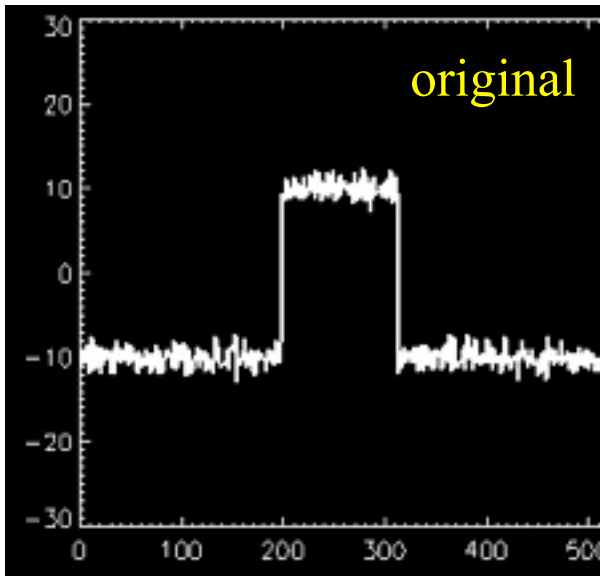


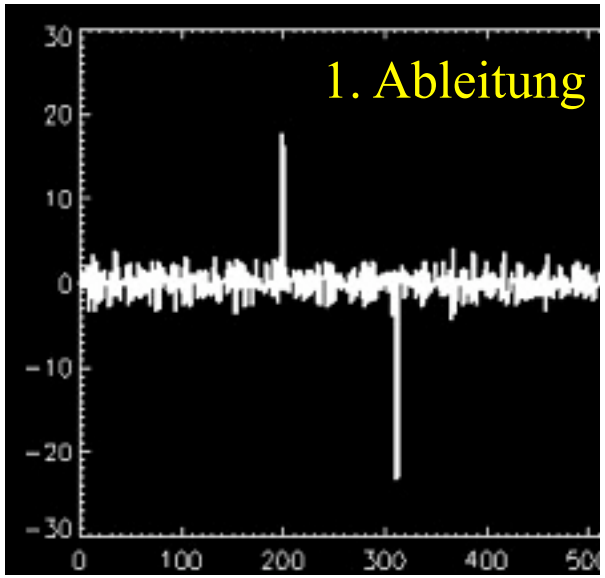
$$H_{F_{\max}}(u,v) = \begin{cases} \dfrac{1}{1 + \left(F_{\max}^2 \Big/ \left(u^2 + v^2\right) \right)^k} & u \neq 0 \vee v \neq 0 \\ 0 & \text{sonst} \end{cases}$$

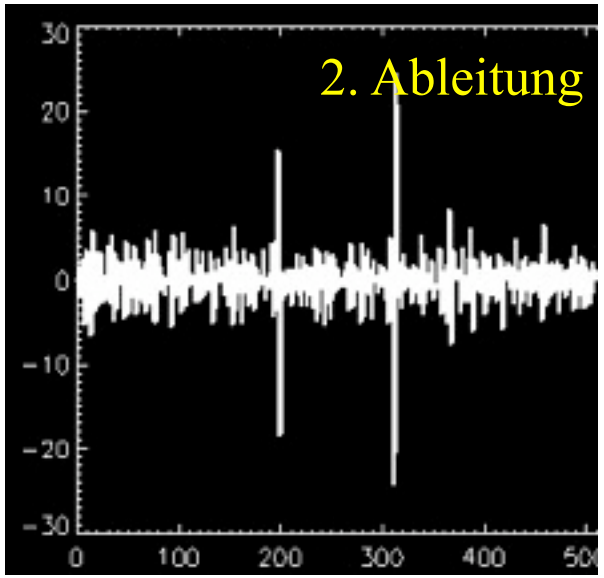


Kanten im Ortsraum

Kanten und Rauschen haben ähnliche Charakteristika im Frequenzraum ➔ Kantendetektor verstärkt Rauschen.





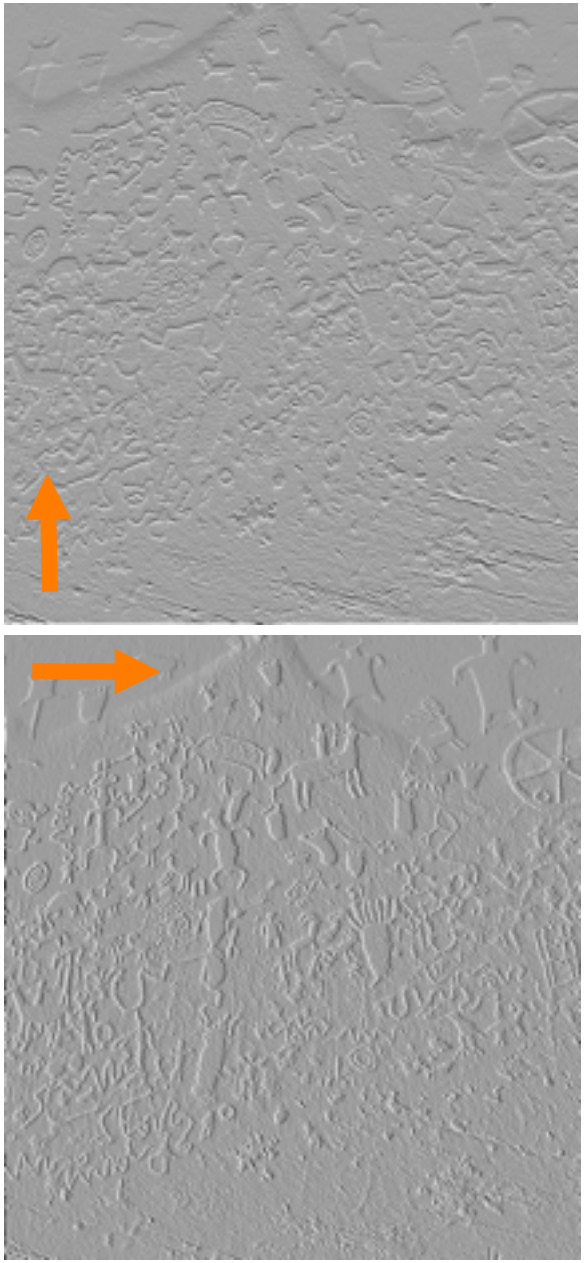


Kanten im 2-D Raum

Differenzbildung in *n*-Richtung



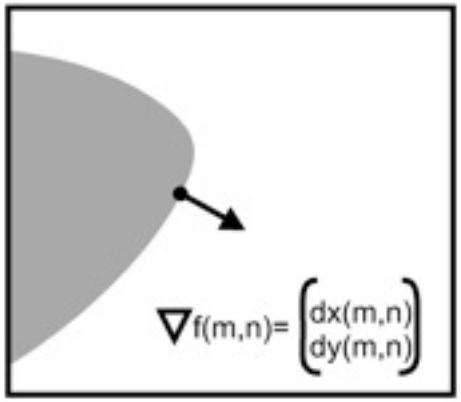
Pseudo-3D Eindruck:



Differenzbildung in *m*-Richtung

Gradientenschätzung über Konvolution

- Gradient im *N*-dimensionalen Raum ist ein *N*-dimensionaler Vektor aus *N* partiellen Ableitungen.
- Jede partielle Ableitung kann durch eine Differenz abgeschätzt werden, die durch Konvolution berechnet werden kann.
- Beispiele:

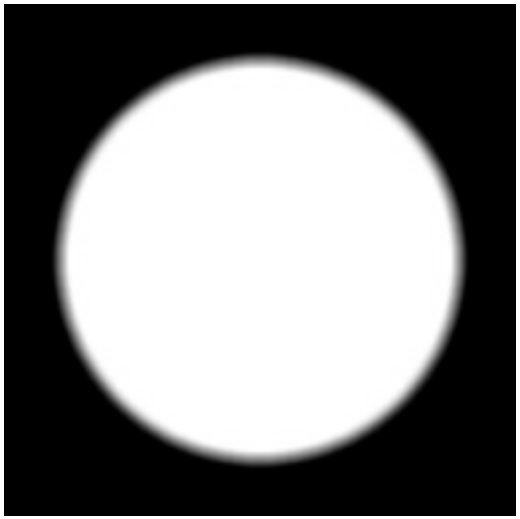


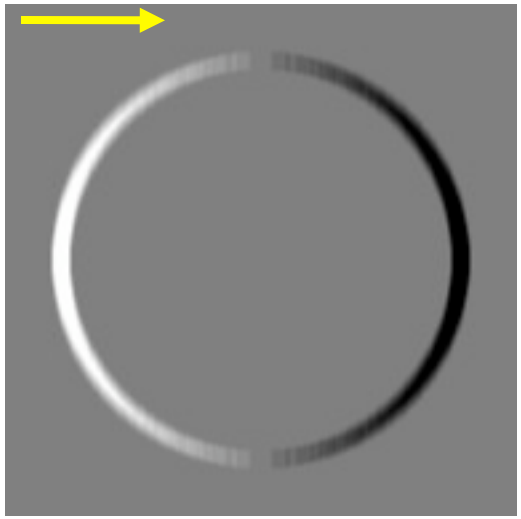
$$\vec{G}(m,n) = \begin{pmatrix} G_x(m,n) \\ G_y(m,n) \end{pmatrix}, \quad G_x(m,n) \approx [f * g_x](m,n), \quad G_y(m,n) \approx [f * g_y](m,n)$$

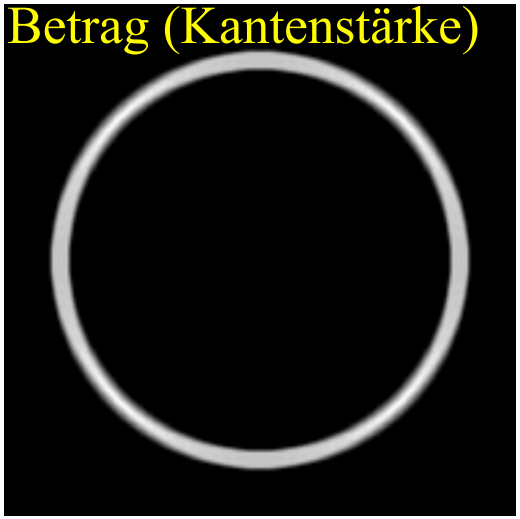
$$g_x = \begin{bmatrix} -1 & 1 \end{bmatrix} \quad g_y = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \qquad g_{R1} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad g_{R2} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$ (Robert's Gradient)

$$g_x = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad g_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$ (symmetrischer Konvolutionskern)

Elemente des Gradienten

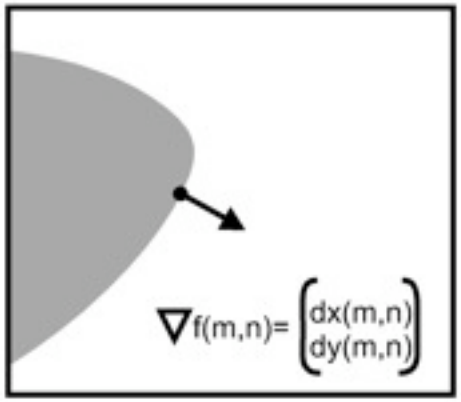


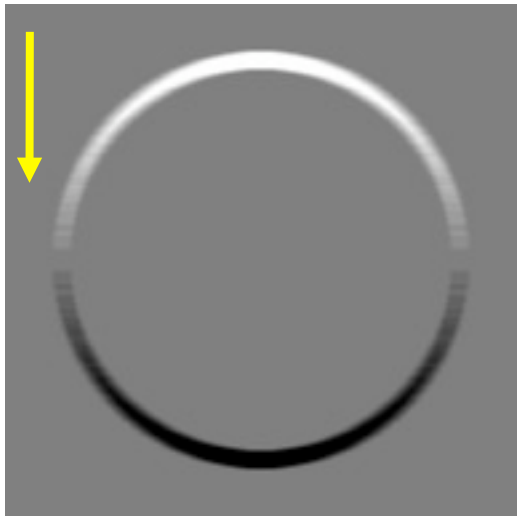




Betrag : sqrt($G_x^2+G_y^2$)

Richtung: $\tan^{-1}(G_y/G_x)$

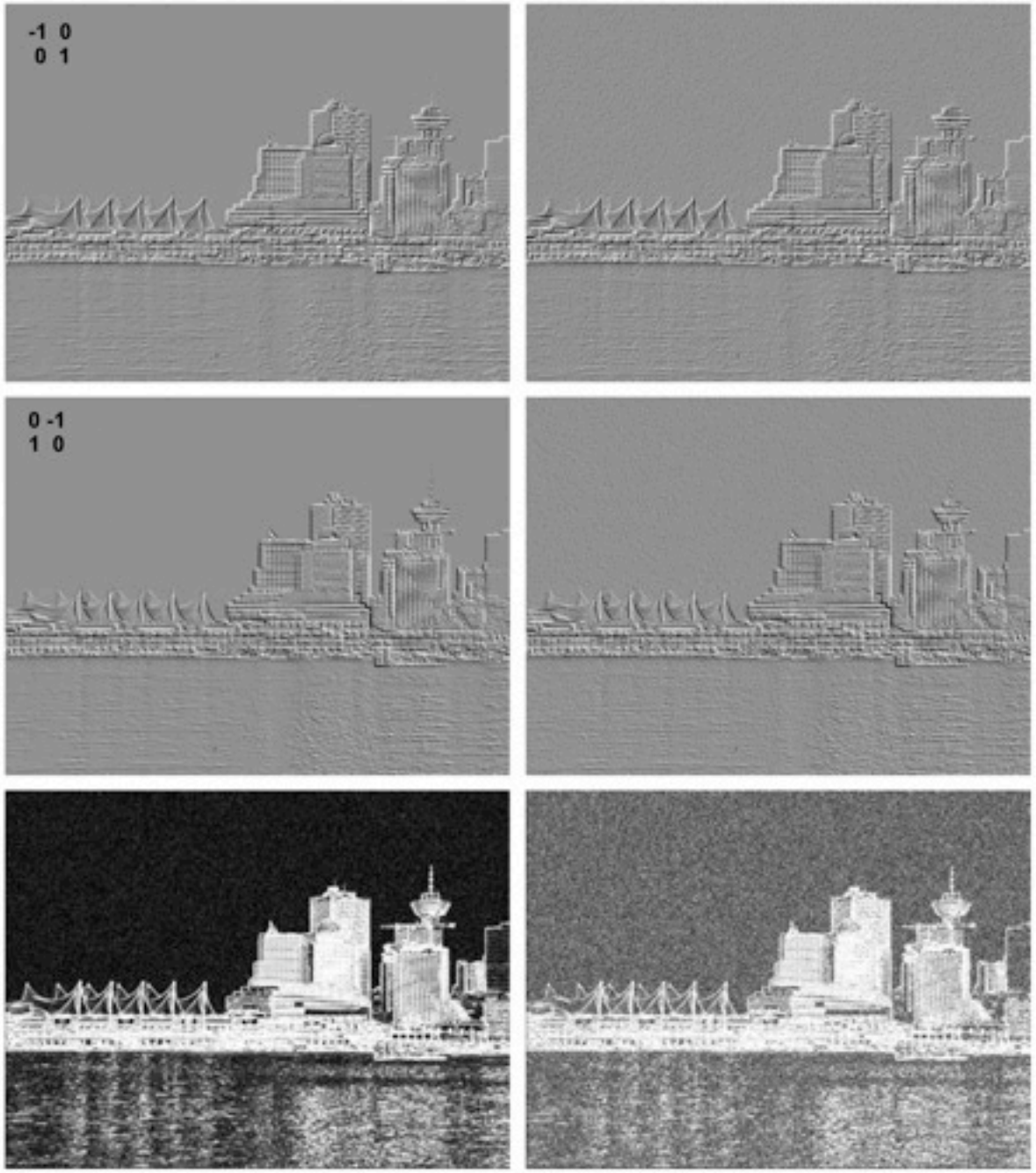




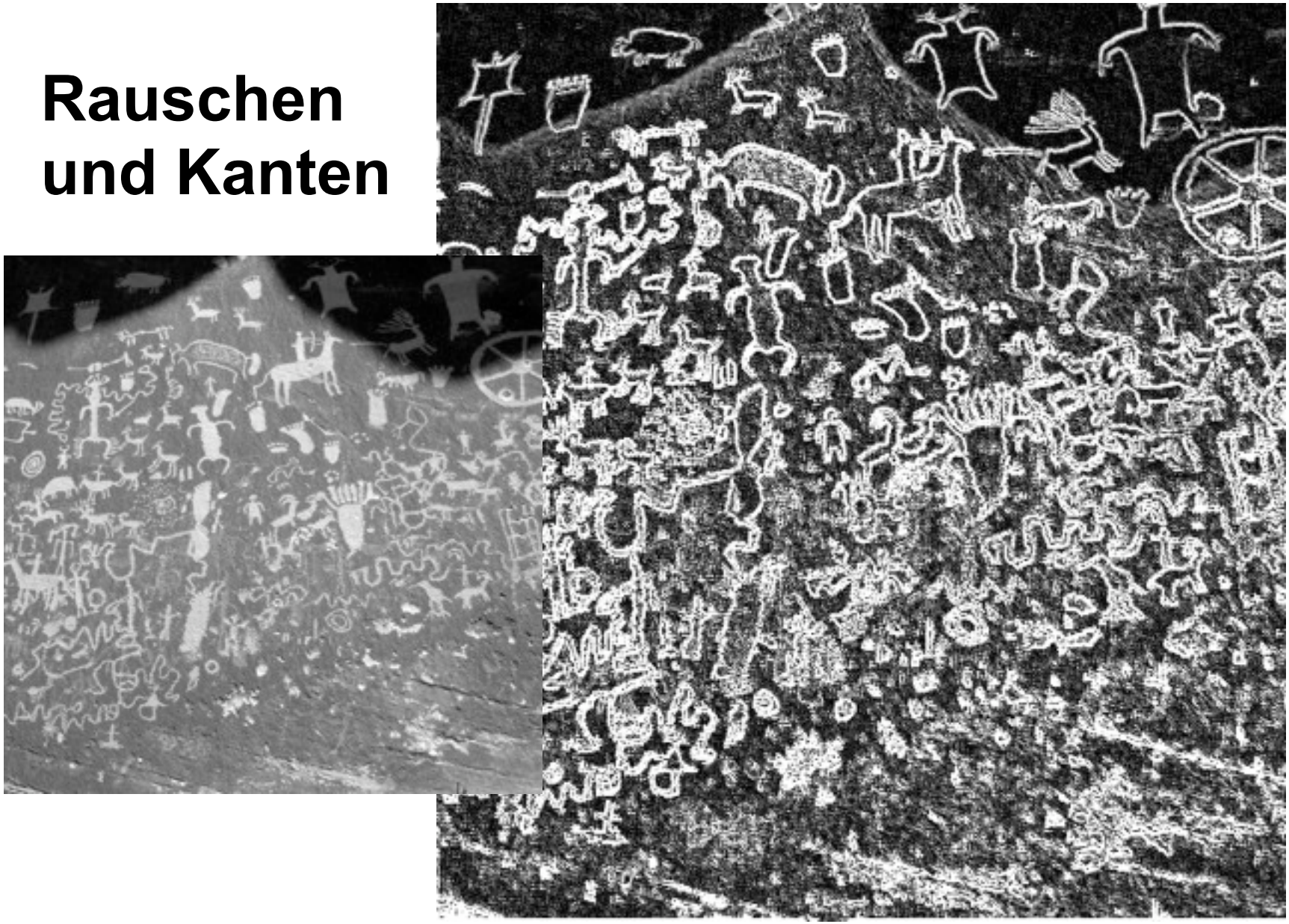


Robert‘s Gradient

Die beiden Komponenten des Robert’s Gradient sowie der Betrag des Gradienten (unten) auf einem rauscharmen (links) und einem verrauschten (rechts) Bild



Rauschen und Kanten

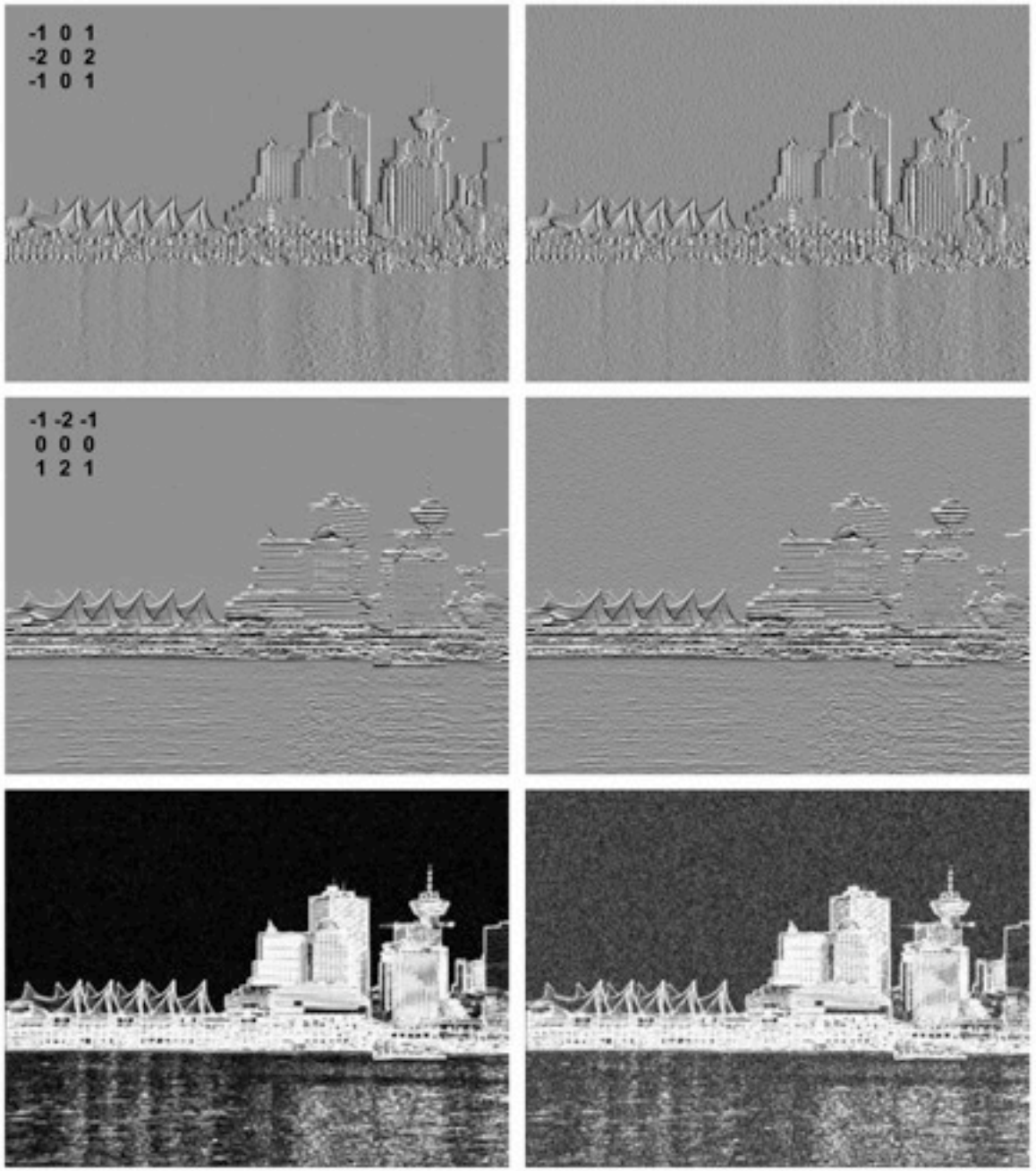


Hintergrundartefakte (Rauschen) werden verstärkt.

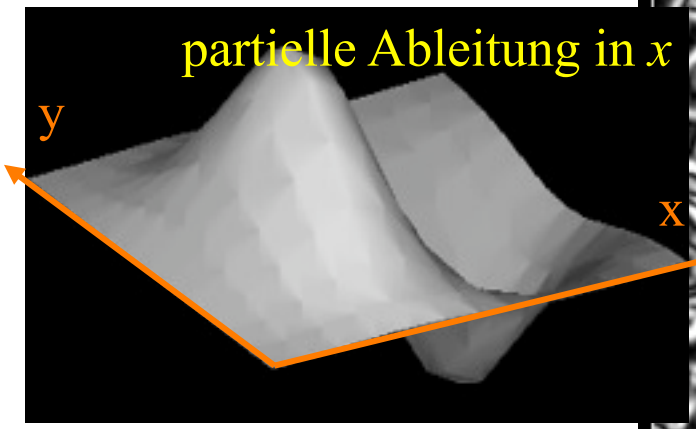
Sobel Operator

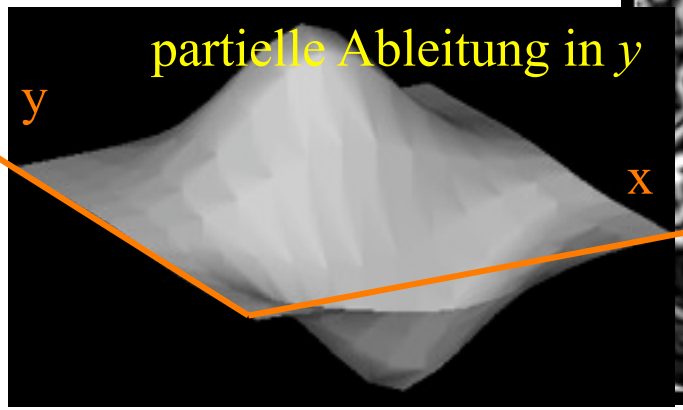
-1	0	1			-1	-2	-1
-2	0	2		und	0	0	0
-1	0	1			1	2	1

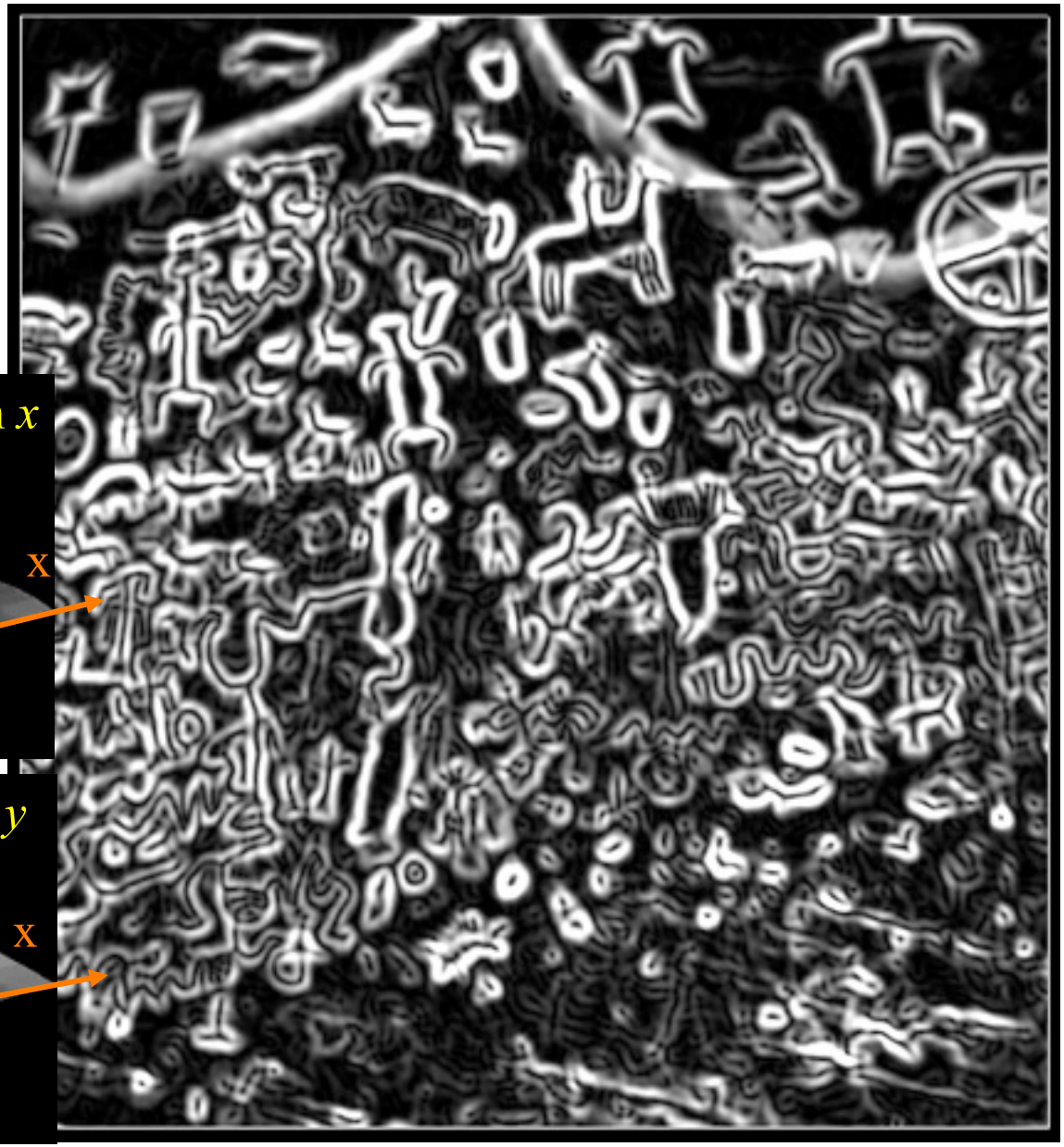
Faltungskern kombiniert Glättung und Differenzierung in einem Operator.



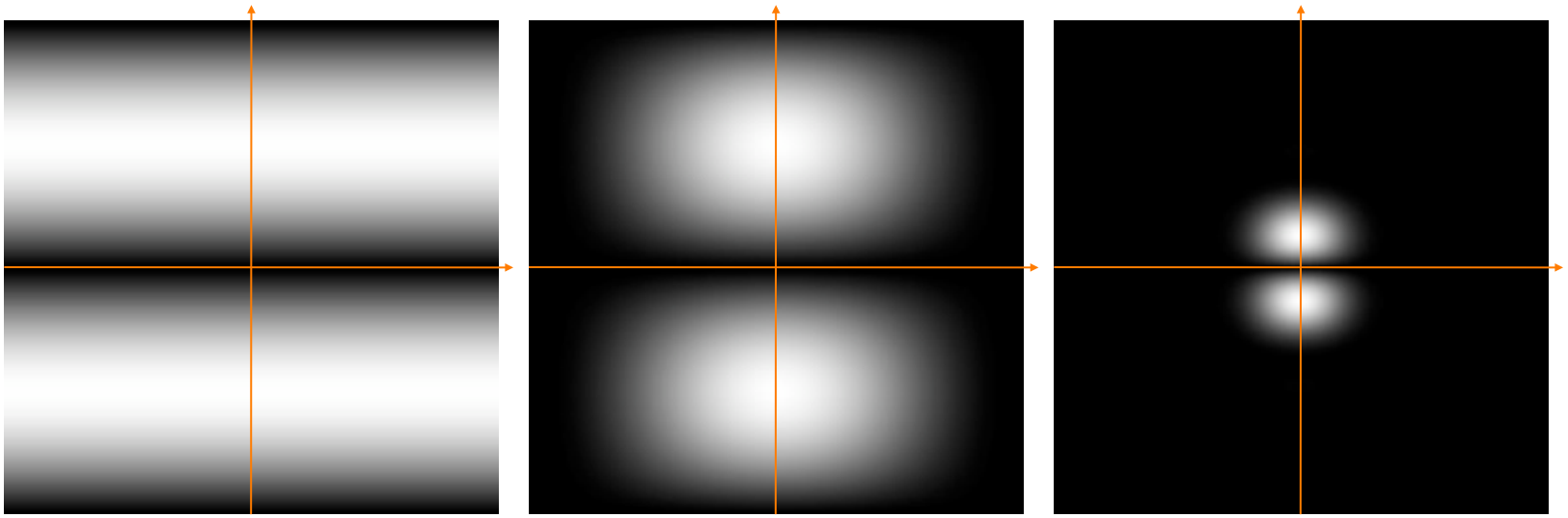
Ableitungen der Gaussfunktion







Vergleich der Differenzfilter im Frequenzbereich



Einfache Differenz

Sobelfilter

Approximation der ersten Ableitung der Gaußfunktion

Sobel vs. Gauß

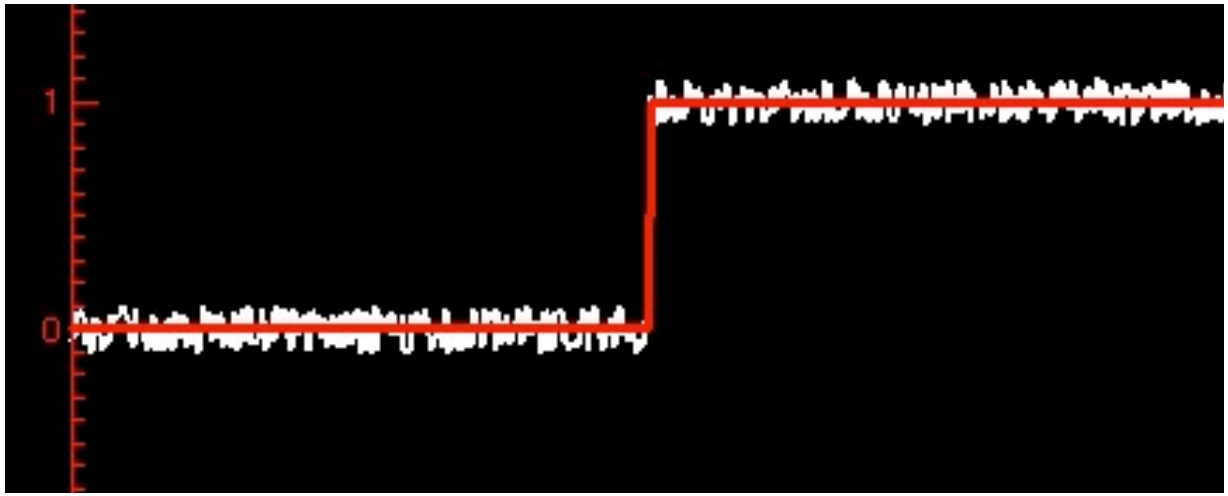




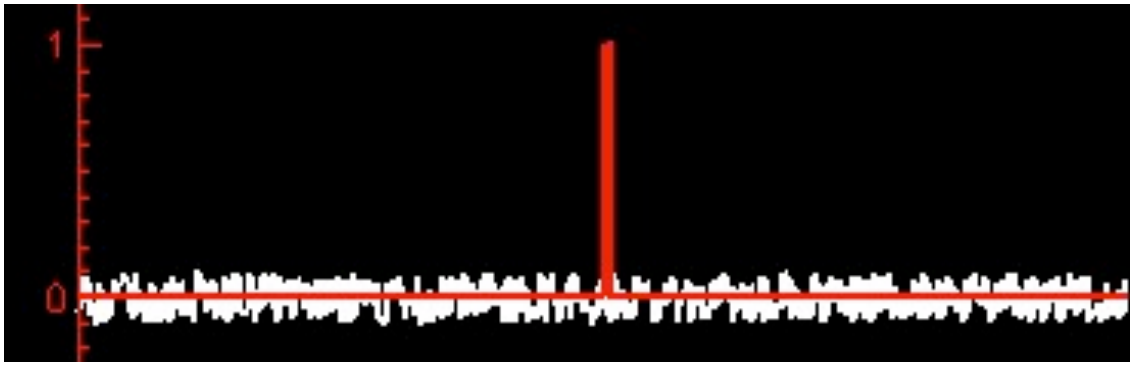


Optimale Kantenfilterung

- Annahme: Bild ist homogen mit Ausnahme von Funktionswertänderungen an idealen Kanten.
- Repräsentation im Frequenzraum
  - homogener Bereich mit Grauwert *g*:
    $F(0,0) = g$ und $F(u,v)=0$ für $u,v \neq 0$
  - Kante: $F(0,0) = 0$ und $F(u,v) \neq 0$ für $u,v \neq 0$
  - Rauschen: $F(u,v) \neq 0$



Kanten und Rauschen



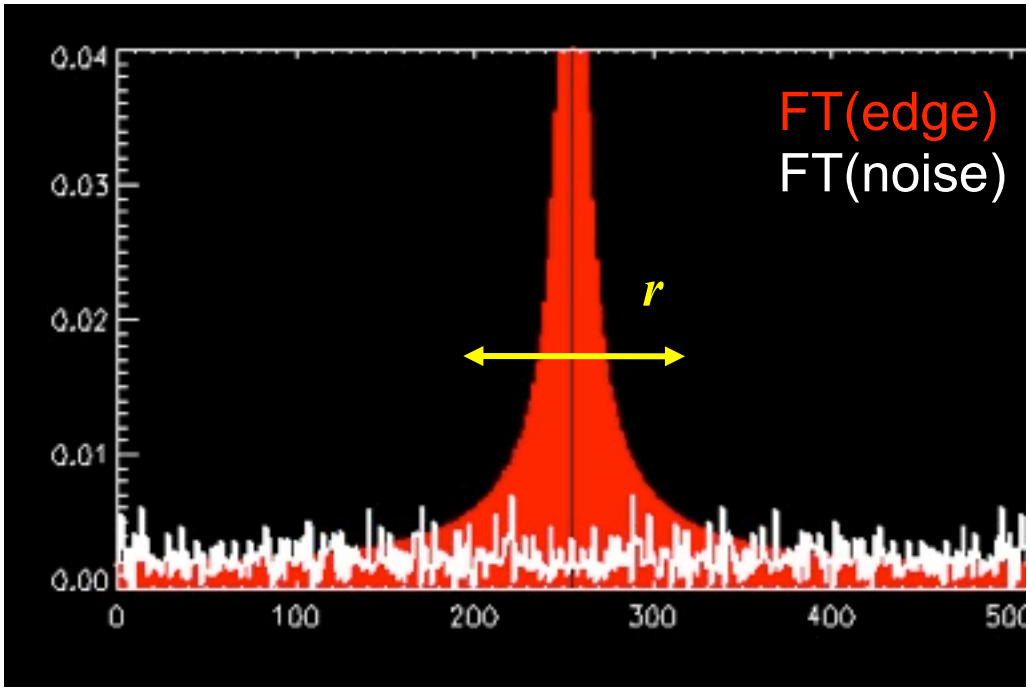
Doppelkante: beide Kantenorte müssen unterscheidbar sein.

Filter suchen mit:

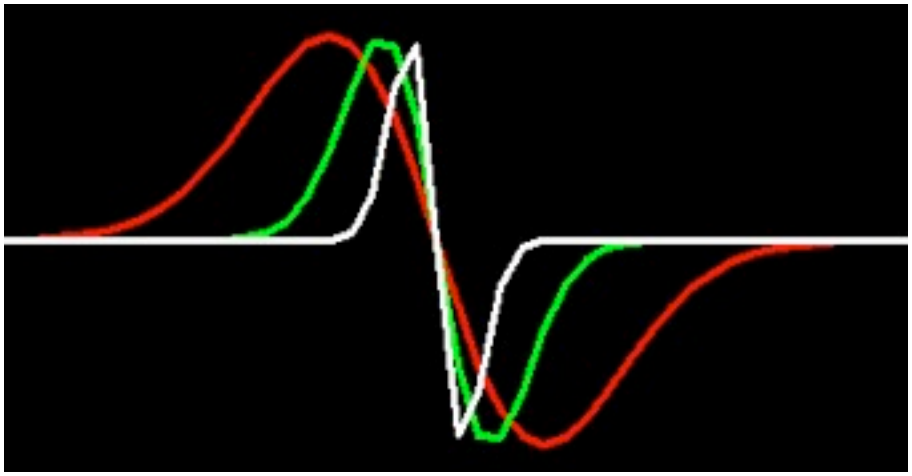
F(0,0)=0

F(u,v)= w, falls (u^2+v^2)<r^2

0, sonst

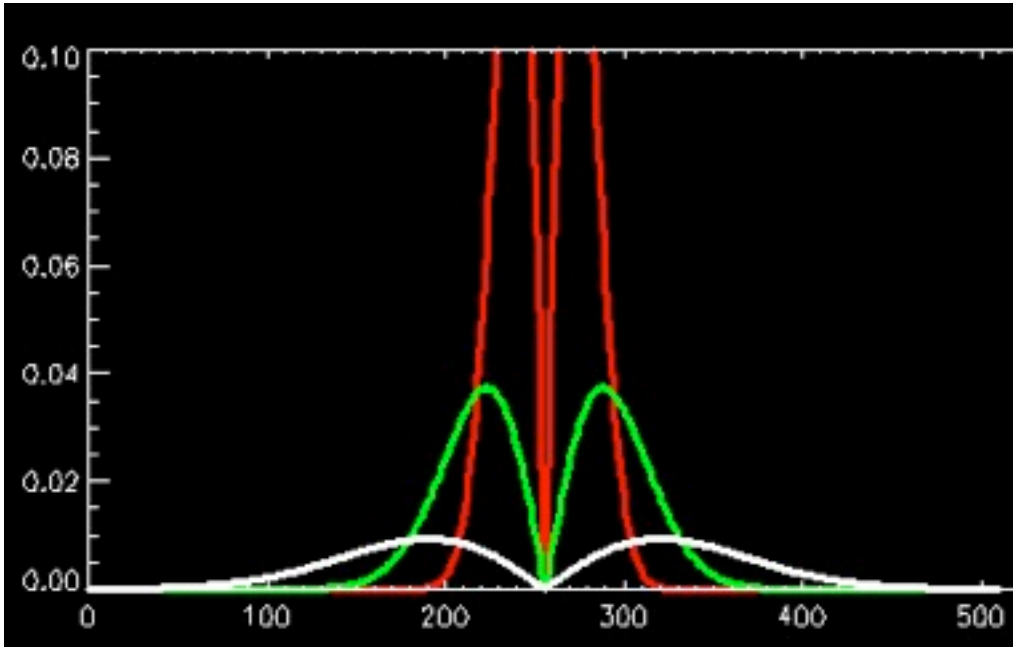


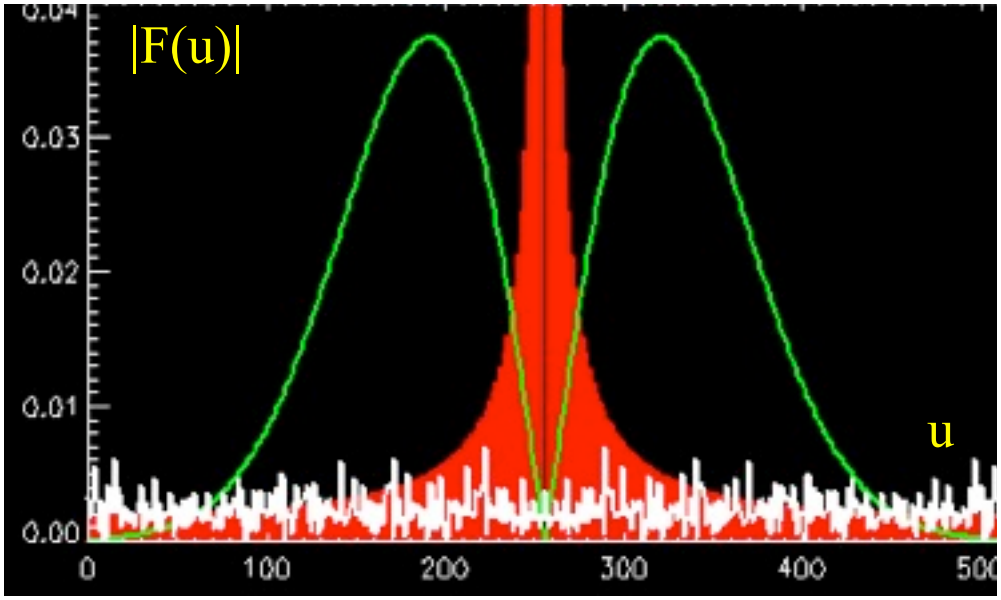
Filter



Gaußfilter mit unterschiedlicher Varianz (im Ortsraum).

Repräsentation
im Frequenzraum

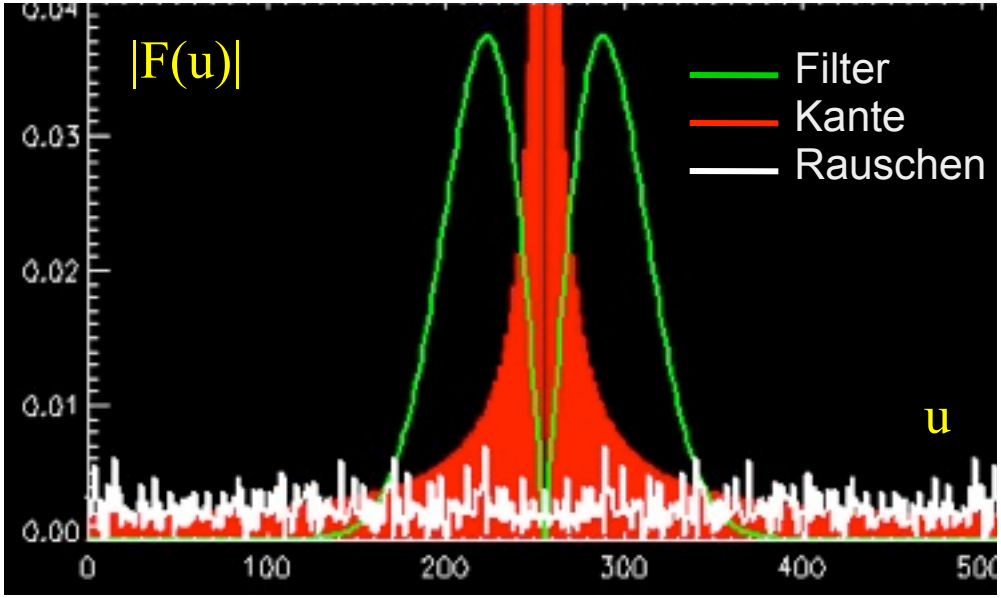




σ=1.0

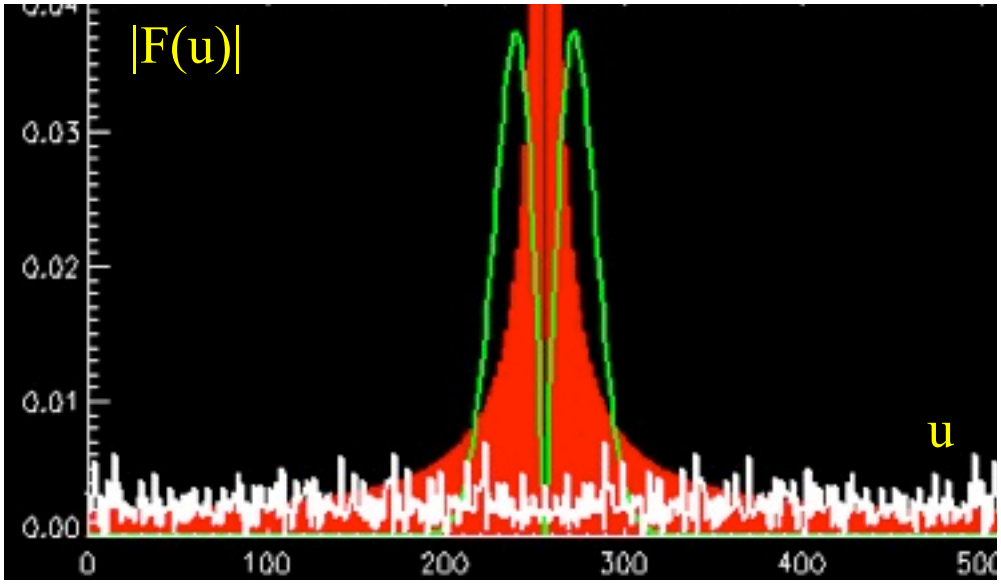
Filter unterdrückt
Rauschen ungenügend

Filter



σ=2.0

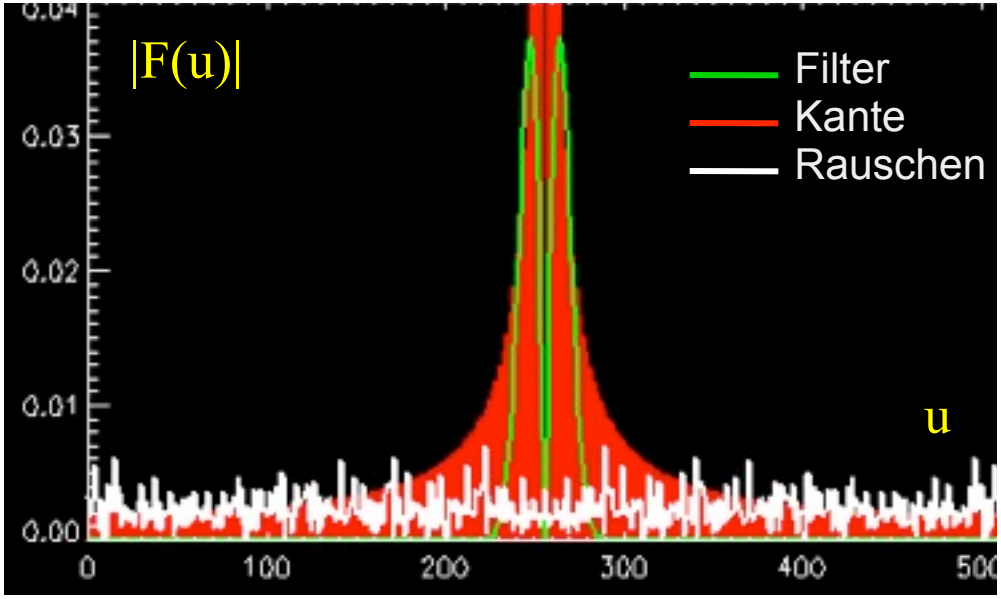
Filter ist nahezu optimal.



σ=4.0

Filter ist nahezu optimal.

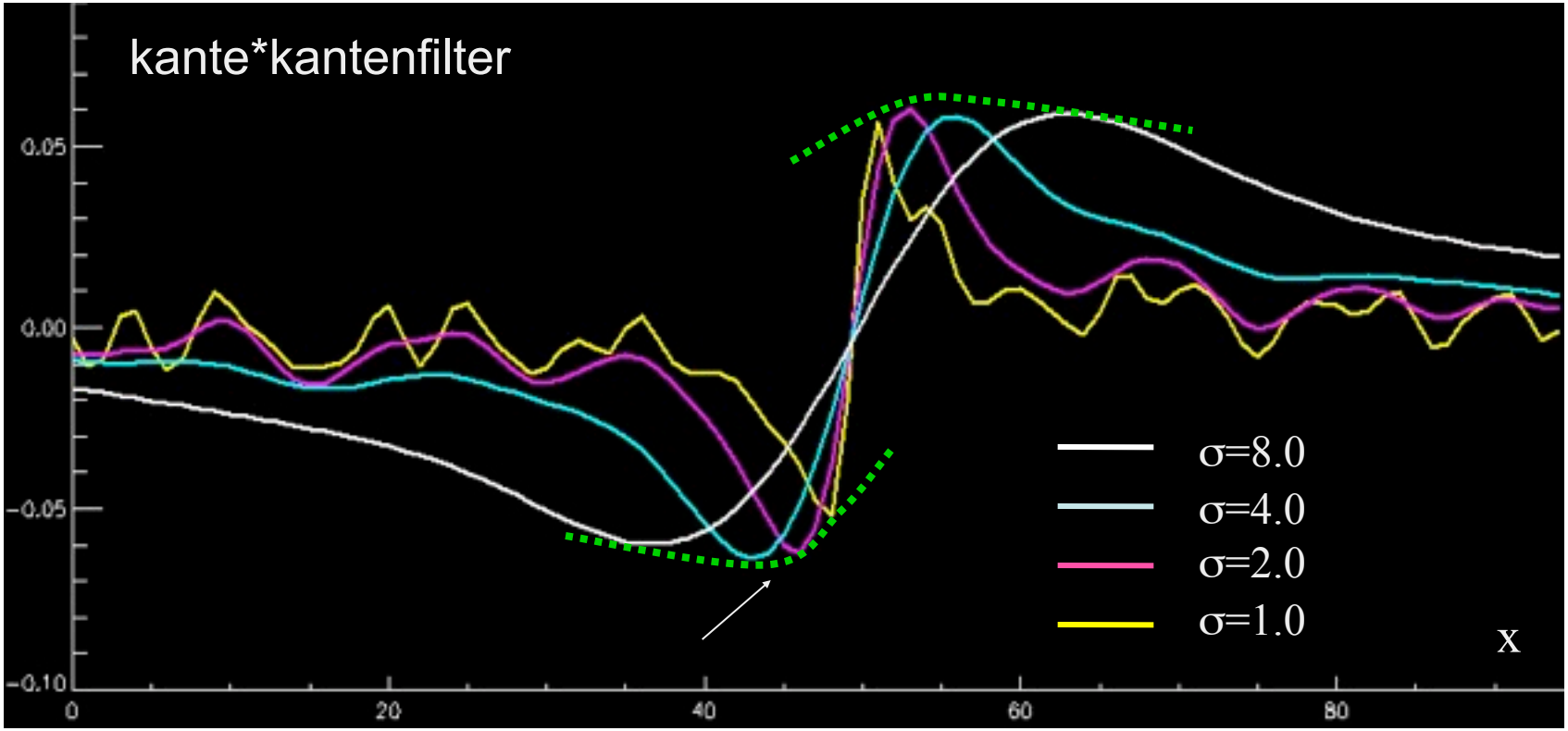
Filter



σ=8.0

Filter unterdrückt Kante.

Filterresultat



Zweidimensionales optimales Filter

- Rauschunterdrückung und Kantenhervorhebung gemäß Frequenzraumcharakteristik.
- Filterung nur entlang größter Steigung
- Canny Edge Enhancement:
  - Richtung der Kante schätzen.
  - Sequenz von Filtern mit unterschiedlicher Glättungscharakteristik in Gradientenrichtung berechnen.

Kompassfilter

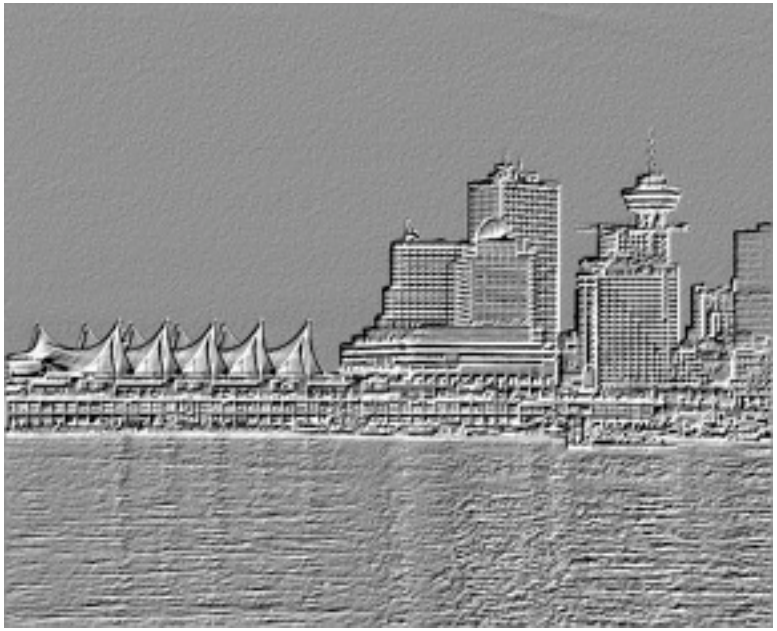
- Dienen der Hervorhebung von Kanten in einer bestimmten Richtung.
- Prewitt-Operator

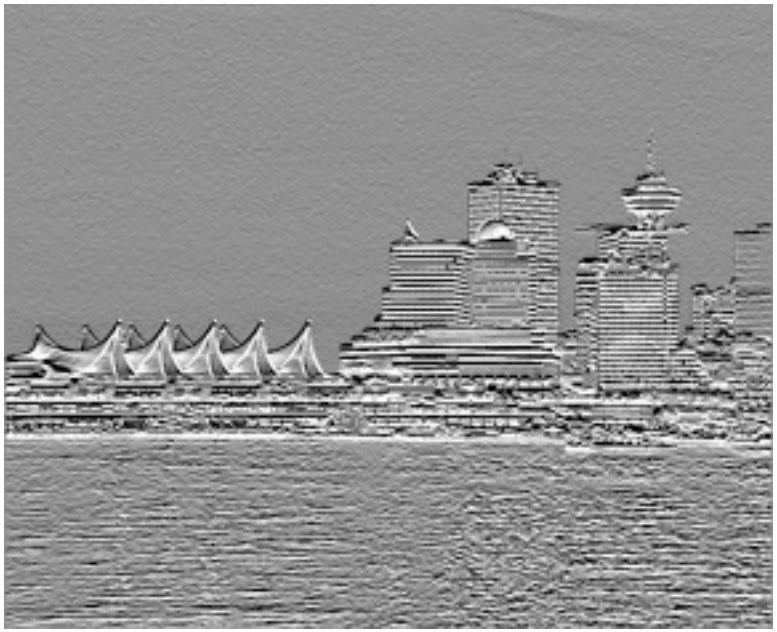
$$h_1 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix} \quad h_2 = \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix} \quad h_3 = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad h_8 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{pmatrix}$$

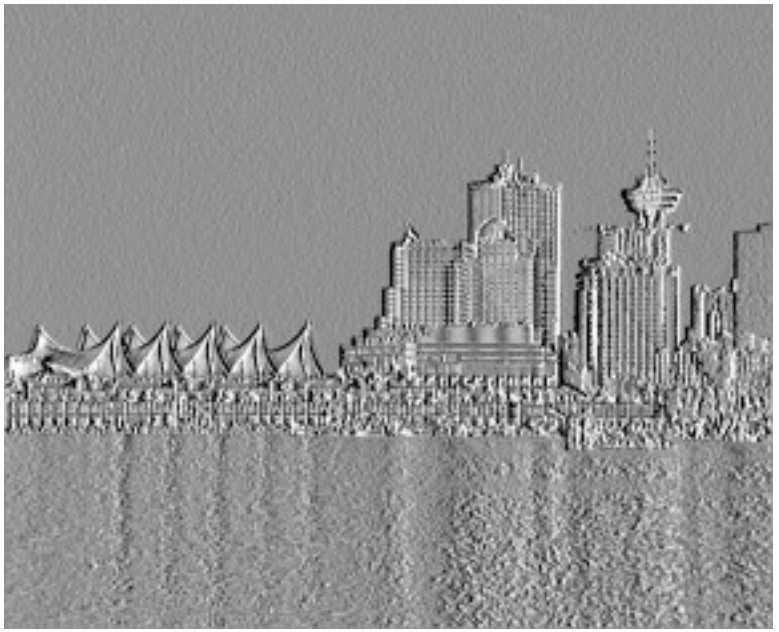
- Kirsch-Operator

$$h_1 = \begin{pmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{pmatrix} \quad h_2 = \begin{pmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{pmatrix} \quad h_3 = \begin{pmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{pmatrix} \quad h_8 = \begin{pmatrix} 3 & 3 & 3 \\ 3 & 0 & -5 \\ 3 & -5 & -5 \end{pmatrix}$$

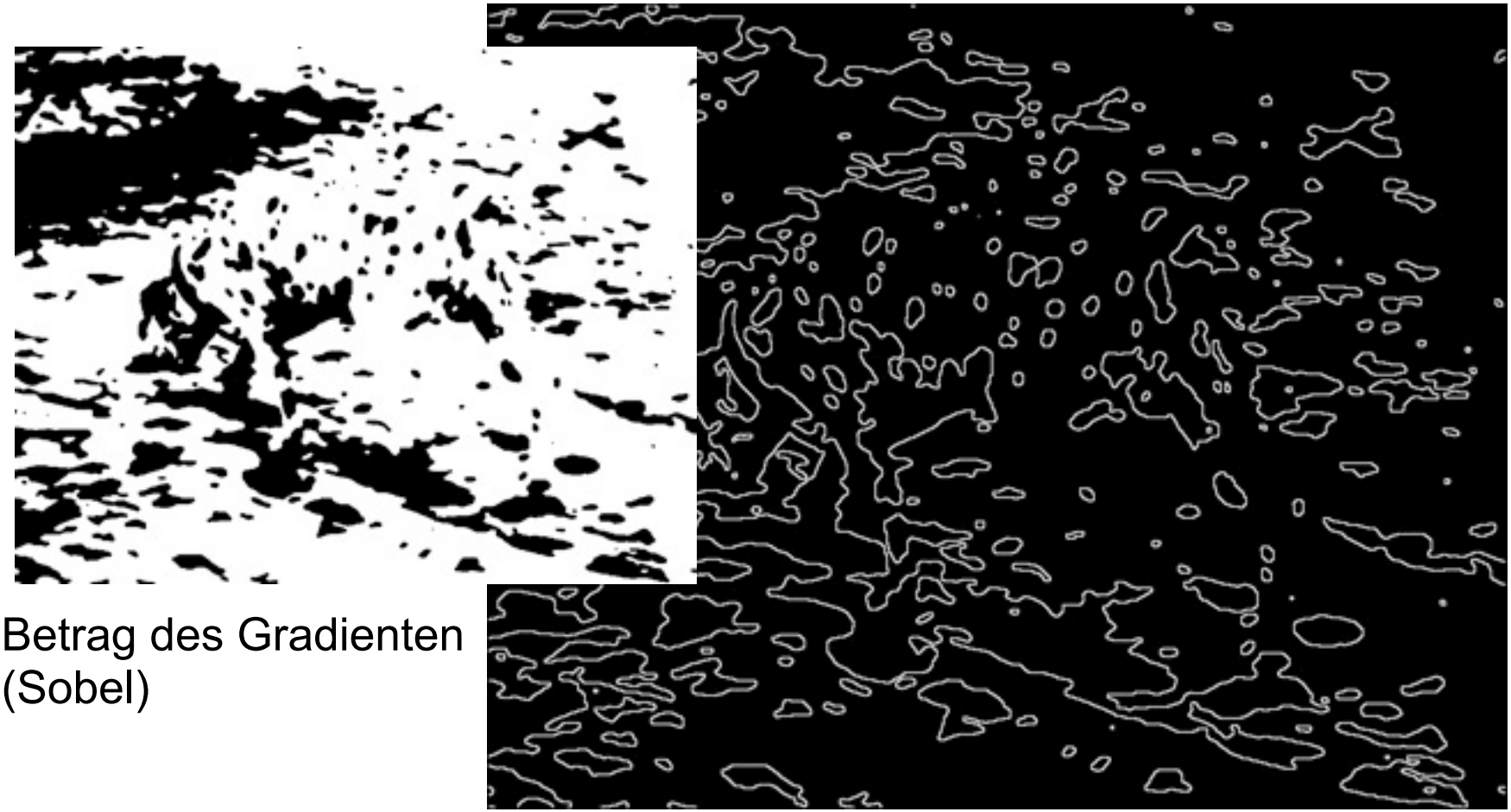
Kirschoperator





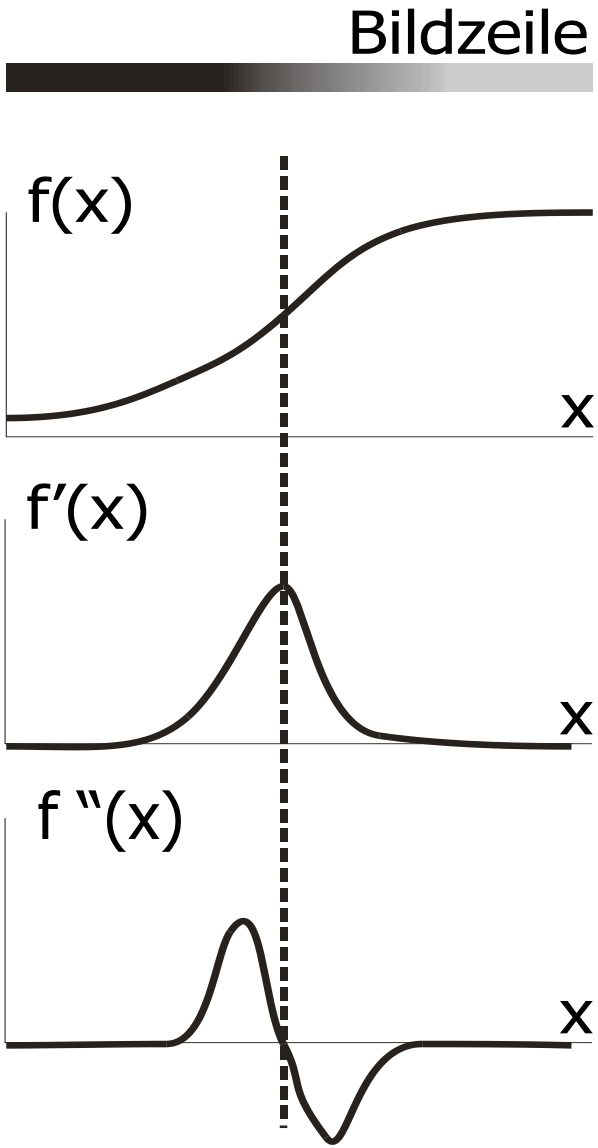


Ecken und Regionengrenzen



Betrag des Gradienten (Sobel)

Wo ist die Regionengrenze?



Vorzeichenwechsel ist leichter zu erkennen, als ein Minimum oder Maximum.

- ➔ Gradient (Länge) als Kennzeichen für die Wichtigkeit einer Kante
- ➔ zweite Ableitung für den Ort der Kante (Nulldurchgang)

Operatoren zur Berechnung der zweiten Ableitung:

- Laplace Filter
- Marr-Hildreth Filter (LoG Filter, Mexican Hat)
- DoG (Difference of Gaussians)

Laplace Funktion

- Summe der partiellen zweiten Ableitungen: $\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$
- Nulldurchgänge der Laplacefunktion produzieren zusammenhängende Kurven entlang von Bildkanten.
- Approximation durch Kombination einer doppelten Differenzbildung in *x*- und *y*-Richtung,
  - z.B. Faltung eines Differenzoperators [-1 1] mit sich selbst: [-1 2 -1]
  - ergibt für [-1 1]*[-1 1] + [-1 1]^T*[-1 1]^T:

$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

  - Summe **aller** partiellen Ableitungen:

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} + \frac{\partial^2 f(x,y)}{\partial x \partial y} + \frac{\partial^2 f(x,y)}{\partial y \partial x} : \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Laplace-Operator

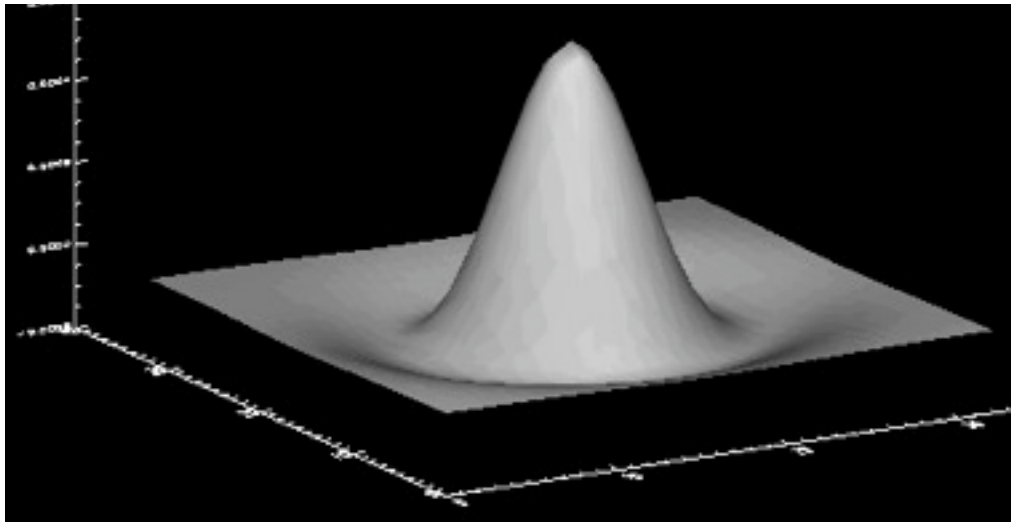


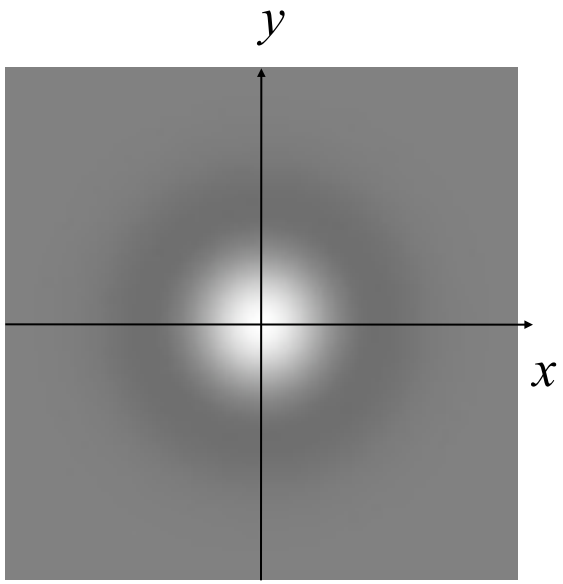


- Nulldurchgänge des Laplace-Operators. Dunkle Pixel bezeichnen negative, helle Pixel bezeichnen positive Werte.

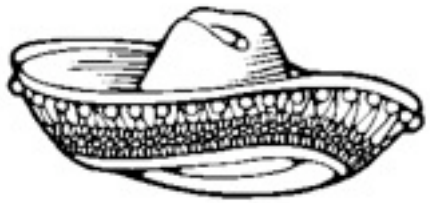
Marr-Hildreth-Filter = LoG-Filter

LoG-Filter: Laplacian-of-Gaussian,
d.h. der Faltung mit dem Laplacefilter geht eine Glättung mit einer Gaußfunktion voraus.

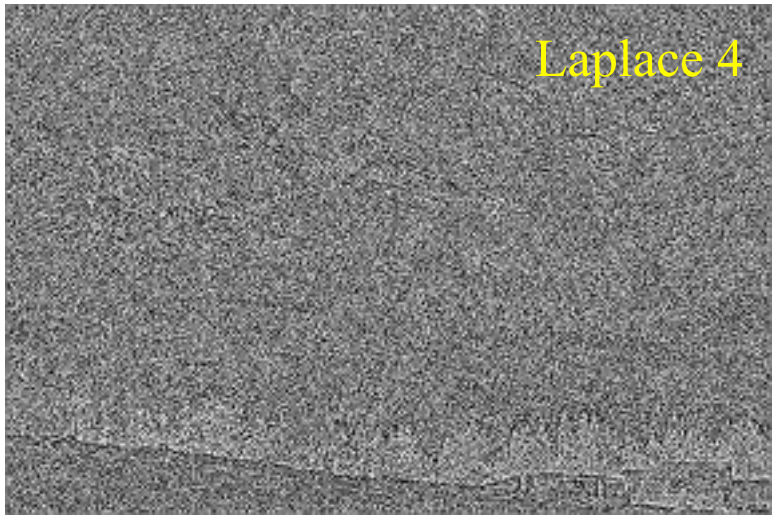


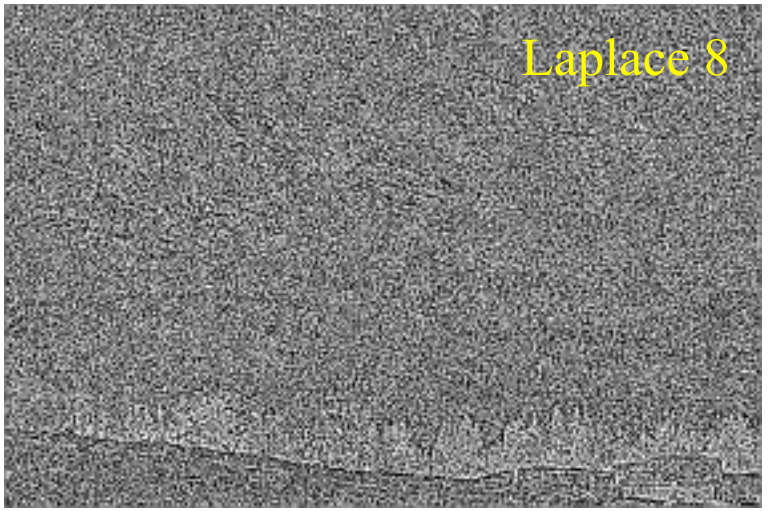


$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left(1-\frac{x^2+y^2}{2\sigma^2}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

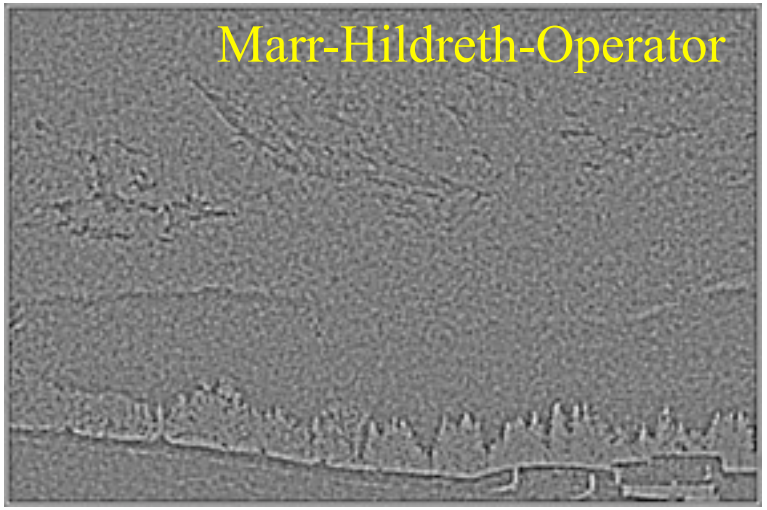


Auch genannt: „Mexican hat“ filter


Laplace 4


Laplace 8

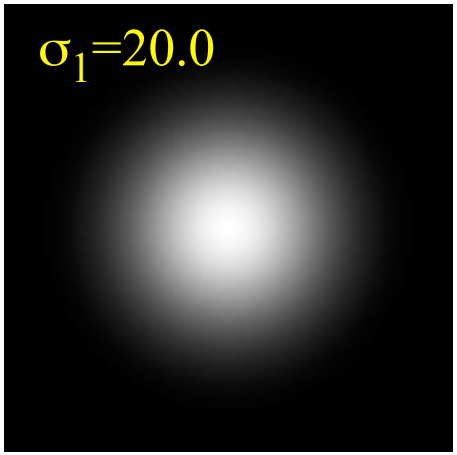


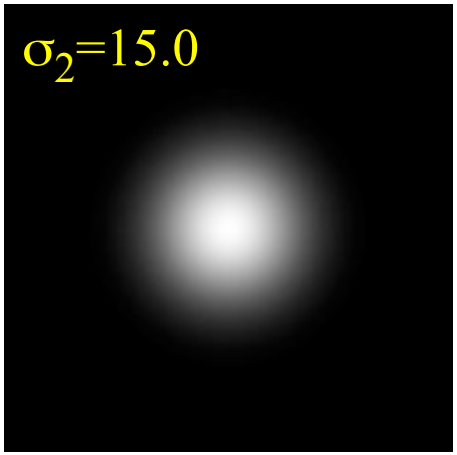

Marr-Hildreth-Operator

DoG (Difference of Gaussians)

Subtraktion zweier Gaußfilter mit unterschiedlicher Varianz σ^2:

$$DoG(x,y,\sigma_1,\sigma_2) = [\sqrt{2\pi}\cdot\sigma_1]^{-1} \cdot \exp(-(x^2+y^2)/2\sigma_1^2) - [\sqrt{2\pi}\cdot\sigma_2]^{-1} \cdot \exp(-(x^2+y^2)/2\sigma_2^2)$$







DoG und LoG-Filter sind in ihrer Wirkungsweise vergleichbar.

..nochmal kurz zusammengefasst:

- Rauschen und Kanten haben im Frequenzbereich ähnliche Attribute.
- Kantenhervorhebung in Orts- und Frequenzraum, Kompassfilter
- Kanteneigenschaften
- Zweite (partielle Ableitung): Laplace-Filter, LoG-Filter,…

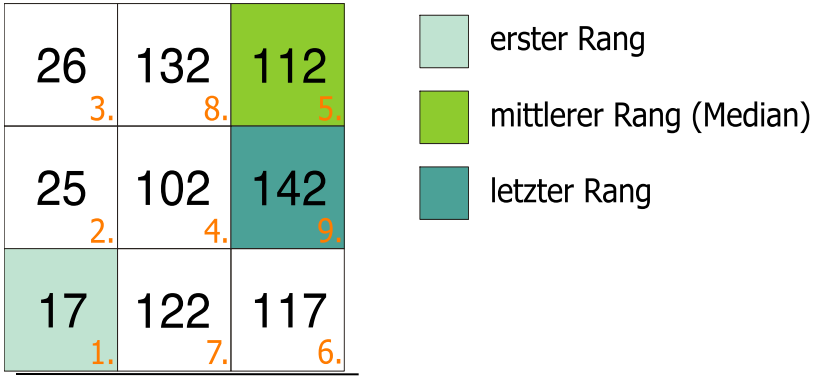
Nichtlineare Filterung

- Rauschen und Kanten haben im Frequenzbereich ähnliche Attribute.
- Ist ein nichtlineares Filter denkbar, das für Rauschen und Kanteneigenschaften unterschiedlich sensitiv ist?
  - Rauschen sind räumlich gleichverteilte Grauwertvariationen.
  - Grauwertvariationen an Kanten sind nicht räumlich gleichverteilt.
  - Filter muss diesen Unterschied berücksichtigen.

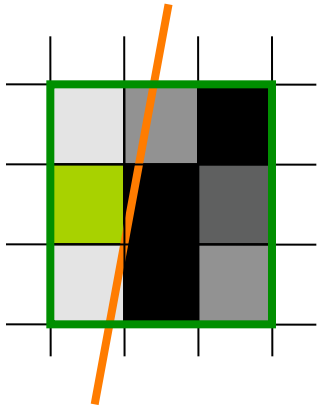
Rangordnungsfilter

- Vorgehen:
  - Sortierung der Elemente in einer Filtermaske
  - Auswahl des an einer bestimmten Stelle einsortierten Werts
  - Eintragung des ausgewählten Werts in die zentrale Position
- Eigenschaften:
  - Es entstehen keine neuen Werte
  - Filter ist nichtlinear, nicht kommutativ, nicht assoziativ

Gebräuchlichstes Rangordnungsfilter ist das Medianfilter



Medianfilter



- Annahmen:
  - Medianfilter ist quadratisch mit ungerader Seitenlänge.
  - Kantensignal ist größer als das Rauschsignal
  - Kante im Filterbereich verläuft in diesem Bereich (nahezu) gerade.
  - Grauwert ist (nahezu) konstant in einer Umgebung von Größe des Filters.
- Kante verläuft durch Filterbereich:
  - Der Mittelpunkt des Filters liegt auf der gleichen Seite der Kante wie die Mehrzahl der Pixel (z.B. rechts von der Kante).
  - Pixel von dieser Seite (z.B. rechts) wird selektiert (kantenerhaltend).
- Keine Kante im Filterbereich:
  - Median nähert sich dem Erwartungswert mit Anzahl der Stichproben (rauschunterdrückend).

Medianfilter





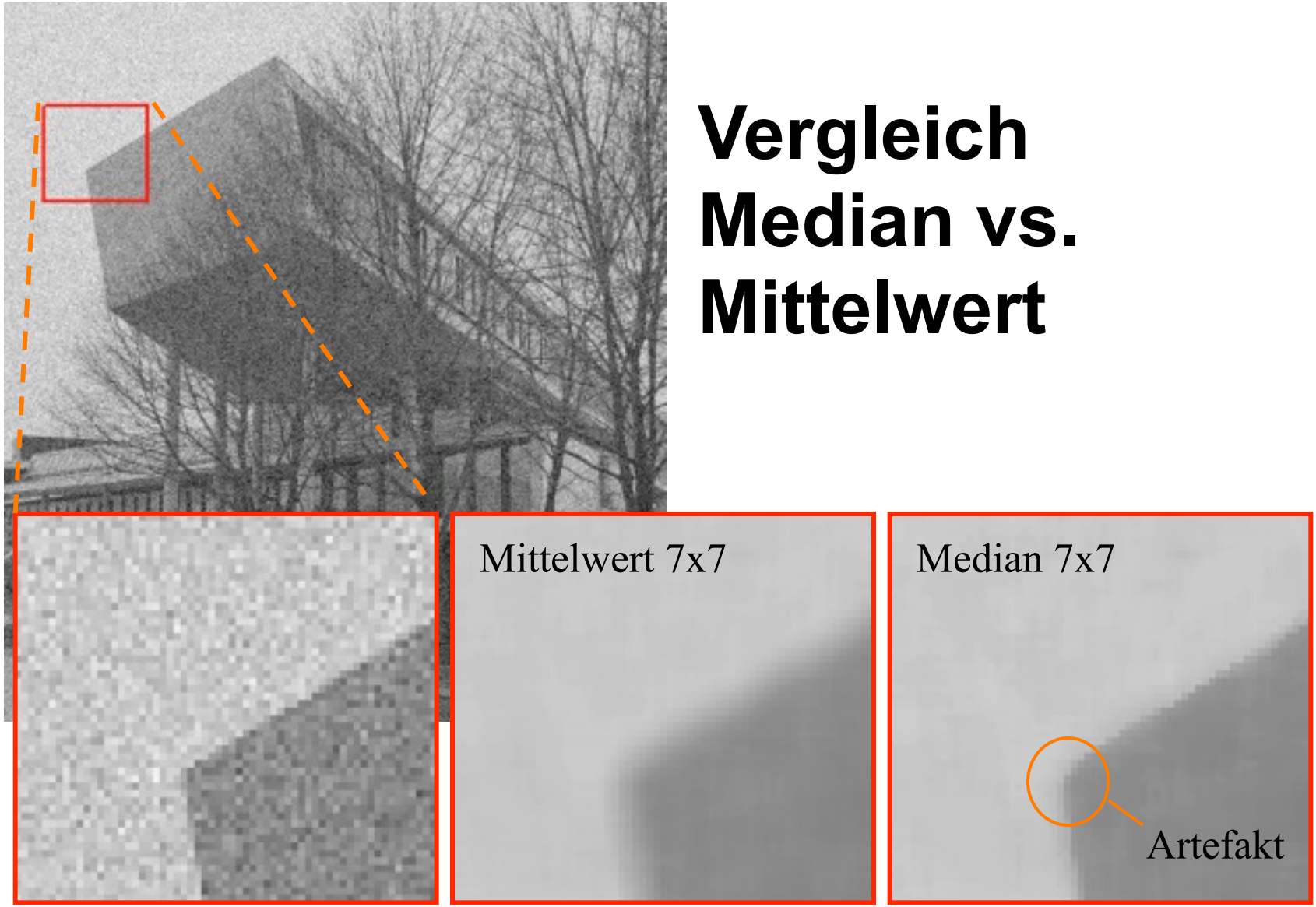
Median zur Entfernung von Salt&Pepper-Rauschen in einer 3x3 Umgebung.

Medianfilter



- Durch Medianfilterung (rechts) kann Impulsrauschen im Gegensatz zur Mittelwertfilterung (Mitte) beseitigt werden.

Vergleich Median vs. Mittelwert

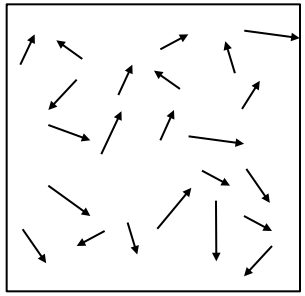


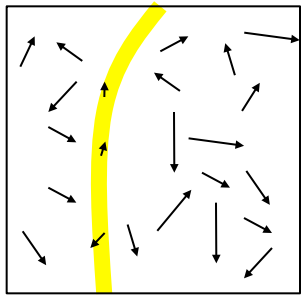
Diffusionsfilter

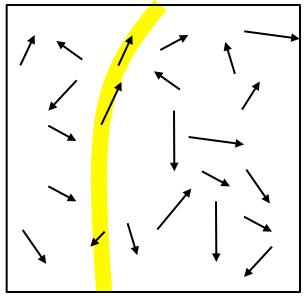
- Diffusionsprozess ist Modell für die Filterung
- Diffusion:
  - Flüssigkeit mit lokal variierender Dichte
  - Lokaler Ausgleich in Abhängigkeit vom Dichtegradienten
- Übertragung auf ein Bild:
  - Dichte ist die Bildfunktion
  - Dichtegradient ist Grauwertgradient

Diffusion

- Homogene Diffusion: Szene besteht aus einem einzigen Material und die Diffusion ist proportional zum Dichtegradienten dieses Materials.
- Inhomogene Diffusion: Szene enthält Barrierematerialien, wo die Diffusion erschwert wird.
- Anisotrope, inhomogene Diffusion: Diffusion an Barrierematerialien ist richtungsabhängig.







Fick‘sches Gesetz

$$\vec{j}(x,y) = -\mathbf{D}(x,y) \times \nabla u(x,y) = -\mathbf{D}(x,y) \times \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix}$$

D – Diffusionstensor

∇u - Dichtegradient

- Der Diffusionstensor spezifiziert die Art des Fluss $\vec{j}$ in Abhängigkeit vom Dichtegradient ∇u.
- Die Konzentrationsveränderung durch $\vec{j}$ ist:

$$\frac{\partial u(x,y,t)}{\partial t} = -\operatorname{div} \vec{j}(x,y,t)$$

- Der Divergenzoperator ist definiert durch:

$$\operatorname{div} \vec{j} = \begin{pmatrix} \frac{\partial j_x(x,y,t)}{\partial x} + \frac{\partial j_y(x,y,t)}{\partial y} \end{pmatrix}$$

Fick‘sches Gesetz (Namensherkunft, Quelle: Wikipedia)

- **Adolf Eugen Fick** (* 3. September 1829 in Kassel, † 21. August 1901 in Blankenberghe, Flandern) war ein deutscher Physiologe.
- Er stellte 1855 die nach ihm benannten Gesetze der Diffusion auf, die eine wesentliche Grundlage zur Entwicklung der Kontaktlinse bildeten.
- Die Adolf-Fick-Stiftung vergibt alljährlich den Adolf-Fick-Preis an einen herausragenden Physiologen. Der Preis wird als die wichtigste Auszeichnung auf dem Gebiet der deutschsprachigen Physiologie angesehen.



Berechnung der Diffusion

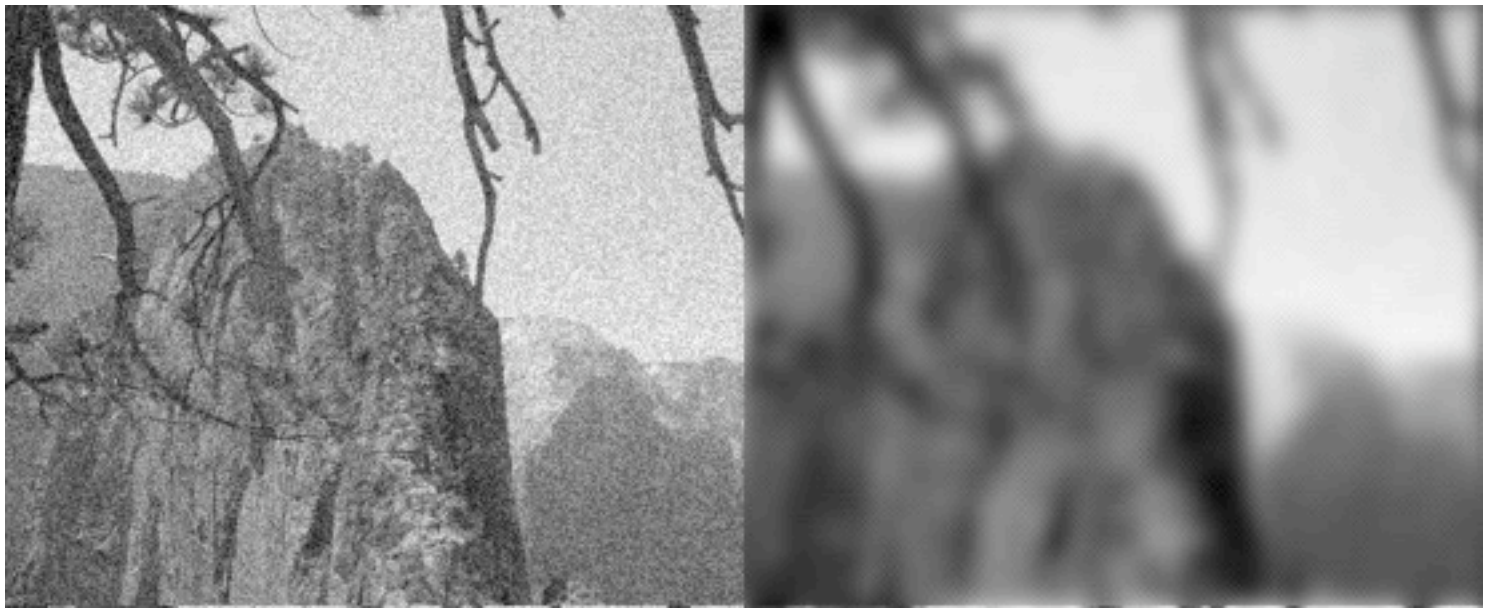
- Die Belegung des Diffusionstensors gibt an, ob es sich um homogene, inhomogene oder anisotrope, inhomogene Diffusion handelt.
- Die Differentialgleichung lässt sich im Allgemeinen nicht analytisch lösen.
- Iterative Approximation:

$$u(x, y, t_{i+1}) = u(x, y, t_i) + (t_{i+1} - t_i) \frac{\partial u(x, y, t_i)}{\partial t}.$$

Die Intervallgröße muss sorgfältig gewählt werden, damit der Schätzfehler durch die Approximation nicht zu groß wird.

Homogene Diffusion

$$\mathbf{D} = \begin{pmatrix} \varepsilon & 0 \\ 0 & \varepsilon \end{pmatrix} \quad \text{und} \quad \operatorname{div} \vec{j} = \varepsilon \frac{\partial f(x,y,t)}{\partial x} + \varepsilon \frac{\partial f(x,y,t)}{\partial y}$$



Inhomogene Diffusion

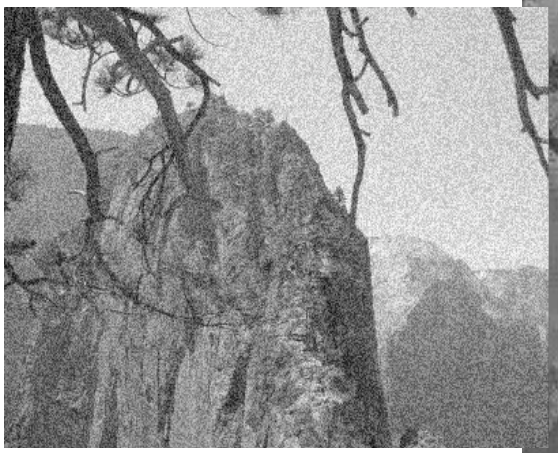
- **Ziel**: keine (wenig) Diffusion an Kanten.
- **Problem**: Kantenverlauf ist nicht bekannt.
- **Annahme**: Kante = hoher Gradient, d.h. Diffusionstensor muss von der Gradientenlänge abhängig sein.

$$\mathbf{D} = \begin{pmatrix} \varepsilon\left(\|\nabla u\|^2\right) & 0 \\ 0 & \varepsilon\left(\|\nabla u\|^2\right) \end{pmatrix} \quad \text{mit} \quad \varepsilon\left(\|\nabla u\|^2\right) = \varepsilon_0 \frac{\lambda^2}{\|\nabla u\|^2 + \lambda^2}$$

Der Parameter λ steuert, wie stark der Gradient die Diffusion beeinflusst.

Inhomogene Diffusion

Rauschen an den Kanten bleibt erhalten.

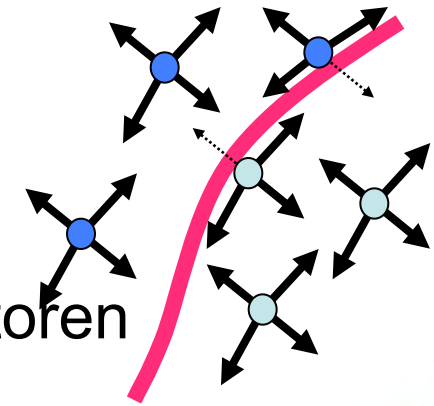




Anisotrope, inhomogene Diffusion

- Ziel: keine Diffusion über Kanten hinaus, aber entlang von Kanten.
- Diffusionstensor muss Kantenrichtung mit berücksichtigen.
- Diffusionstensor kann zerlegt werden in:

$$\mathbf{D} = \begin{pmatrix} e_{1,1} & e_{2,1} \\ e_{1,2} & e_{2,2} \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} e_{1,1} & e_{1,2} \\ e_{2,1} & e_{2,2} \end{pmatrix}$$



- Adäquate Bestimmung der Eigenvektoren und Eigenwerte löst das Problem.



Eigenvektoren für die anisotrope Diffusion

$$\begin{pmatrix} e_{1,1} & e_{1,2} \end{pmatrix} = \frac{\nabla u}{\|\nabla u\|}, \quad \lambda_1 = \varepsilon\left(\|\nabla u\|^2\right)$$

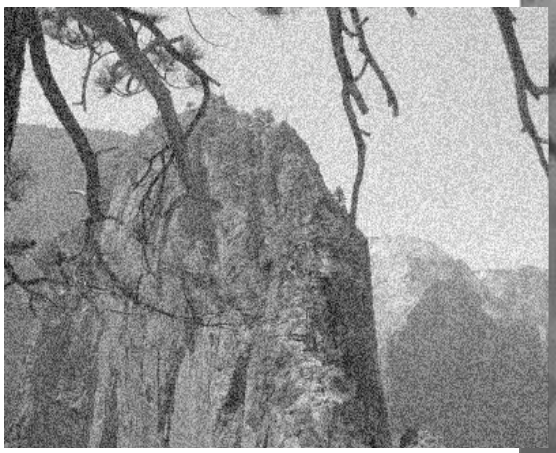
$$\begin{pmatrix} e_{2,1} & e_{2,2} \end{pmatrix} = \begin{pmatrix} e_{1,2} & -e_{1,1} \end{pmatrix}, \quad \lambda_2 = 1.$$

$$\text{mit} \quad \varepsilon\left(\|\nabla u\|^2\right) = \varepsilon_0 \frac{\lambda^2}{\|\nabla u\|^2 + \lambda^2}$$

- Erster Eigenvektor in Richtung des Gradienten und Eigenwert sinkt mit Gradientenlänge: keine Diffusion über Kanten hinweg.
- Zweiter Eigenvektor parallel zur Kante und Eigenwert konstant: Diffusion parallel zur Kante ist von der Kantenstärke unabhängig.

Anisotrope Diffusion







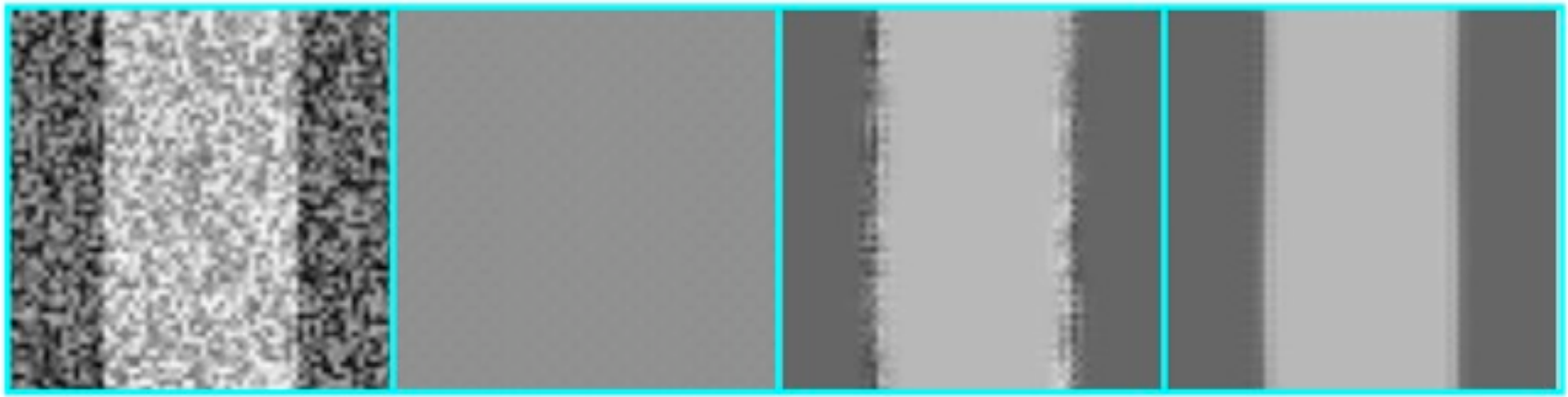
Anistrope Diffusion





Diffusion im Vergleich

- Homogene Diffusion: vollständige Homogenisierung
- Inhomogene Diffusion
  - Isotrop: Erhält Kanten, aber dort auch Rauschen.
  - Anisotrop: unterdrückt Rauschen auch an Kanten.
- Modell für Dichtegradient und Kante sind gleich
- Es gibt keine 100%-ige Kantenentscheidung: Jede Diffusion führt nach genügend vielen Iterationen zur vollständigen Homogenisierung



Resultat



σ=2.0 σ=5.0 σ=9.0

20 Iterationen

200 Iterationen

..Also nochmal kurz:

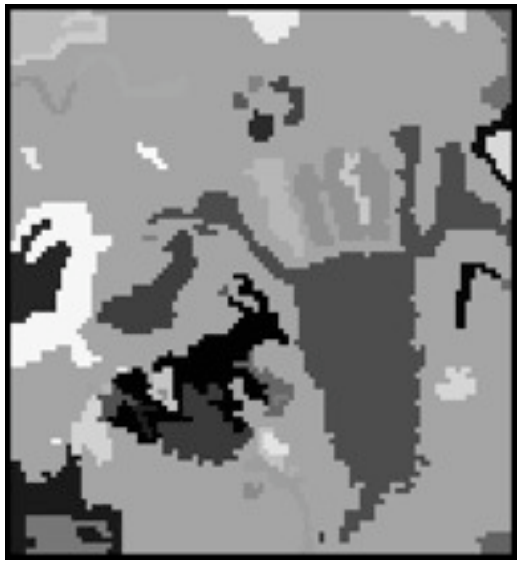
- Durch nichtlineare Operatoren können Kanten und Rauschen getrennt behandelt werden.
- Medianfilterung und Kantenmodell der Medianfilterung.
- Diffusionsfilter
  - Diffusionsmodell
  - Homogene, inhomogene und anisotrope Diffusion
  - Parametrisierung

Themen jetzt gleich

- Segmentierung
  - Thresholding
  - Region labeling
  - Flood fill
  - Relaxation labeling
- Regionenbasierte Segmentierung
  - Multiskalenstrategien
  - Region Merging
  - Split-and-Merge
  - Textursegmentierung
- Kantenbasierte Segmentierung
  - Edge Linking und Canny Edge Operator
  - Nulldurchgänge
  - Wasserscheidentransformation

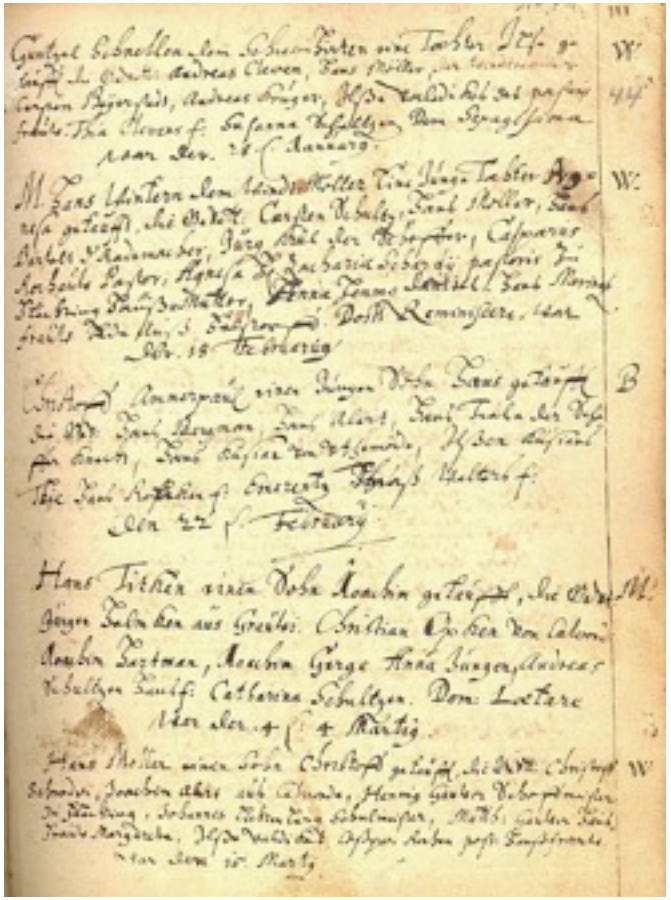
Segmentierung

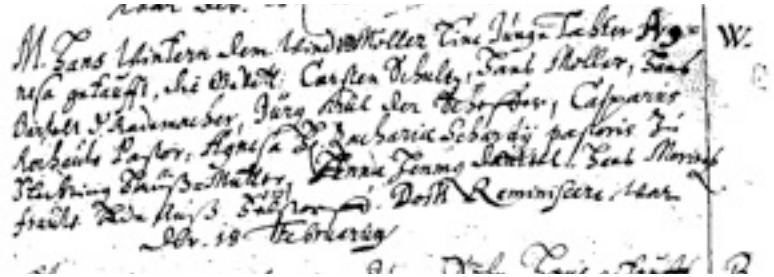
- Ziel: Pixelaggregate schaffen, denen Bedeutung zugeordnet werden kann
- Pixel gehören zusammen, wenn sie einem zusammenhängenden Gebiet angehören.
- Homogenitätsbedingung ist vom Bildinhalt unabhängig (Segmentierung ist datenbasiert).
- Modellbasierte Segmentierung: Zwitter zwischen Segmentierung und Analyse.



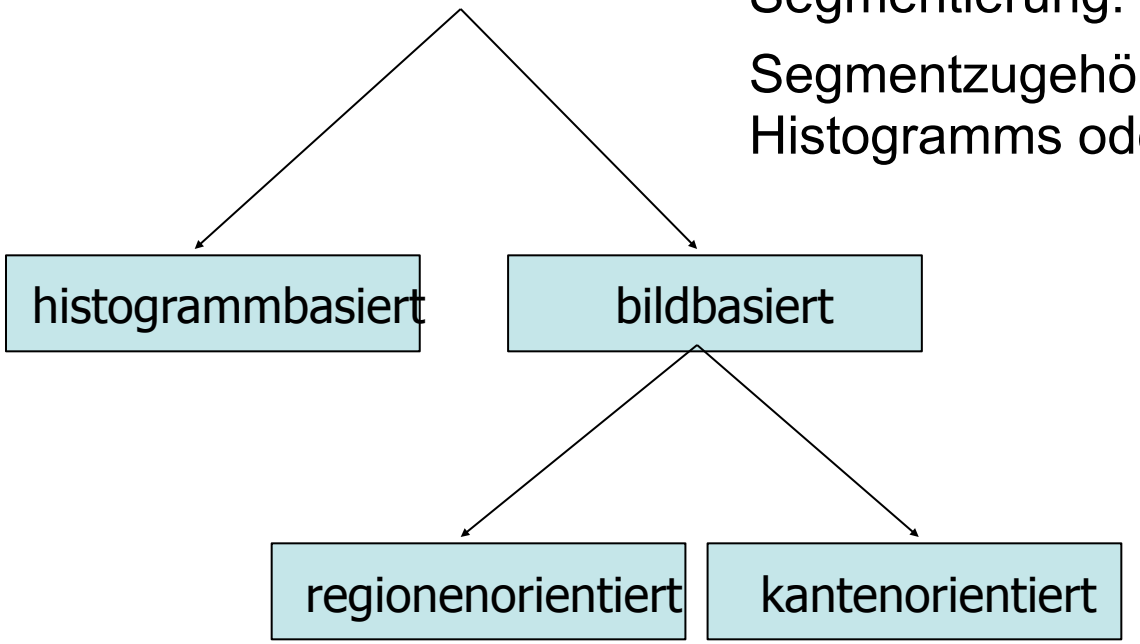
Segmentierung

- Zerlegung eines Bildes in semantische Einheiten.
- Segmente: Träger der Bedeutung von Strukturen eines Bildes.
- Eigenschaften einer Segmentierung:
  - **vollständig**: jedes Pixel ist einem Segment zugeordnet.
  - **überdeckungsfrei**: ein Pixel ist genau einem Segment zugeordnet.
  - **zusammenhängend**: jedes Segment bildet ein zusammenhängendes Gebiet.





Segmentierungsmethoden



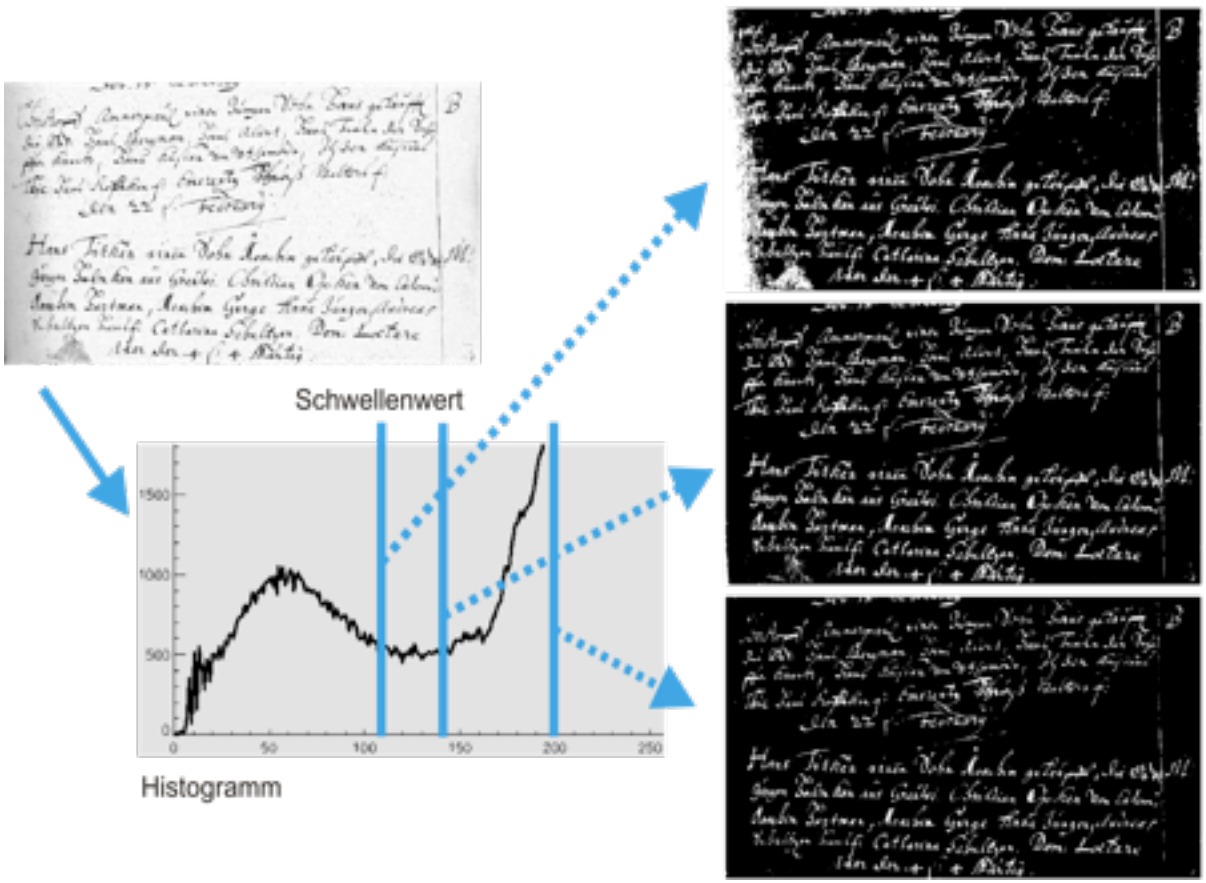
Histogrammbasierte oder bildbasierte Segmentierung:

Segmentzugehörigkeit wird anhand des Histogramms oder des Bildes entschieden

Regionen- oder Kantenorientierung:

Segmente werden durch ihre Grenzen oder ihr Inneres definiert

Histogrammbasierte Segmentierung



Annahme: Bild besteht aus zwei Anteilen die sich durch ihren Grauwert unterscheiden.

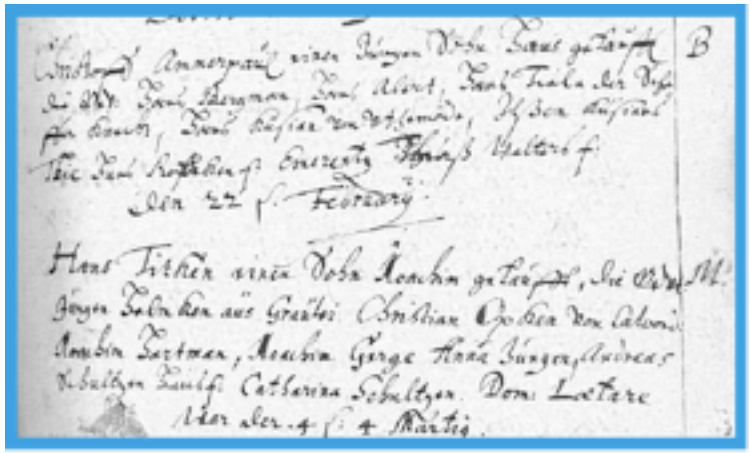
Aufgabe: Schwellenwert zwischen den beiden Grauwerten finden.

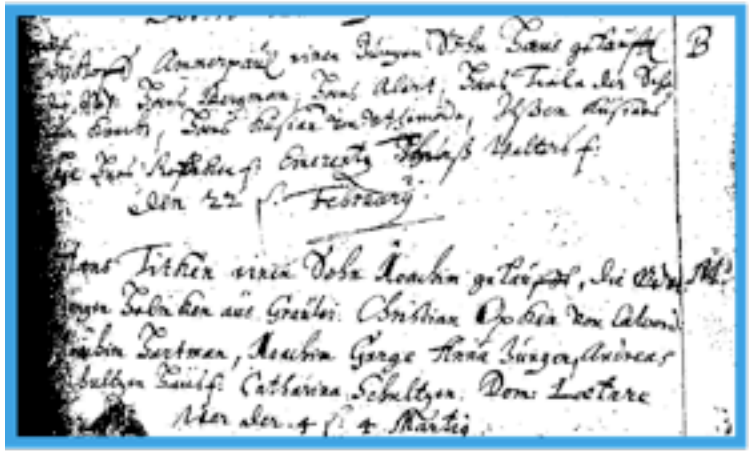
$$b(i,j) = \begin{cases} 1, & \text{falls } f(i,j) > T \\ 0, & \text{sonst} \end{cases}$$

Shading

Helligkeitsvariationen zerstören die bimodale Verteilung der Häufigkeiten.

- Schwellenwert ist nicht mehr für das gesamte Bild definierbar.



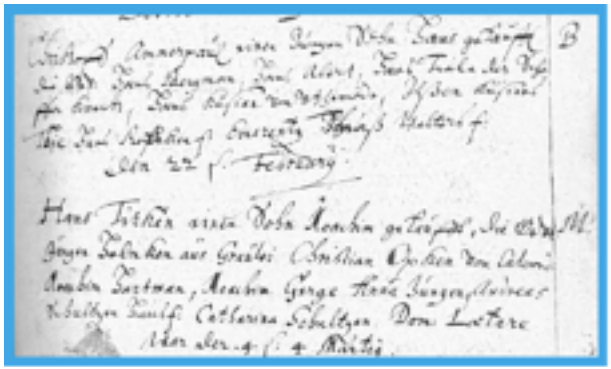


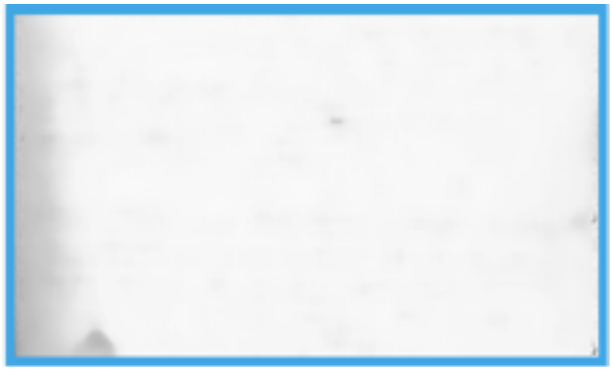
Berücksichtung von Shading

- Homogenes Bild unter gleichen Bedingungen aufnehmen.
- Shading-Bild aus dem Bild selbst bestimmen.
- Variierende Schwellenwerte.

Bestimmung des Shadingbilds

- Falls die überwiegende Mehrzahl des Bildes aus Vorder- oder Hintergrund-pixeln besteht, kann das Shadingbild durch einem Rangordnungsfilter erzeugt werden.
- Das Rangordnungsfilter muss mindestens so groß sein, dass immer mindestens ein Vordergrund- und ein Hintergrundpixel enthalten ist.





13×13 Maximumsfilter

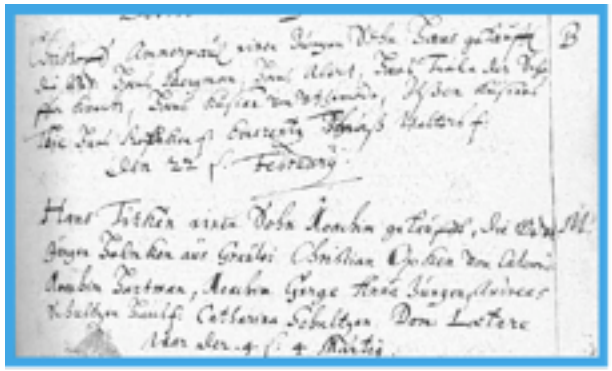
Shading-Korrektur

Berechne aus dem Hintergrundsbild Shadingfunktion $s(i,j)$

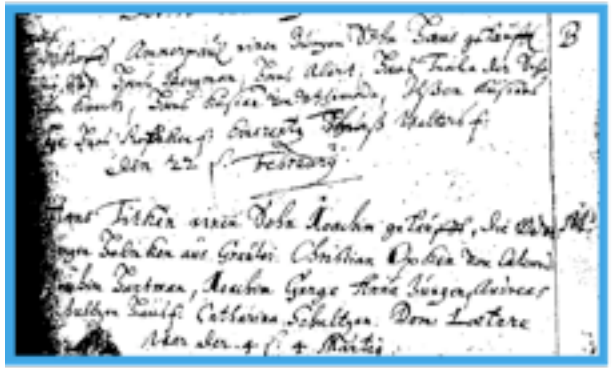
Korrektur:

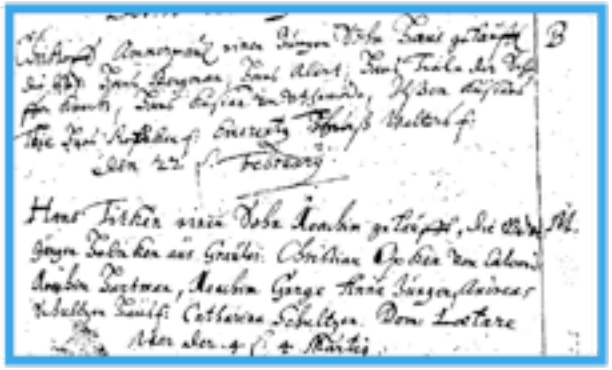
$$f'(i,j)=f(i,j)\cdot s(i,j)$$

Segmentierung auf dem korrigierten Bild.





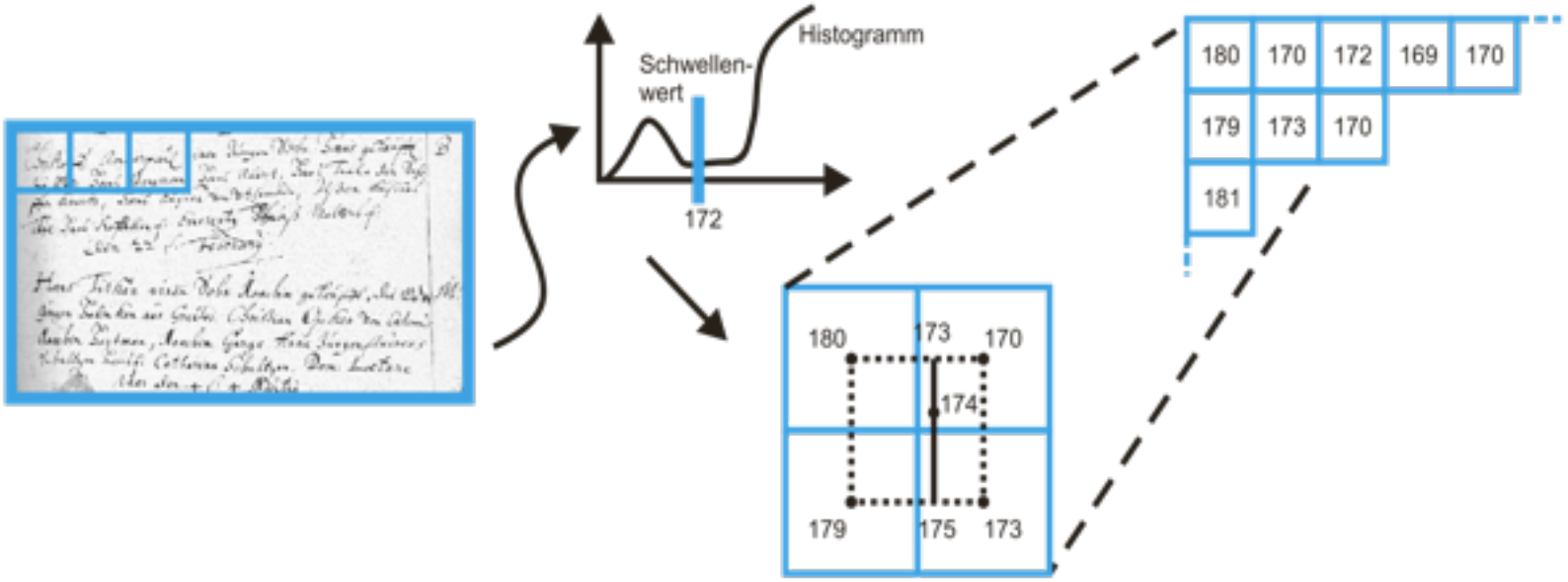




ohne Korrektur

mit Korrektur

Variierende Schwelle



Lokale Schwellenwerte aus Histogramme in Teilregionen.

(Lineare) Interpolation von Schwellenwerten $T(i,j)$ an allen anderen Punkten.

Segmentierung durch $f(i,j) > T(i,j)$.

Region Labeling

- Schwellenwert zerlegt das Bild in Vordergrund und Hintergrundsegmente.
- **Region Labeling** bestimmt Ort und Anzahl aller zusammenhängenden Gebiete im Binärbild *b*:

```
region.initialise() // Region der Größe M,N erzeugen und
label=1                    // mit Null initialisieren, Startlabel=1
for (i,j) = 0, (M,N) do    // Doppelschleife über i und j
  if region.labels(i,j) = 0 then // dieser Ort ist noch nicht
                                                    Teil einer Region
          label = label+1                            // neues Label
  vergeben
          region.flood_fill(i,j,label)  // zusammenhängendes
                                                    Gebiet um
  (i,j) mit
                                                    Label füllen
```

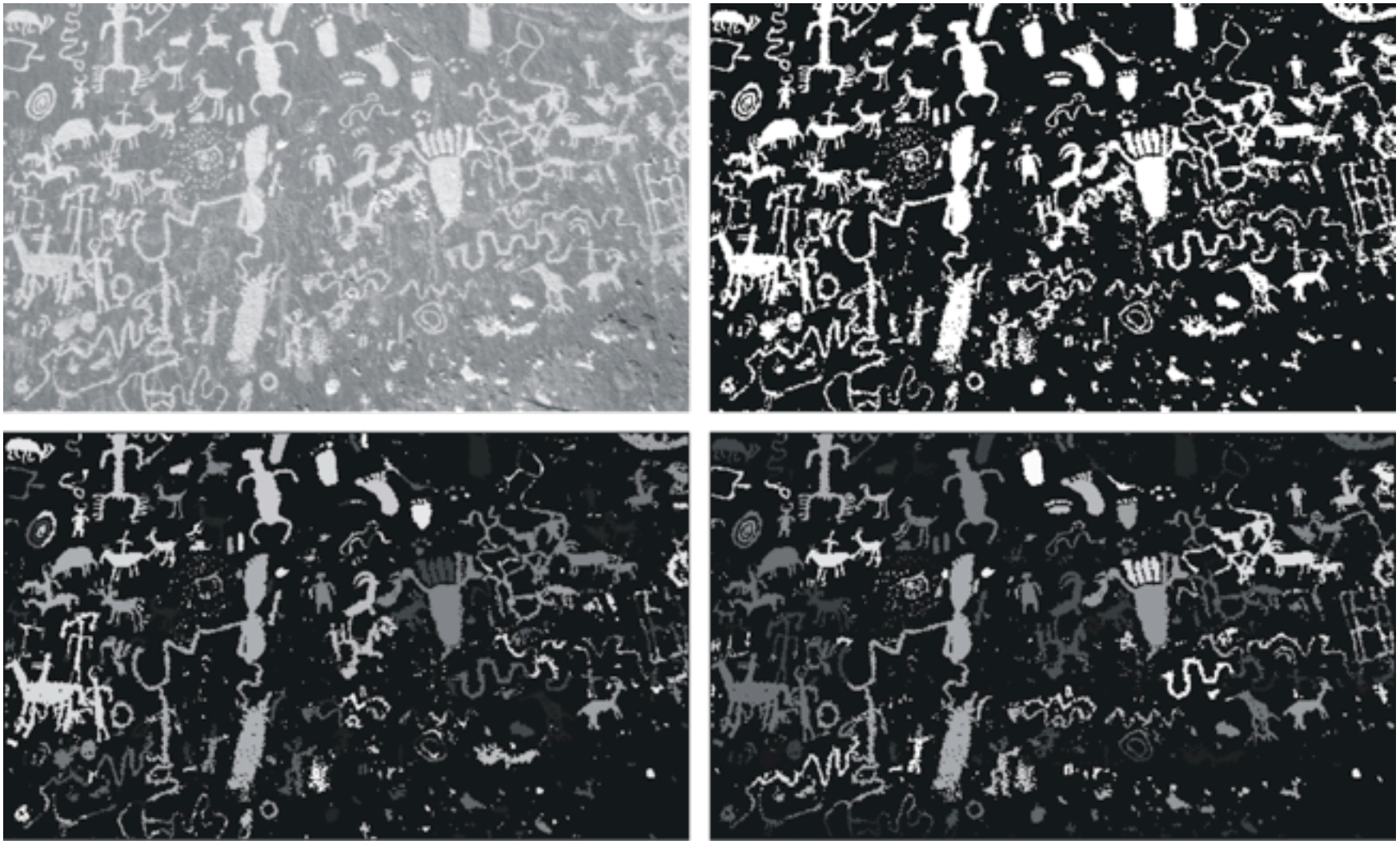
Flood Fill

```
flood_fill(i,j,label)  // Variablen zur Auswertung der
       Zusammenhangsbedingung sind global verfügbar
if f(i,j) erfüllt Zusammenhangsbedingung then
    region(i,j) = label      // Region an (i,j) mit label
                                  // versehen
    flood_fill(i-1,j,label) // Nachbarpixel untersuchen
    flood_fill(i,j-1,label)
    flood_fill(i+1,j,label)
    flood_fill(i,j+1,label)
```

Region labeling + flood fill:
- Vollständige Segm.?
- Überdeckungsfrei?
- Zusammenhängend?

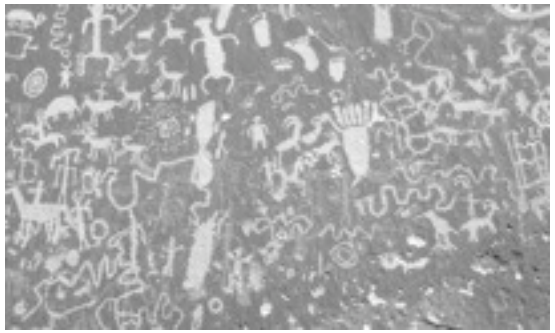
Bsp. für Zusammenhangsbedingung:
- hat den gleichen Grauwert wie Saatpunkt

Resultat



Nachverarbeitung

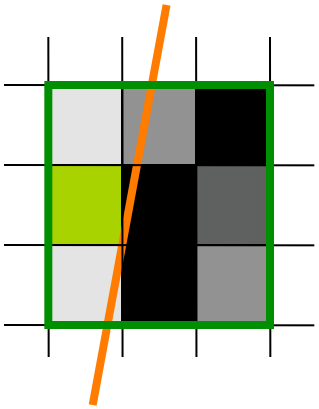
- Trennung nach Grauwerten wird nicht perfekt sein.
- Schwellenwertbild enthält falsche Regionen:
  - kleine fälschlicherweise als Segmente identifizierte Regionen.
  - Störungen am Rand von Regionen.
- Nachverarbeitung
  - Medianfilterung auf den Labeln
  - Entfernung von zu kleinen Regionen
  - Relaxation Labeling



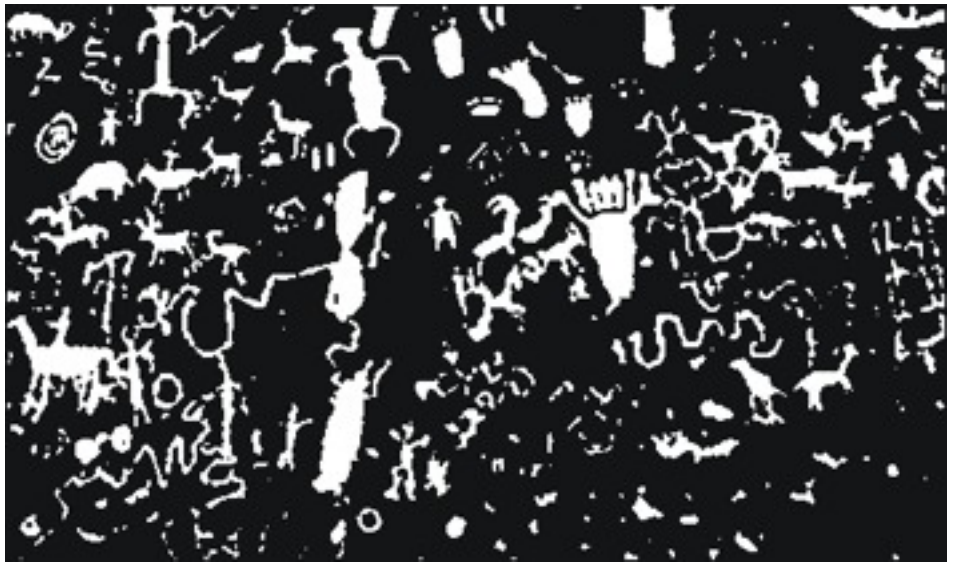


Medianfilterung auf Labeln

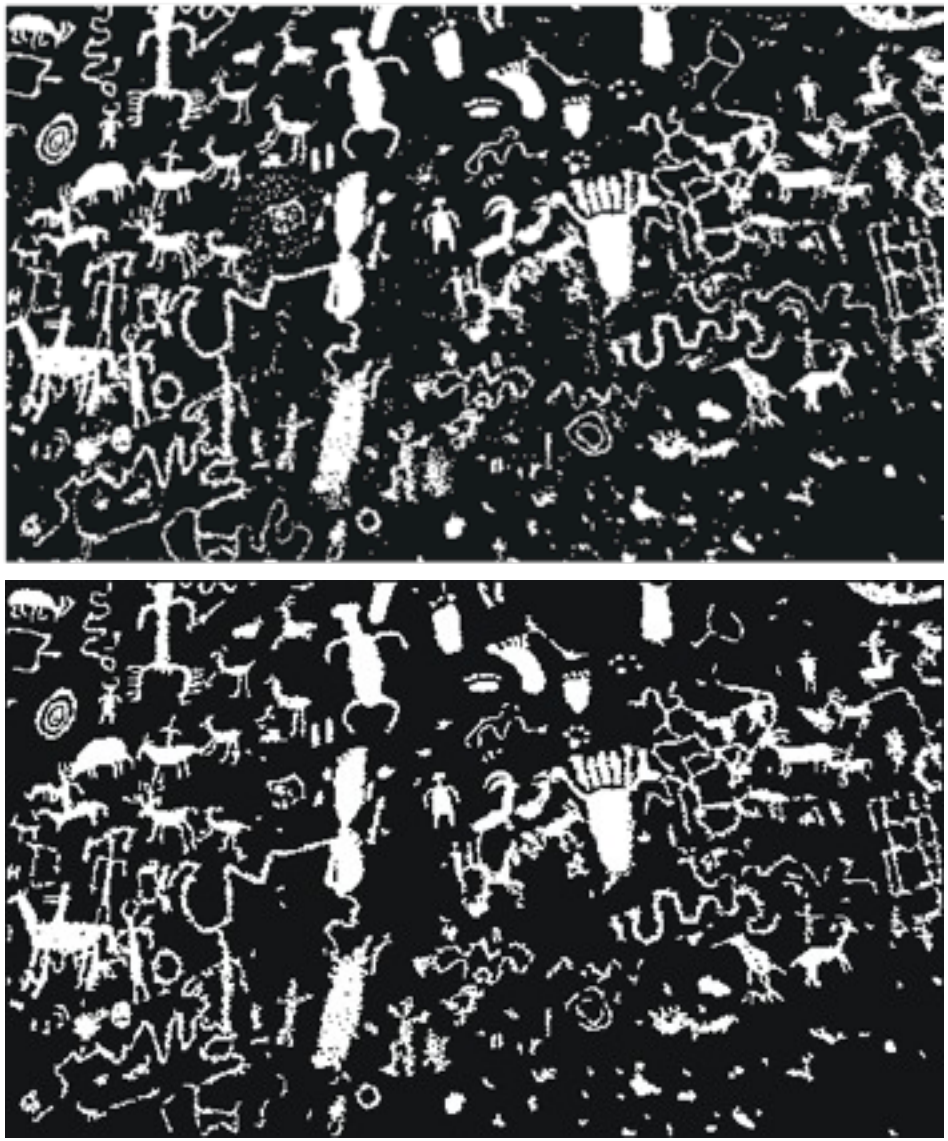
Wieso geht das überhaupt?
→ siehe Def. Medianfilter







Entfernung kleiner Gebiete



Relaxation Labeling

- Jedes Pixel erhält jedes Label mit einer vorgegebenen Wahrscheinlichkeit

  (Bsp.: Schwellenwertresultat, Wahrscheinlichkeiten „70% weiß“ und „30% schwarz“ für weiße Pixel; umgekehrt für schwarze Pixel)
- Benachbarte kompatible Pixel unterstützen sich.
- Relaxationsprozess: Zuordnungswahrscheinlichkeiten ändern sich mit dem Maß der Unterstützung.
- Zu definieren:
  - Kompatibilität
  - Einfluß der Kompatibilität auf die Labelwahrscheinlichkeiten.

Labelwahrscheinlichkeit

- Aufzählung aller Pixel in Liste $p_0, p_1, \ldots, p_{NM-1}$.
- K verschiedene Label
- Initiale Labelwahrscheinlichkeit P^0 für jedes Pixel, und jedes Label l_k vergeben, z.B.

$$P^0(p_i, l_k) = \begin{cases} 0.8 & , \text{falls } l_k = l(p_i) \\ 0.2 & , \text{sonst} \end{cases}$$

Labelwahrscheinlichkeit darf nicht 0 oder 1 sein (Gewissheiten werden nicht verändert)

Kompatibilität

- Ein Pixel p_i mit Label l_k hat eine Kompatibilität r mit einem Pixel p_j, dessen Label l_l sei:

$$r((p_i, l_k), (p_j, l_l))$$

- Kompatibilitätskoeffizient für Binärbilder (K=2 Label) z.B.

$$r((p_i, l_k), (p_j, l_l)) = r(l_k, l_l) = \begin{cases} 1 & , \text{falls } l_k = l_l \\ 0 & , \text{sonst.} \end{cases}$$

(d.h., nur gleiche Label unterstützen sich)

Unterstützung eines Pixels

- Unterstützung $q^{(n)}$ von Pixel p_i durch Pixel p_j zur Iteration n

$$q_j^{(n)}(p_i, l_k) = \sum_{l=0}^{K-1} P^{(n)}(p_j, l_l) \cdot r((p_i, l_k), (p_j, l_l))$$

- Unterstützung von p_i durch alle Pixel

$$Q^{(n)}(p_i, l_k) = \sum_{j=0}^{NM-1} c_{ij} q_j^{(n)}(p_i, l_k)$$

mit Einflussparameter c_{ij}, z.B.

$$c_{ij} = \begin{cases} 1/8 & , \text{falls } p_j \in N_8(p_i) \\ 0 & , \text{sonst.} \end{cases}$$

Iterationsschritt

$$P^{(n+1)}(p_i, l_k) = \frac{P^{(n)}(p_i, l_k)\left[1 + Q^{(n)}(p_i, l_k)\right]}{\sum_{l=0}^{K-1} P^{(n)}(p_i, l_l)\left[1 + Q^{(n)}(p_i, l_l)\right]}$$

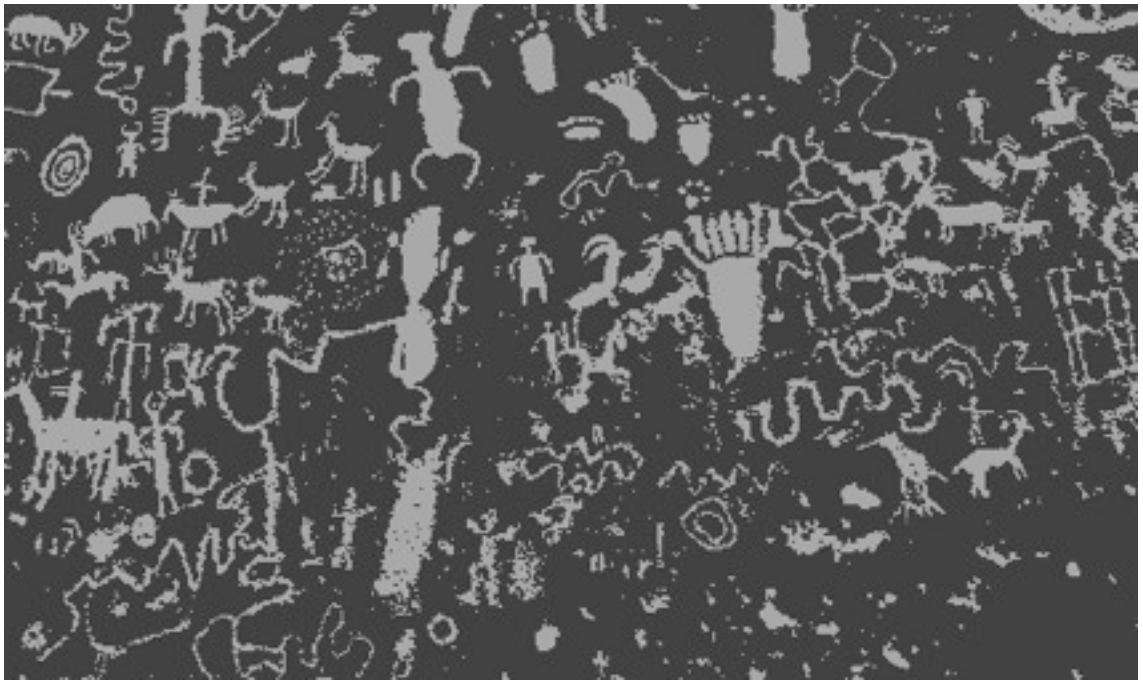




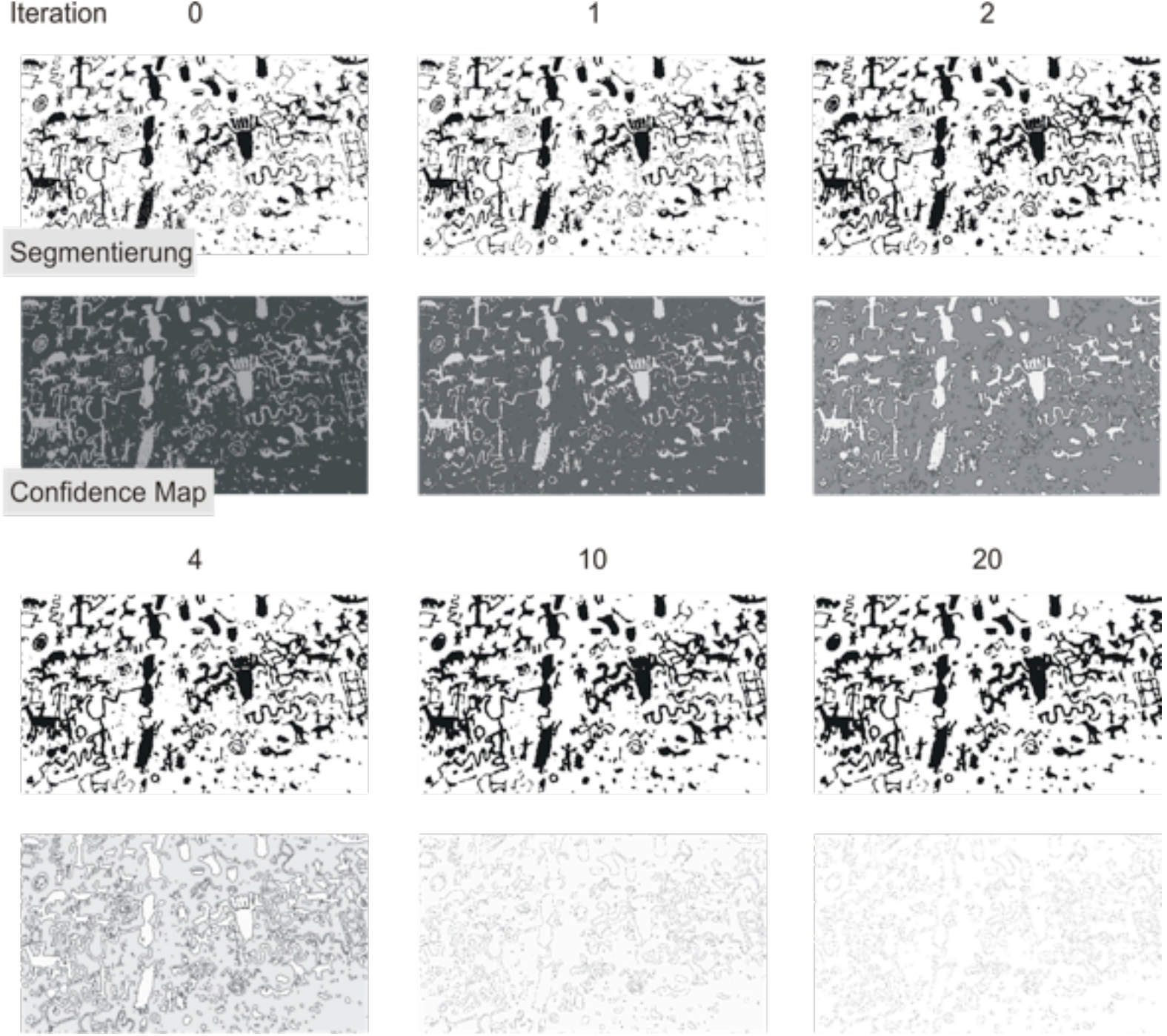
Confidence Map

- Um das Konvergenzverhalten zu beobachten, kann eine **Confidence Map** erzeugt werden.
- Confidence Map: Gibt für jedes Pixel die Zuverlässigkeit der derzeitigen Entscheidung an (weiss = sicher)
- Für 2-Label-Segmentierung: Differenz zwischen gewählten Label und nicht gewähltem Label (evtl. gewichtet mit der Anzahl der Pixel mit diesem Label).

Initiale Confidence Map



Konvergenz



..und nochmal kurz zusammengefasst:

- Zuerst:
  - Schwellenwertsegmentierung
  - Shadingkorrektur
- Dann:
  - Region Labeling
  - Nachverarbeitung
  - Relaxation Labeling

Regionenbasierte Segmentierung

- Multiskalenstrategien
- Region Merging
- Split-and-Merge
- Textursegmentierung

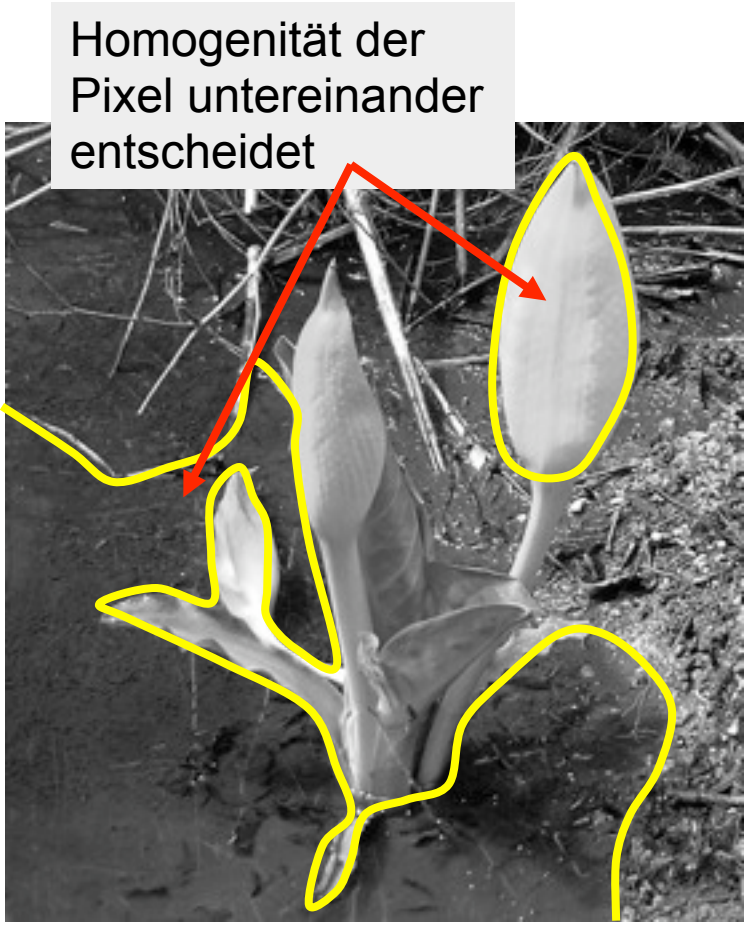
Regionenbasierte Segmentierung

Homogenität im Inneren des Segments.

Homogenitätsbedingung wird bei der Segmentierung ausgewertet.

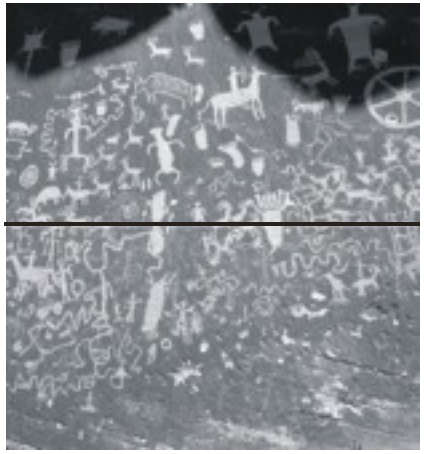
Homogenität ist relativ zu den Attributen eines Segments definiert.

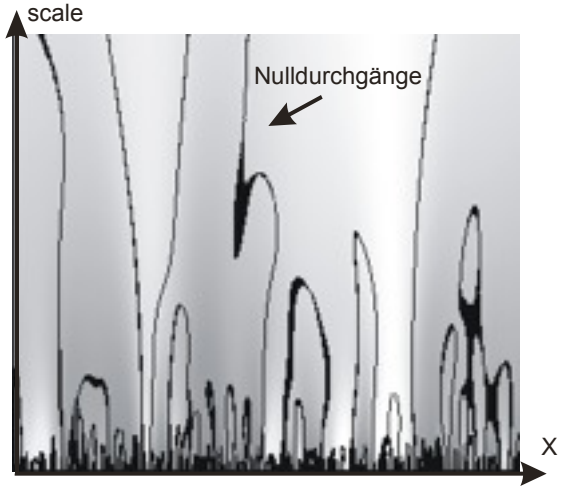
Globale Zusammenhänge über Multiskalenstrategie.



Multiskalenstrategie

- Relative Kriterien für Homogenität können über unterschiedliche Entfernungen verschieden wirken.
- Segmentierung nach Multiskalenstrategie wertet Kriterien auf unterschiedlichen Skalierungen aus.
- Multiskalenstrategie
  - explizit auf einer Multiskalenrepräsentation
  - implizit in den Segmentierungsalgorithmus integriert.





Gaußpyramide

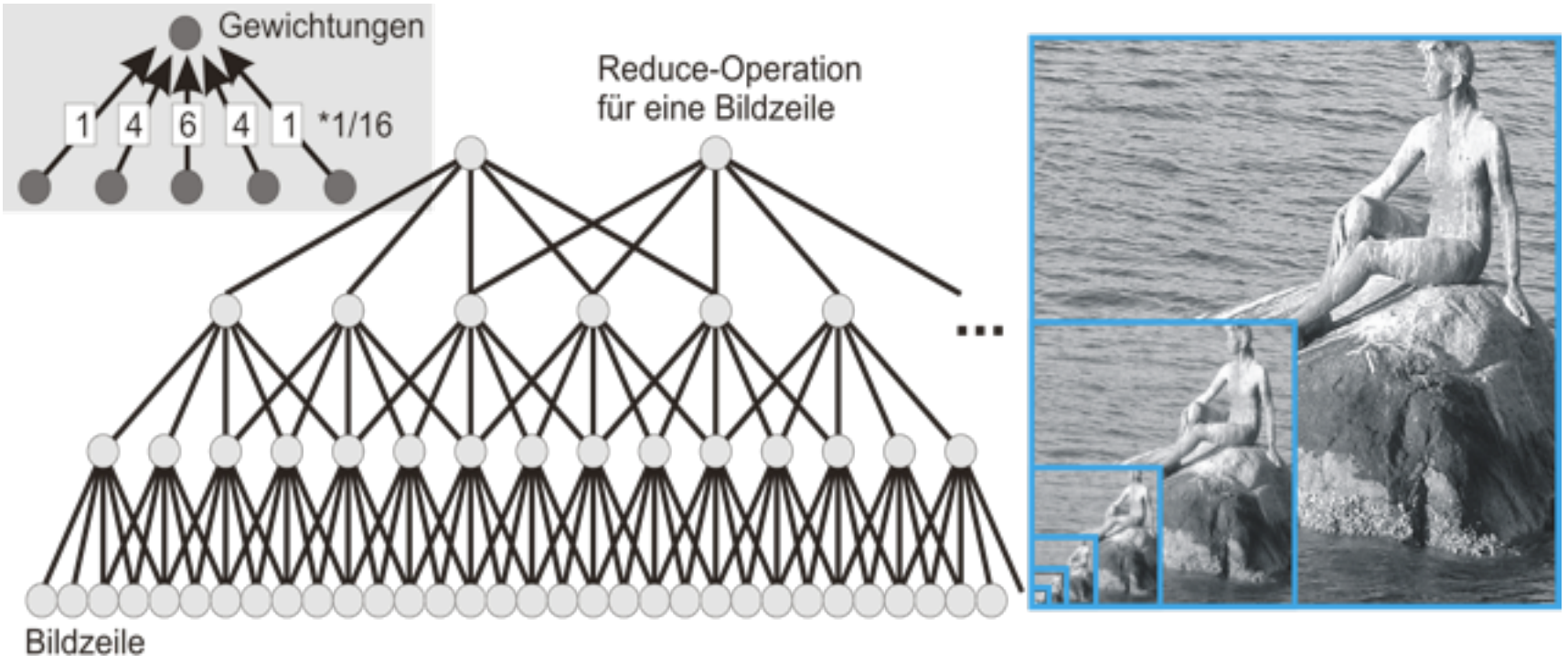
- Das Originalbild wird fortlaufend durch eine „reduce“-Operation skaliert.
- Jedes Pixel der nächsthöheren Skalierungsstufe repräsentiert 4 Pixel der aktuellen Stufe.
- Vor der Reduktion wird der Frequenzumfang durch Filterung vermindert:
  - Gaußfilter

$$\frac{1}{16}\begin{pmatrix}0.87 & 3.91 & 6.44 & 3.91 & 0.87\end{pmatrix}$$

  - Binomialfilter

$$\frac{1}{16}\begin{pmatrix}1 & 4 & 6 & 4 & 1\end{pmatrix}$$

Gaußpyramide



Expand-Operation

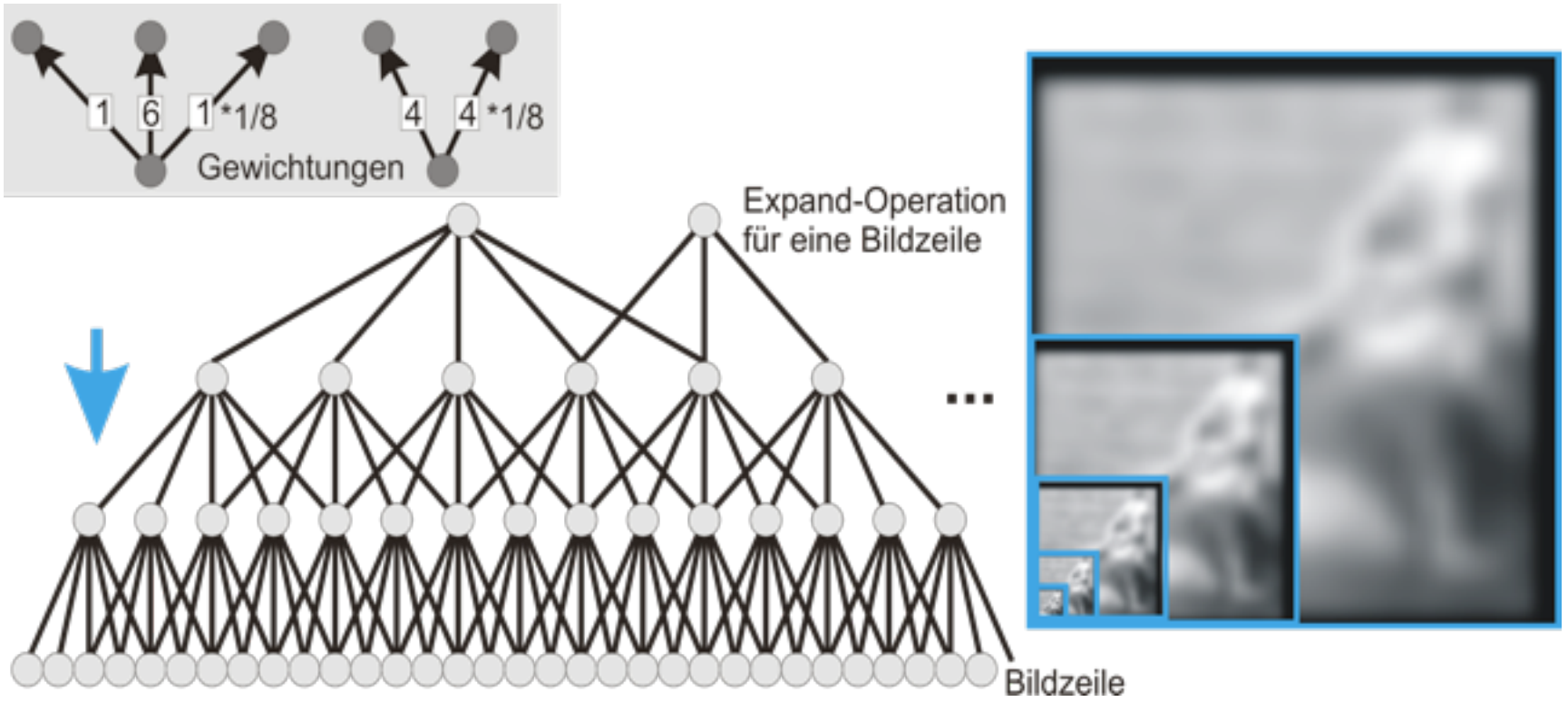
- Um die vorherige Skalierungsstufe zu erzeugen, wird eine „expand“-Operation definiert.
- Pixel der neuen Skalierungsstufe werden durch Interpolation erzeugt:
  - Pixelorte, die auf beiden Skalierungsstufen existieren:

$$\frac{1}{8.18}\begin{pmatrix}0.87 & 6.44 & 0.87\end{pmatrix} \text{ bzw. } \frac{1}{8}\begin{pmatrix}1 & 6 & 1\end{pmatrix}$$

  - Pixelorte, die nur auf der vorherigen Skalierungsstufe existieren:

$$\frac{1}{7.82}\begin{pmatrix}3.91 & 3.91\end{pmatrix} \text{ bzw. } \frac{1}{8}\begin{pmatrix}4 & 4\end{pmatrix}$$

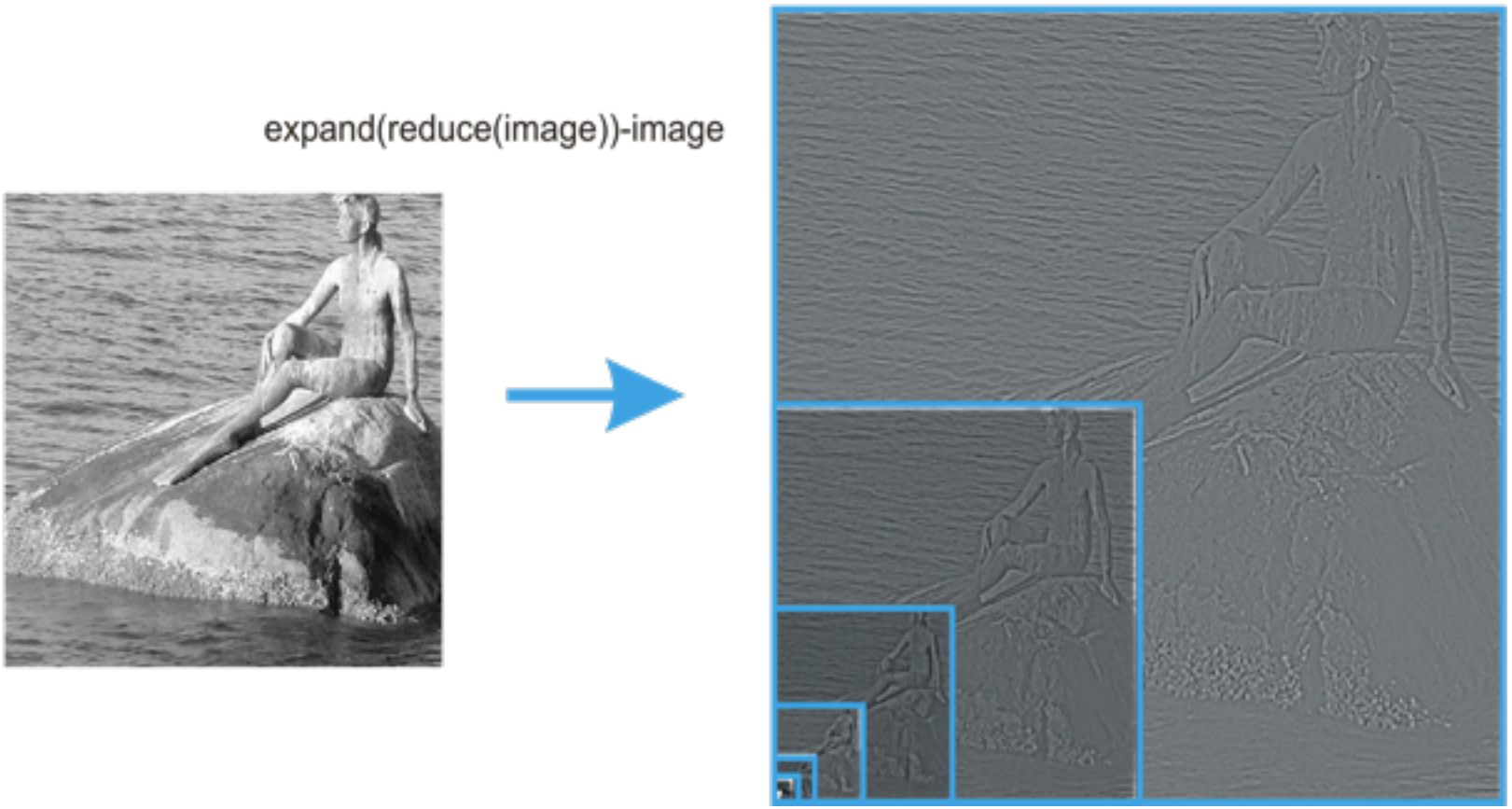
Expand-Operation



Die Expand-Operation ist nicht verlustfrei

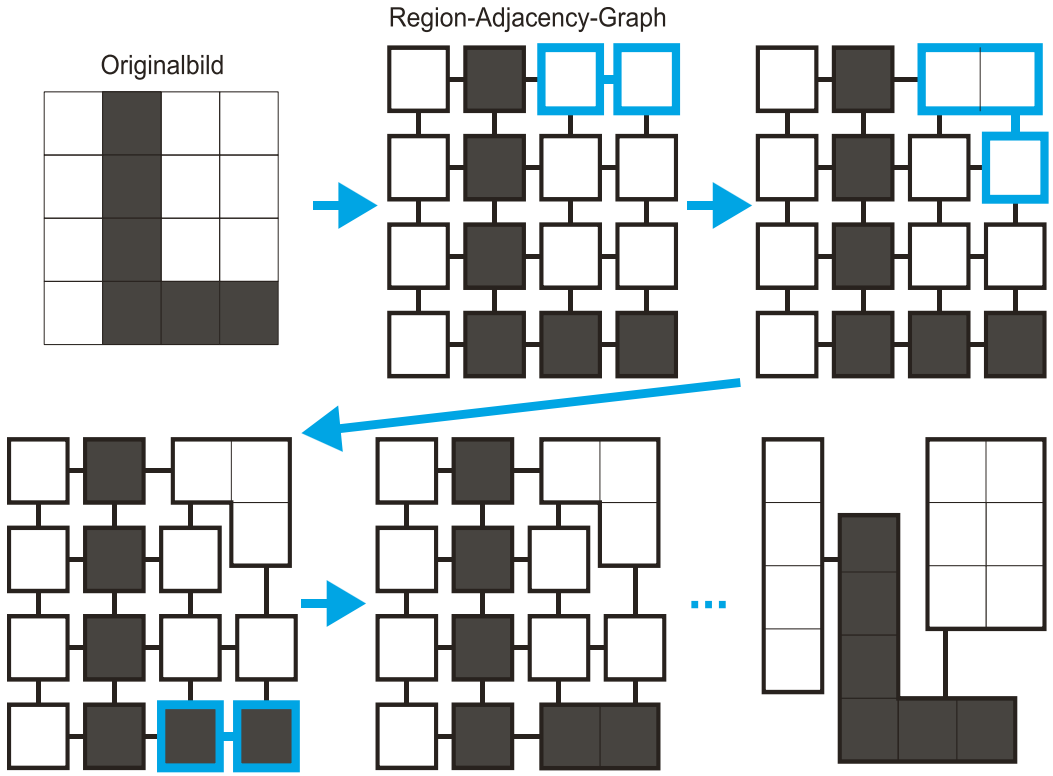
Laplace-Pyramide

- Jede Skalierungsstufe s enthält nur den Unterschied f_s-expand(reduce(f_s))
- Redundanzfreie Repräsentation



Region Merging

- Initial wird jedes Pixel zu einem Segment erklärt.
- Zwei benachbarte Regionen werden zusammengefasst, wenn sie auch gemeinsam das Homogenitätskriterium erfüllen.
- Die Segmentierung ist beendet, wenn keine zwei Regionen mehr existieren, die zusammengefasst werden können.
- Zwischenergebnisse werden in einem Region Adjacency Graph (RAG) gespeichert.



Region Merging

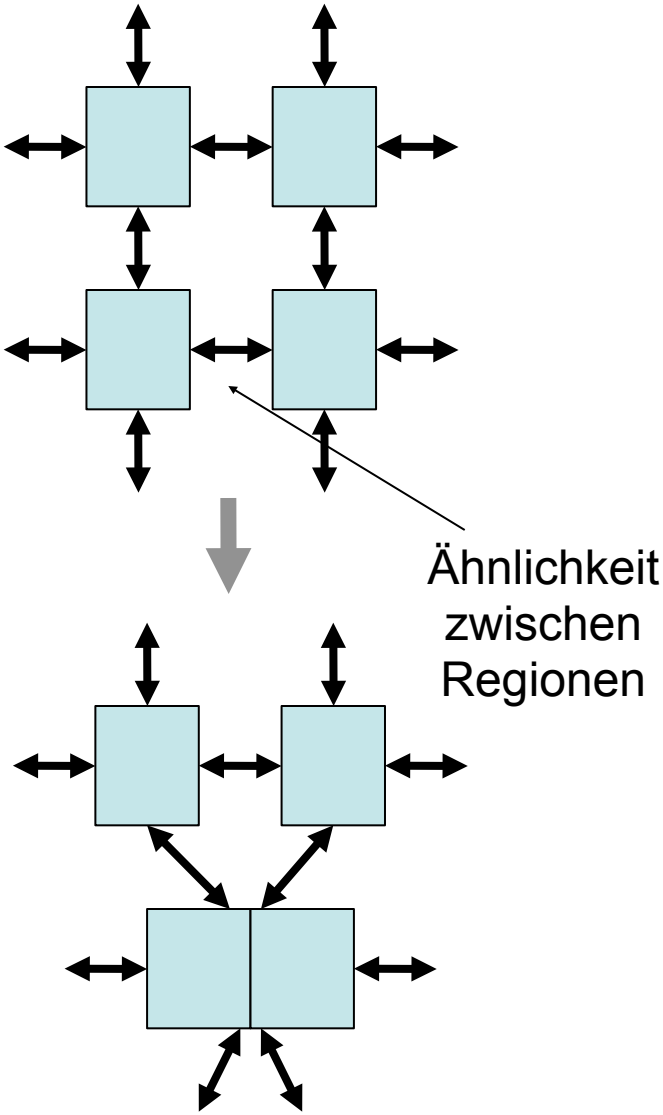
„von Pixeln zu Regionen“:

```
stopMerge = false
while not stopMerge do
   (r1,r2) =
   MaxSimilarity(region)
   if sim(r1,r2)>T then
            region.merge(r1,r2)
   else
            stopMerge=true
```

Bsp. f. Ähnlichkeitskriterium:

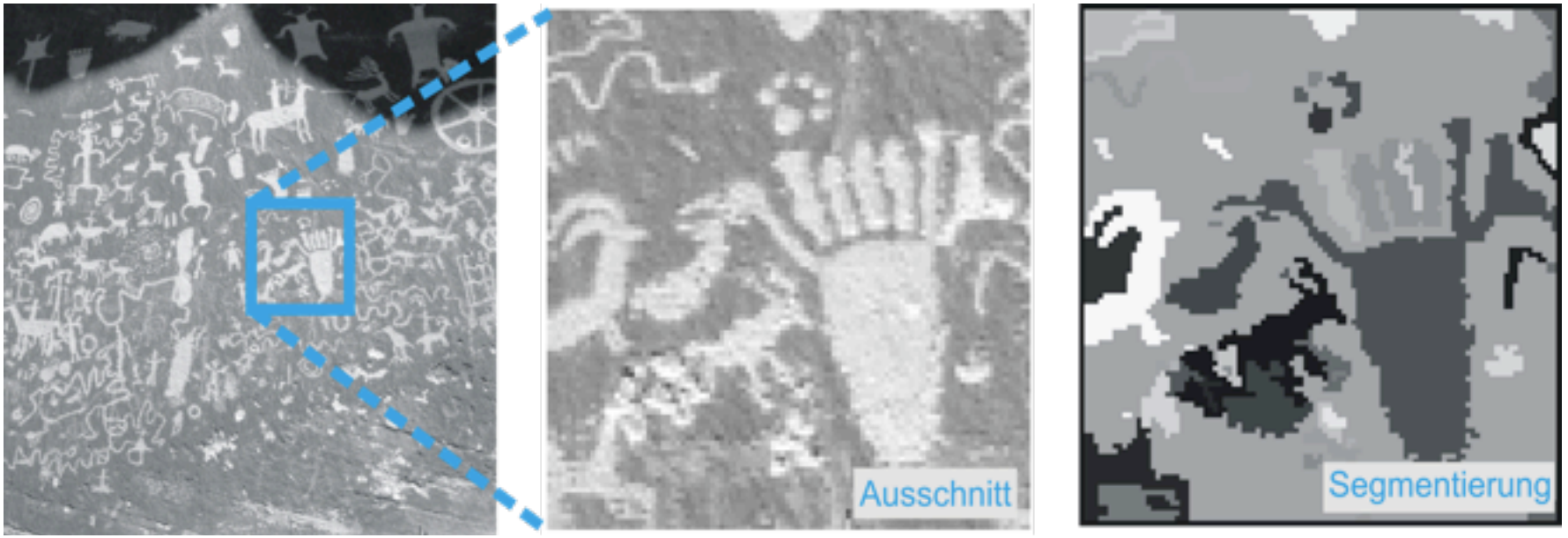
maximaler Grauwertunterschied zwischen Pixeln von r1 und r2.

Region Labeling kann in den Prozess integriert werden.



Region Merging

- Homogenitätskriterium
  - Grauwertdifferenz innerhalb der Region.
  - Wahrscheinlichkeit, dass beide Regionen die gleichen Grauwertverteilungen haben.



Region Merging und Multiskalenstrategie

Modellannahme:

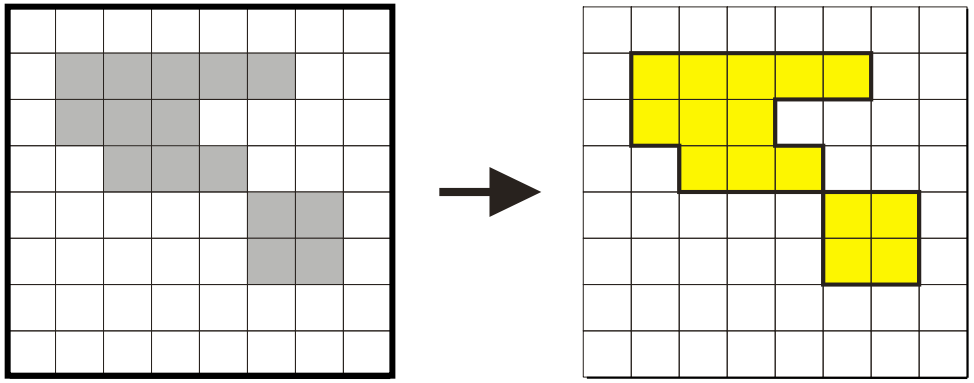
Die gröbste Skalierungsstufe, auf der sich segmentierungs-relevante Eigenschaften manifestieren, ist bekannt.

Prozess:

- Region Merging auf grober Skalierung
- Übertragung des Resultats auf die nächstfeinere Stufe.
- Alle Pixel, die zu Pixeln eines anderen Segments benachbart sind, werden nochmals geprüft.
- Verfahren endet, wenn die die feinste Skalierungsstufe erreicht ist

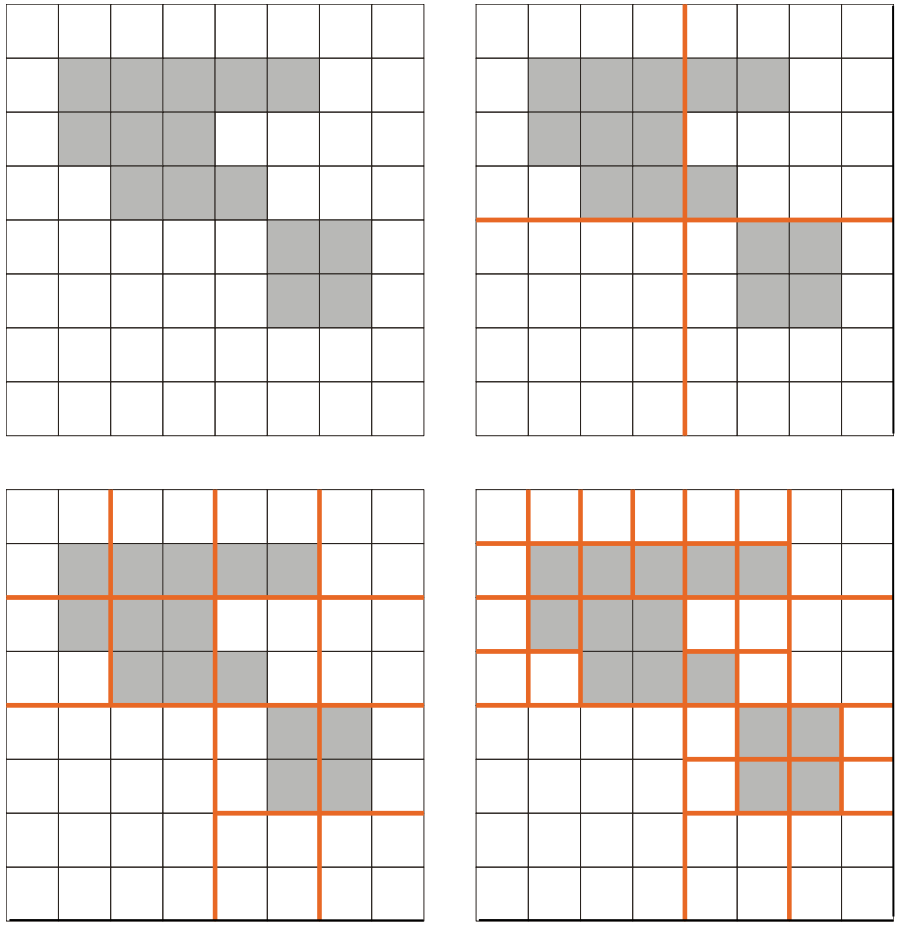
Split & Merge-Algorithmus

Regionenbasiertes Verfahren



- Startbedingung: Das gesamte Bild ist ein Segment.
- Ein Segment wird solange in 4 Untersegmente zerlegt, wie es ein gegebenes Homogenitätskriterium nicht erfüllt
- Benachbarte Segmente werden zusammenfasst, wenn sie auch nach der Zusammenfassung das Homogenitätskriterium erfüllen.
- Resultat ist eine vollständige, überdeckungsfreie Zerlegung des Bildes (Segmentierung gemäß Definition)

Zerlegungsschritt

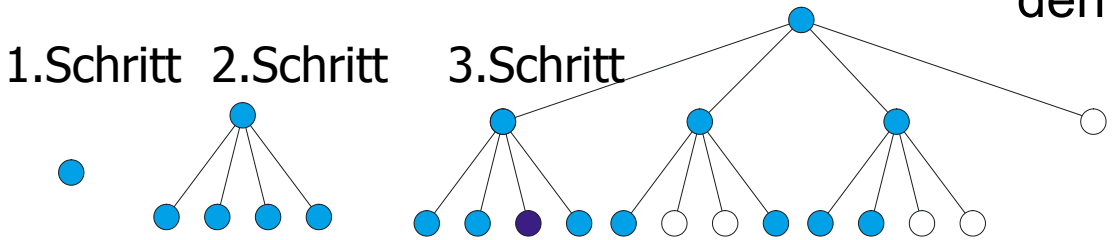


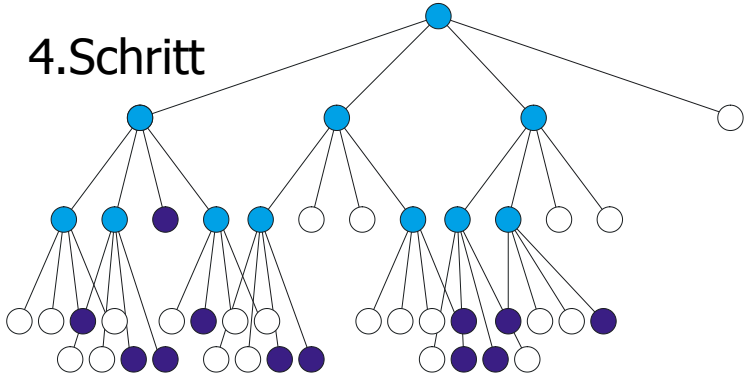
Zerlegung terminiert spätestens auf Pixelebene.

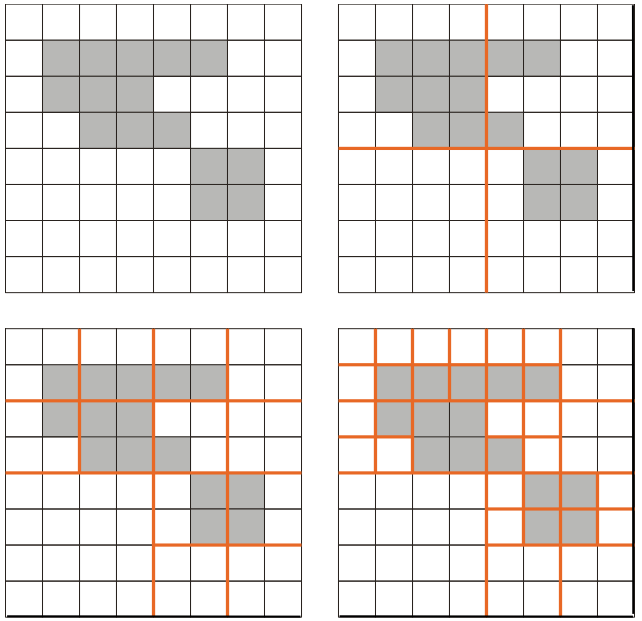
Problem: Datenstruktur zur Dokumentation der aktuellen Zerlegung

Zerlegungsschritt (Quad-Tree Repräsentation)

Wert des Homogenitätsmerkmals einer Region wird im entsprechenden Blatt des Quad-Tree abgelegt



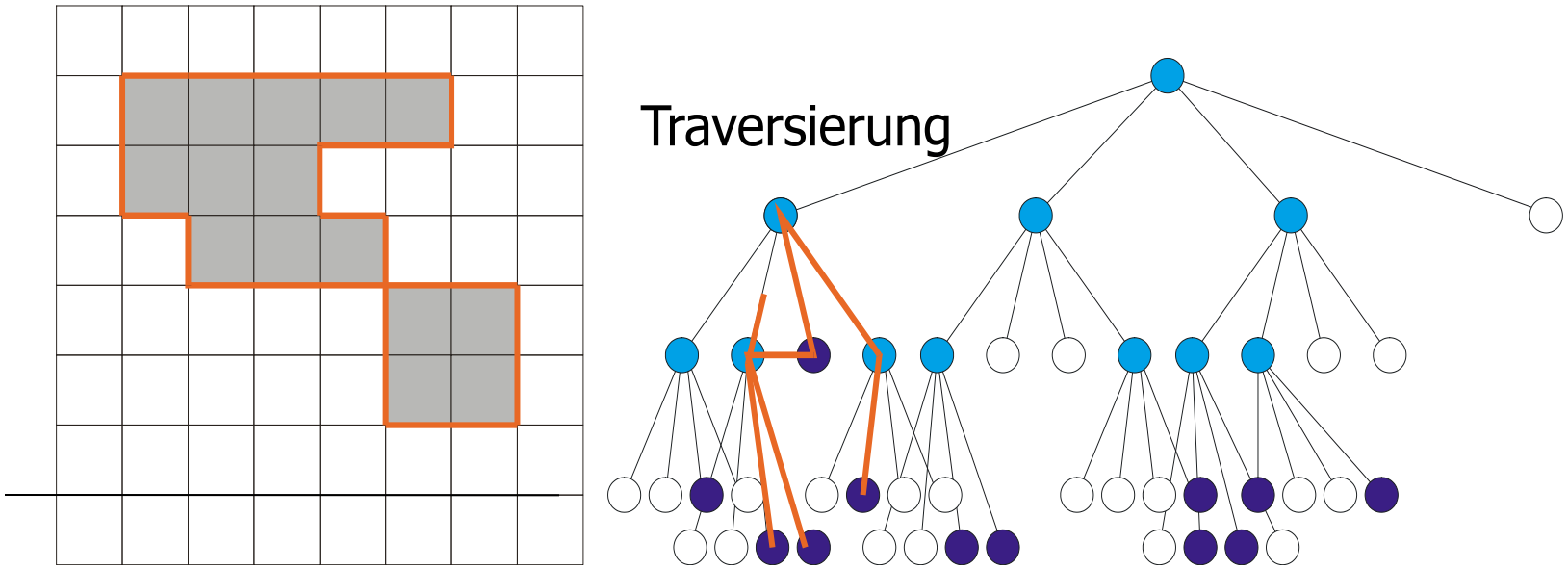




Merging

Quadtree wird traversiert und in einen RAG überführt.

Auf dem RAG wird ein Region Merging durchgeführt.



Split & Merge

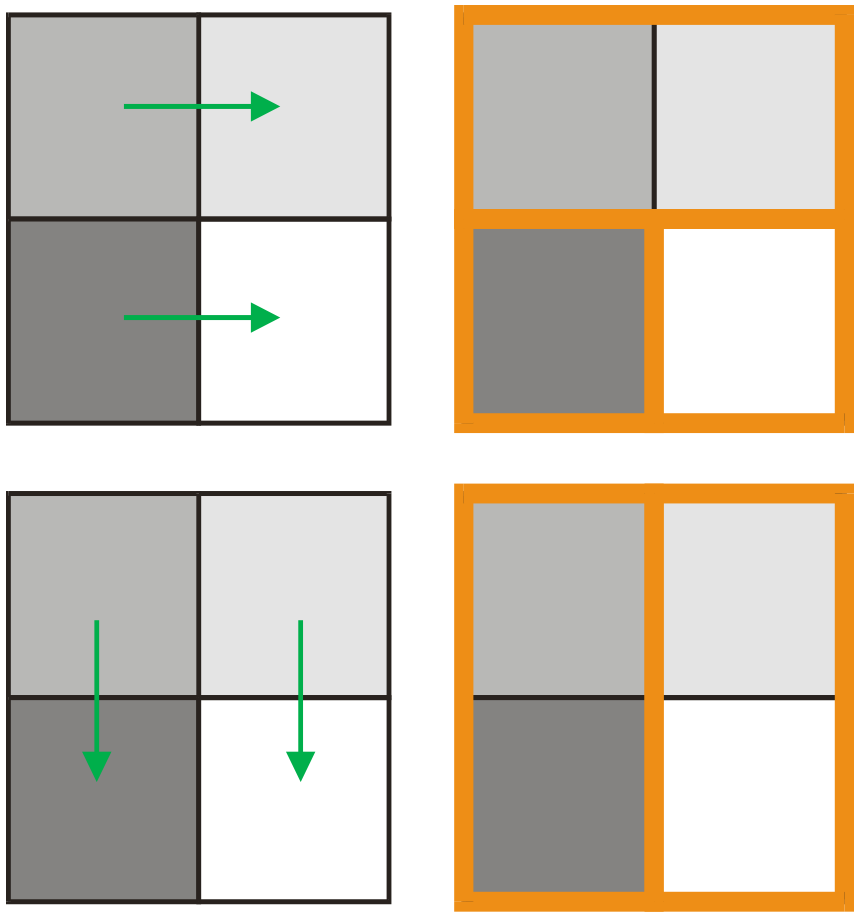
Resultat: Zerlegung des gesamten Bilds in Regionen.

Multiskalenstrategie ist integriert

Homogenitätsmerkmale wie bei Region Merging

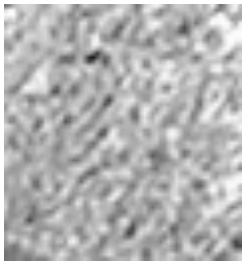
Probleme (Region Merging und Split & Merge):

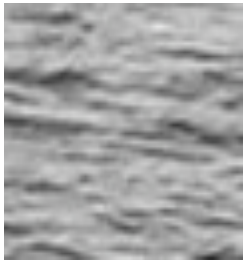
- Merge-Schritt ist bei relativem Homogenitätsmerkmal nicht immer eindeutig
- Minimale Segmentzahl wird nicht immer gefunden



Textur als Homogenitätsmerkmal

- Textur – Musterung der Oberfläche
  - Es existiert keine Definition von Textur
  - Es gibt eine große Anzahl von Texturmaßen
  - Textur ist inhärent skalenabhängig
  - Textur ist eine Eigenschaft einer Gruppe von Pixeln.
- Texturmaße
  - strukturell (Zusammensetzung aus Texturelementen – texel)
  - stochastisch (eine charakterisierbare Grauwertverteilung)
  - spektral (charakteristische Frequenzattribute)

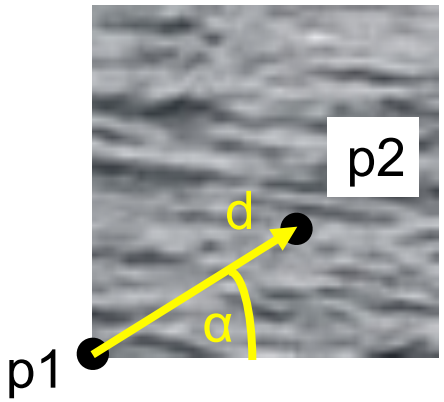


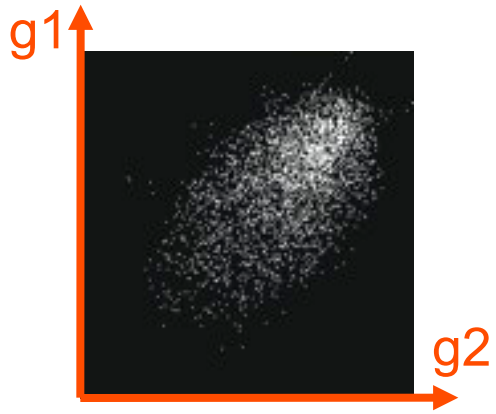



Haralick´sche Texturmaße

- **Co-Occurrence-Matrix =** Zweidim. Histogramm für Pixelpaare.

  Pixel p1 und p2 sind ein Paar, wenn sie Abstand *d* haben und auf einer Linie mit einem gegebenen Winkel α zur *x*-Achse liegen.
- Co-Occurence-Matrix repräsentiert die Korrelation zwischen Pixeln. (Wahrscheinlichkeit P, dass p1 und p2 Grauwerte g1 und g2 haben)
- Meist sind Pixel nicht über große Entfernungen korreliert, daher sind Werte *d*=1, *d*=2 üblich.
- Falls Korrelation über größere Entfernung vermutet wird, dann sollte eine Multiskalenstrategie angewendet werden.

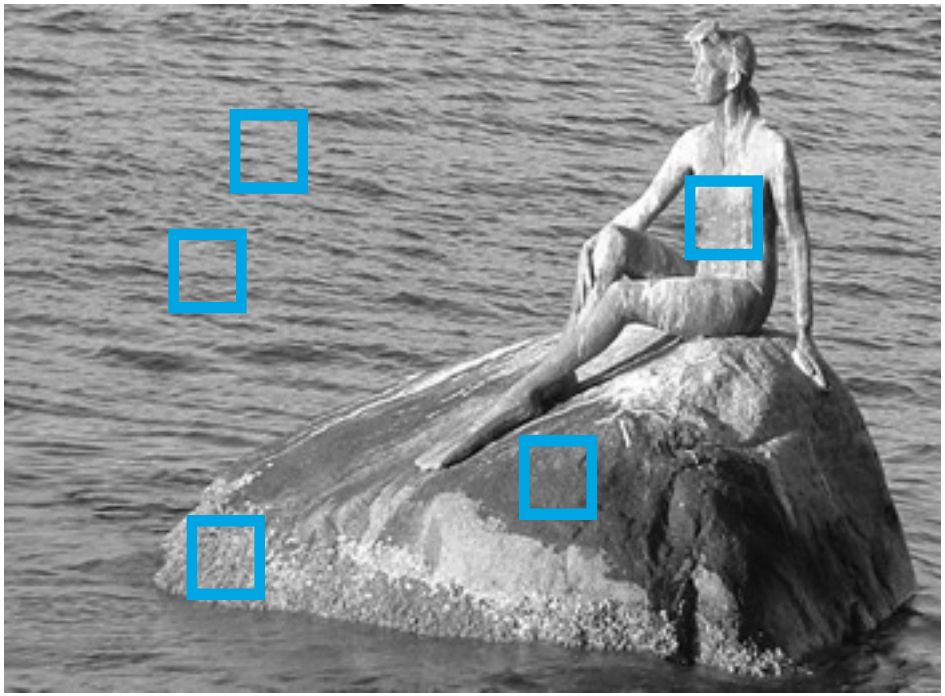


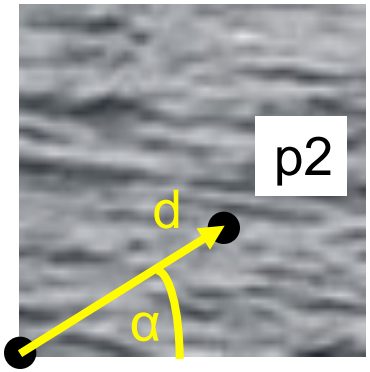


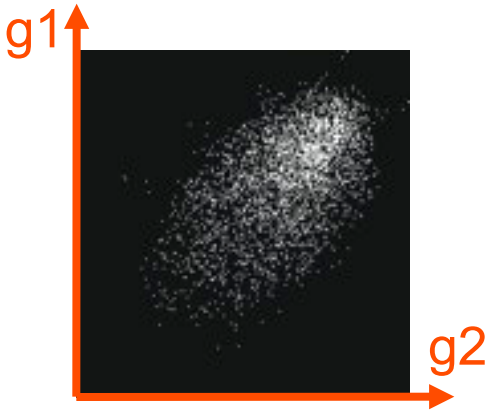


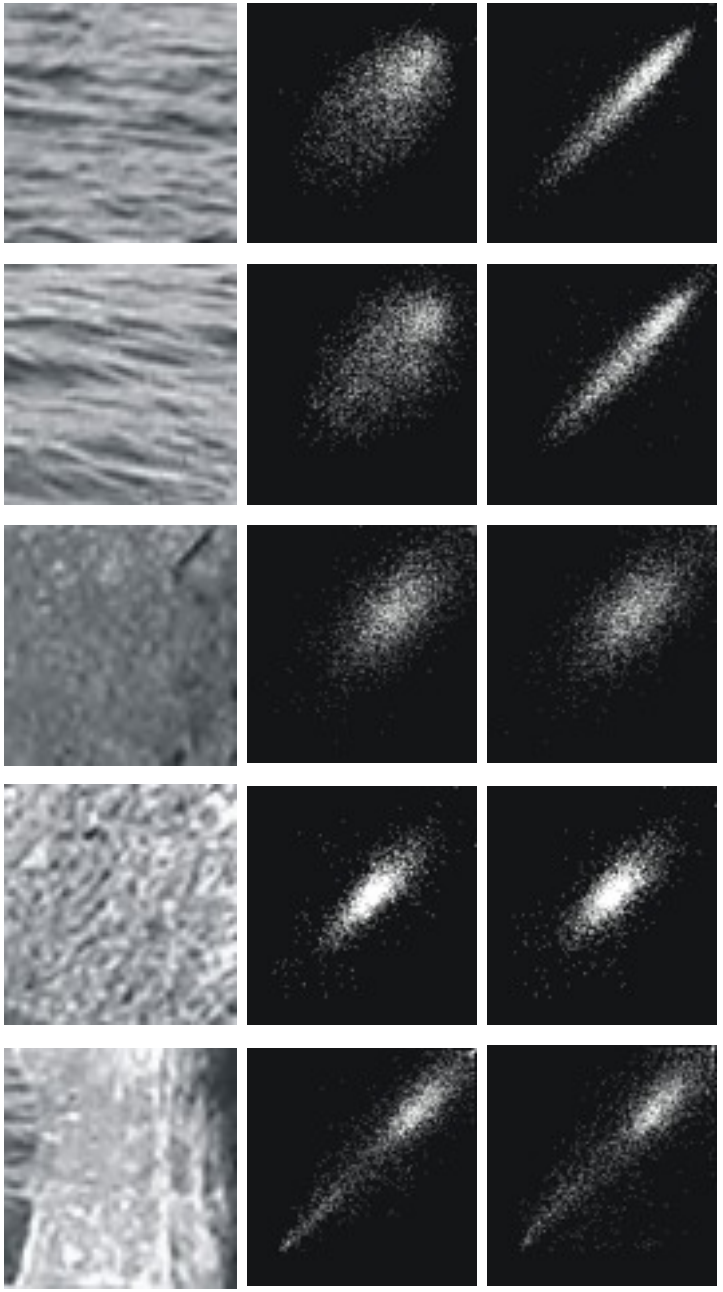
Robert Haralick

Co-Occurrence-Matrix









d=1, α=90°, α=0°

Haralick‘sche Texturmaße

$$\sum_{g_1=0}^{K-1}\sum_{g_2=0}^{K-1} P_{\Delta,\alpha}^2\left(g_1,g_2\right)$$ Energie

$$\sum_{g_1=0}^{K-1}\sum_{g_2=0}^{K-1} \left(g_1-g_2\right)^2 \times P_{\Delta,\alpha}\left(g_1,g_2\right)$$ Kontrast

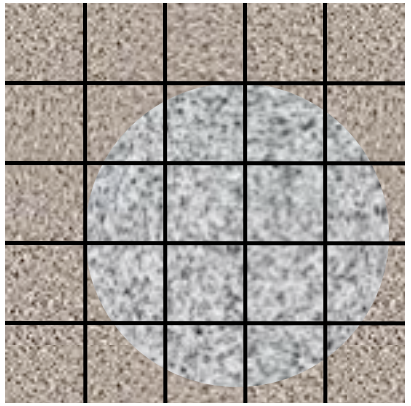
$$\sum_{g_1=0}^{K-1}\sum_{g_2=0}^{K-1} P_{\Delta,\alpha}\left(g_1,g_2\right) \times \log\left[P_{\Delta,\alpha}\left(g_1,g_2\right)\right]$$ Entropie

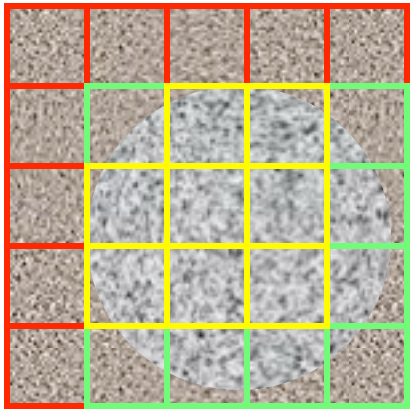
$$\sum_{g_1=0}^{K-1}\sum_{g_2=0}^{K-1} \frac{P_{\Delta,\alpha}\left(g_1,g_2\right)}{1+\left|g_1-g_2\right|}$$ Homogenität

- Lassen sich aus der Co-Occurrence-Matrix berechnen
- Liefern aussagekräftige Kennwerte für verschiedene Texturen

Besonderheiten Texturmerkmal

- Berechnung der Texturmerkmale auf der Basis willkürlicher Regionen.
- Segmentierung
- Berechnung der Zuverlässigkeit
- Erneute Segmentierung





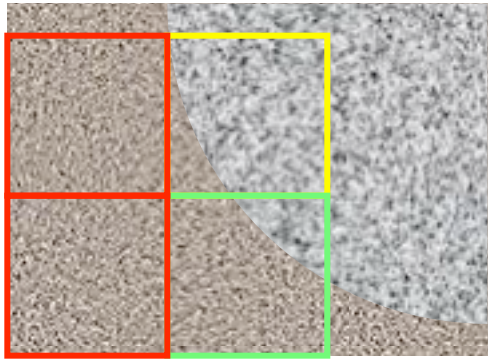
Zuverlässigkeit Texturmerkmal

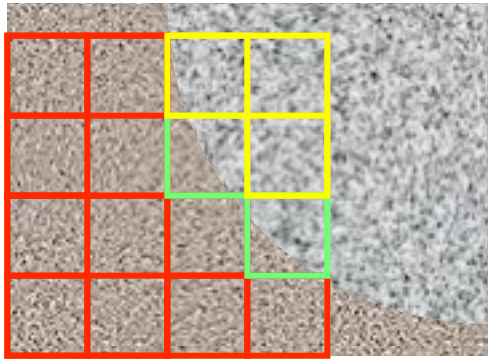
Persistenz:

Falls eine Region in Teile zerlegt wird und das Merkmal in den Teil-regionen berechnet werden kann,

dann sollte das Texturmerkmal in der Teilregion dieselben Werte annehmen, wie das der Ursprungsregion.

Achtung: In der Regel ist das berechnete Maß eine Schätzung, deren Güte von der Regionengröße abhängt.



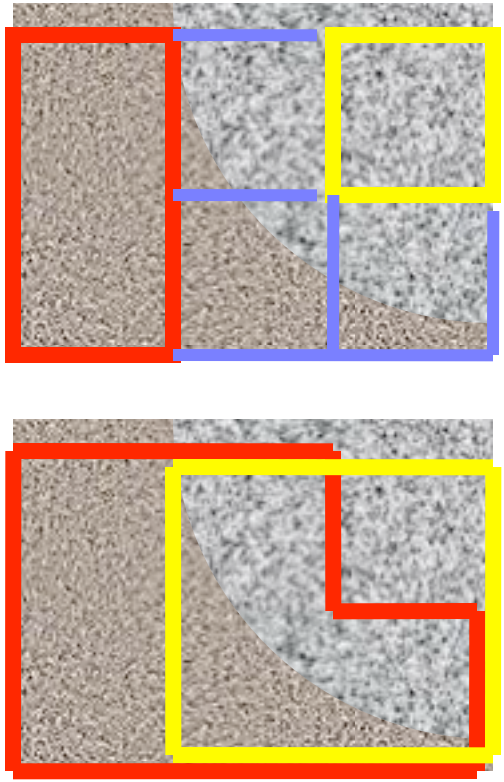


Strategie: Erneute Segmentierung

- Texturmerkmale der Gesamtsegmente erneut berechnen
- Segmente mit hoher Unzuverlässigkeit
  - allen benachbarten zuverlässigen Segmenten zuordnen
  - ähnlichste Kombination selektieren.

  Berechenbarkeit von Texturmerkmalen beliebiger Regionen ist vorteilhaft.

  Kann über mehrere Auflösungsstufen erfolgen.



Die beiden Extremfälle

Naive Textursegmentierung

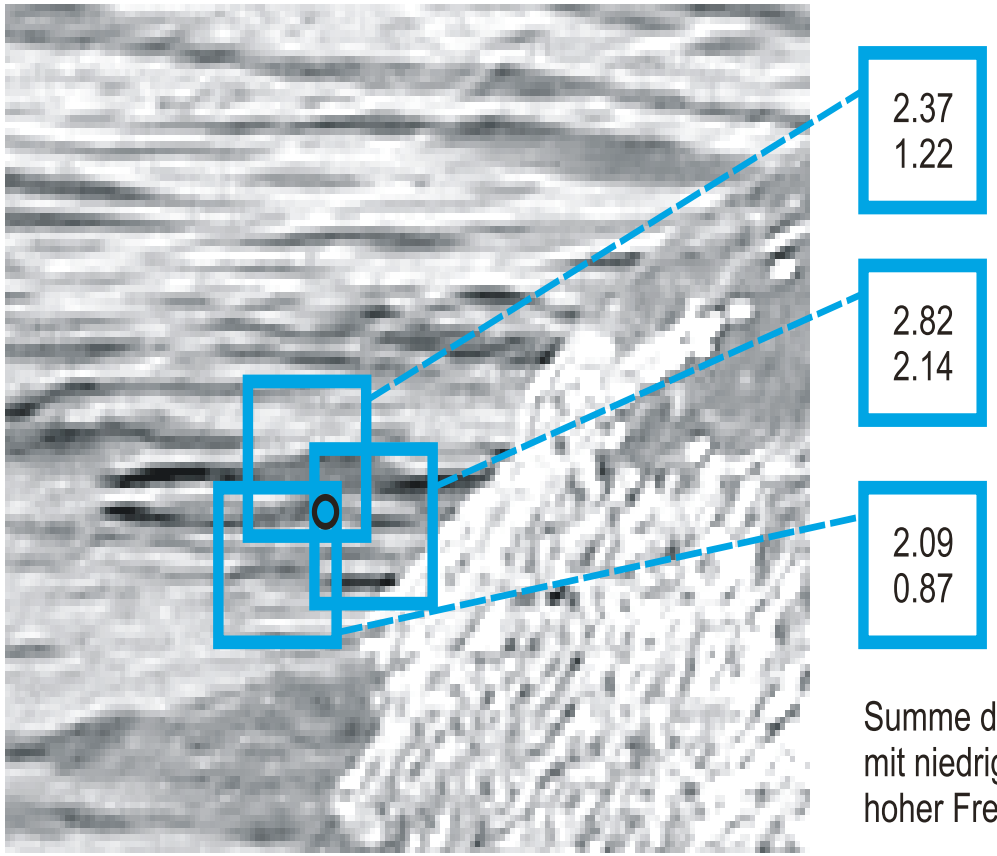
Modell:

- Skalierungsstufe der Textur ist bekannt.
- Blockgröße für die Texturberechnung ist bekannt.
- Geeignetes Texturmaß ist bekannt

Algorithmenskizze:

- Berechne für jedes Pixel das Texturmaß.
- Führe eine homogenitätsbasierte Segmentierung auf den Texturmaßen durch.
- Jedes Pixel erhält für jedes Label, dessen Block das Pixel überdeckt, eine von Null verschiedene Wahrscheinlichkeit.
- Nachverarbeitung ähnlich wie bei Schwellenwertsegmentierung.

Textursegmentierung (naiv)



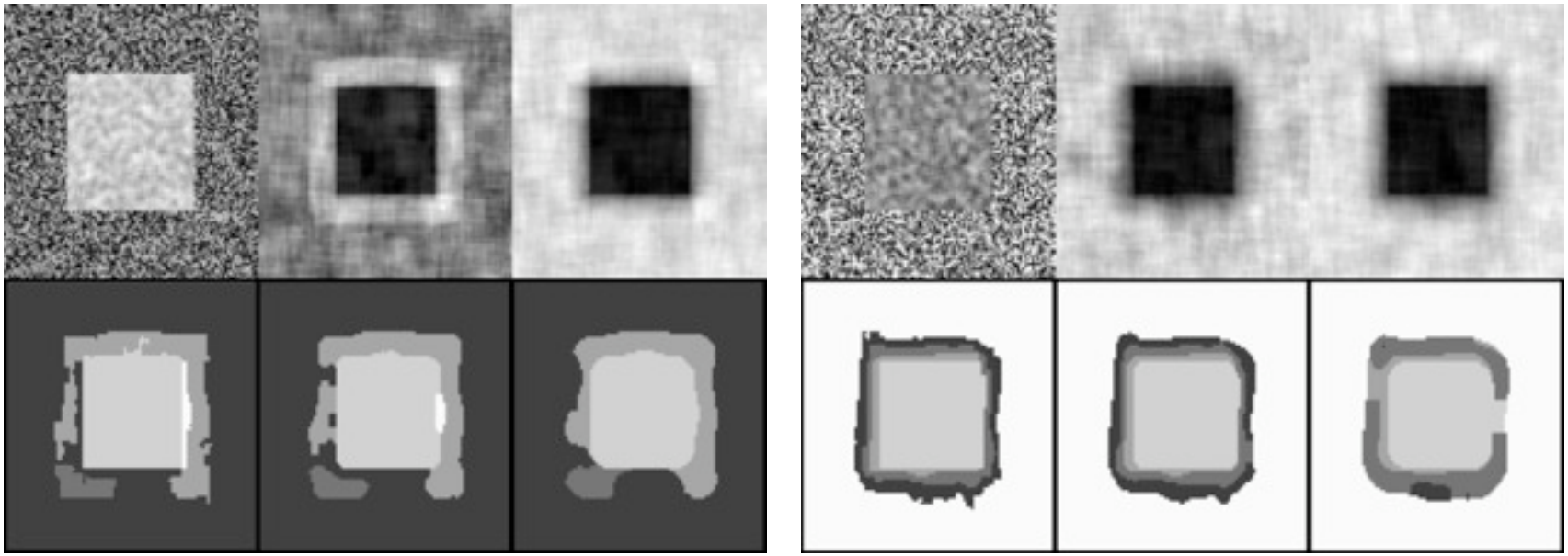
Beispiel:

Jedes Pixel wird von mehreren Blöcken überdeckt.

Sind die Texturen unterschiedlich, dann erhalten diese Blöcke unterschiedliche Label.

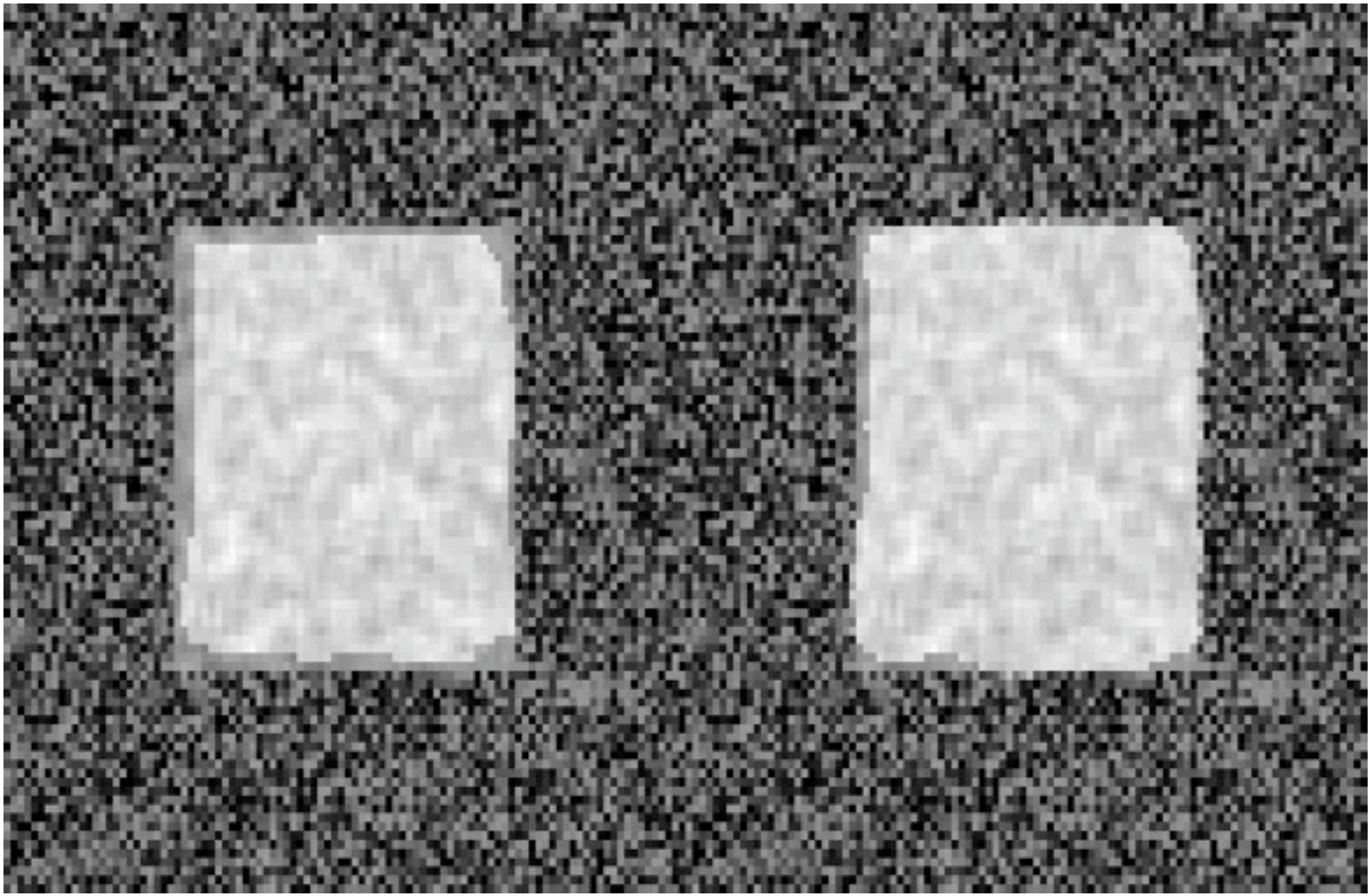
Nachverarbeitung Medianfilterung

Segmentierung von zwei verschiedenen Texturen



Resultat Median 5×5 Median 11×11 Resultat Median 5×5 Median 11×11

Nachverarbeitung Relaxation Labeling



…Also nochmal bis hierhin…

- Multiskalenstrategie
- Regionenbasierte Segmentierung (Region Merging, Split & Merge)
- Datenstrukturen RAG und Quad-Tree
- Texturen und Textursegmentierung

Kantenbasierte Segmentierung

- Edge Linking und Canny Edge Operator
- Nulldurchgänge
- Wasserscheidentransformation

Segmentierung durch Kantenerkennung

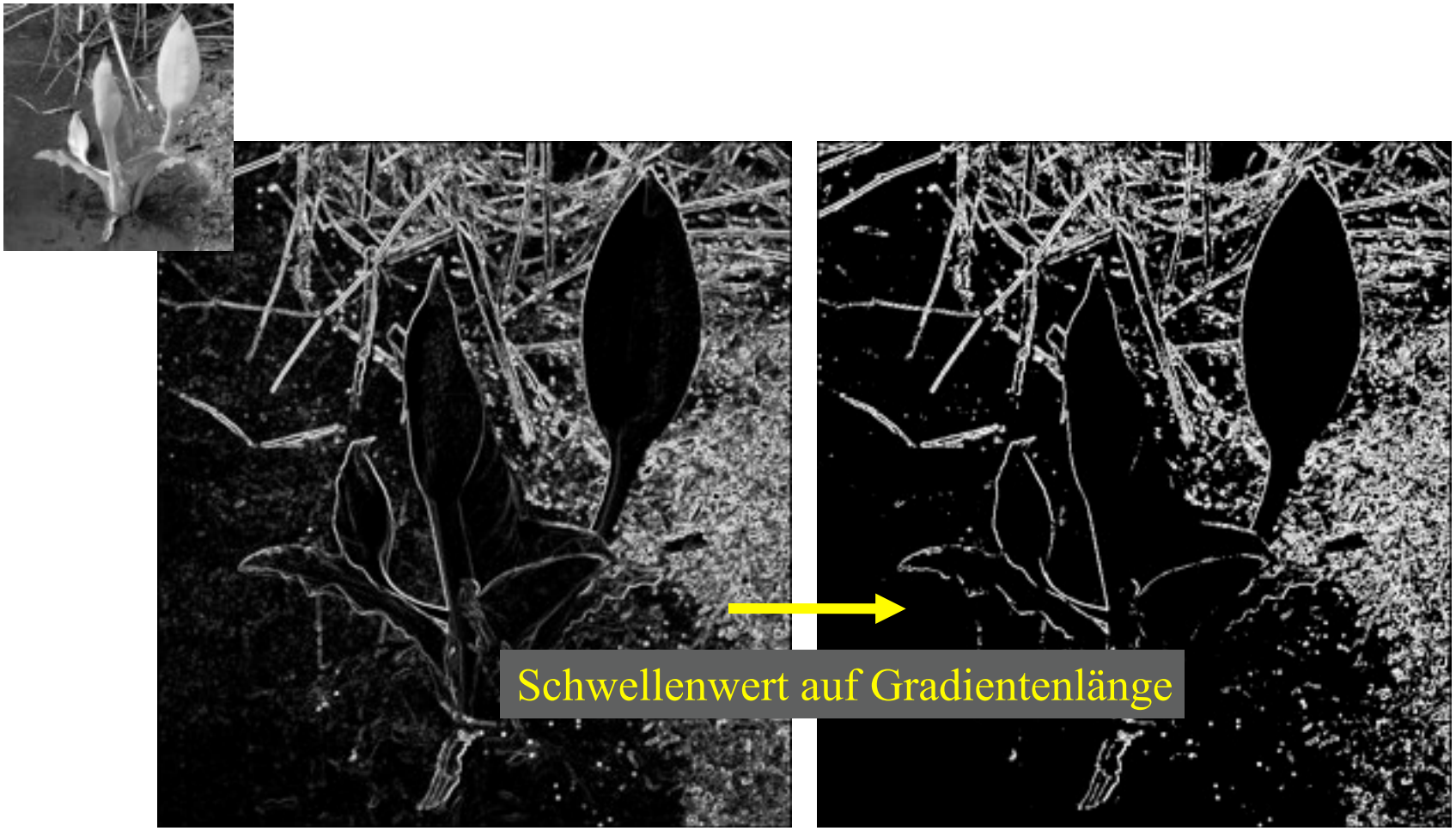


Vorteil: Kantenmerkmale sind robuster gegenüber Shading

Einfache Methode:

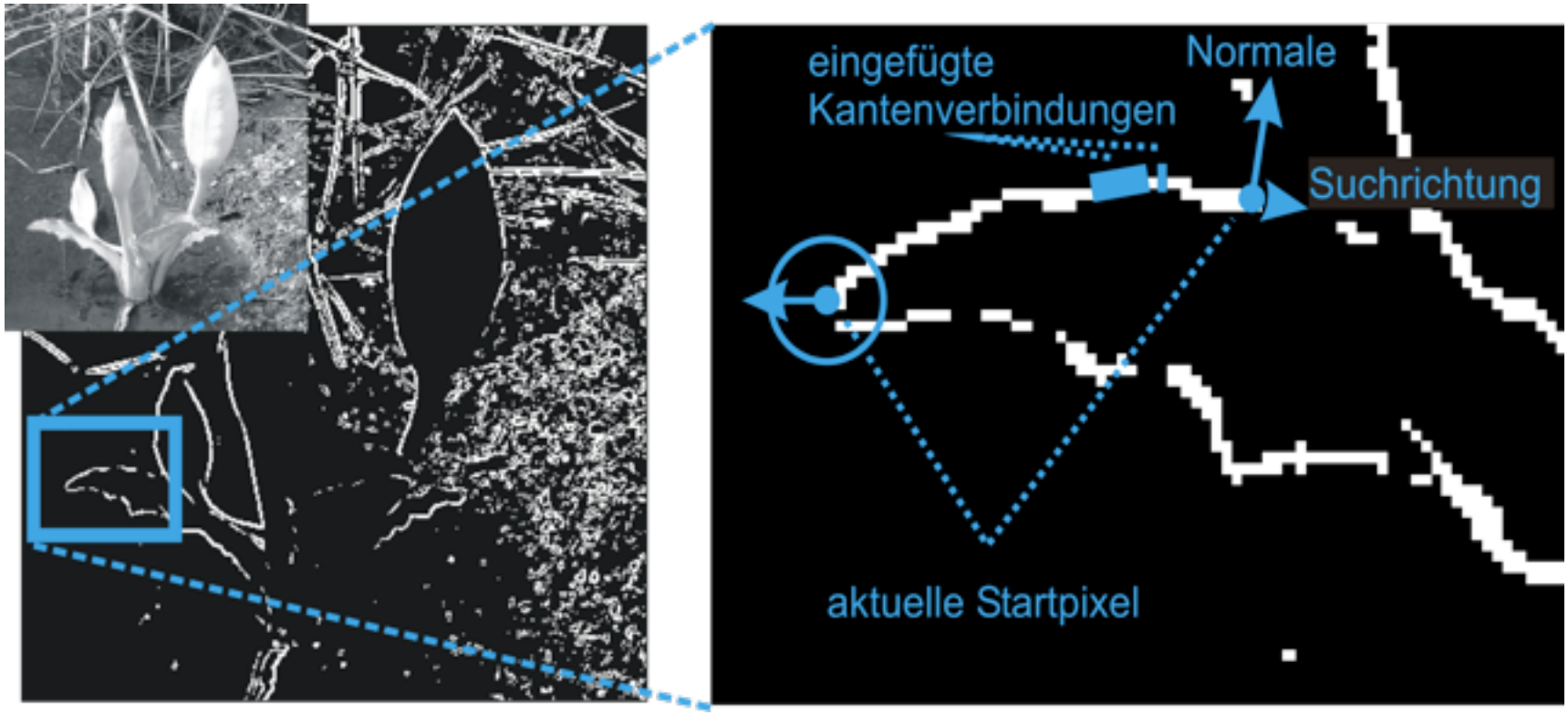
- Gradientenberechnung
- Kantenpunktdetektion (z.B. Schwelle auf Gradientenlänge)
- Region Labeling basierend auf Kantenpunkten.

Segmentierung durch Kantenerkennung



Problem: Kantenpunkte sind nicht Ränder zusammenhängender Gebiete.

Edge Linking



Anfangs sind alle Kantenpixel frei und nicht untersucht.

Edge Linking sucht sich das nächste nicht untersuchte und freie Kantenpixel und versucht es mit anderen Pixeln zu einem Kantenzug zu verknüpfen.

Edge Linking

1. Suche das nächste Kantenpixel, welches noch nicht als „untersucht“ markiert wurde und erkläre es zum Startpixel eines Kantenzugs.
2. Falls sich in der Umgebung des Kantenpixels in einer der beiden Richtungen orthogonal zur Kantenrichtung unmarkierte Kantenpixel befinden, die eine ähnliche Gradientenrichtung und –stärke aufweisen:
   a. Markiere die Pixel als zum selben Kantenzug gehörend.
   b. Erkläre diese Pixel zu neuen Startpixeln.
   c. Gehe zu 2.
3. Falls sich in der Umgebung markierte Pixel befinden, die den obigen Bedingungen genügen, dann wurde eine Verzweigung von Kanten gefunden.
4. Falls ein Kantenpixel gefunden wurde, gehe zurück zu Schritt 1.

Canny Edge Operator

Ziele:

- möglichst viele Kanten fehlerfrei vom Hintergrund unterscheiden zu können (niedrige Rate von Fehldetektionen).
- (unverzweigte) Kanten genau zu lokalisieren.
- für jede Kante genau eine Detektorantwort zu liefern.

Canny Operator besteht aus Kantenhervorhebung und Erzeugung von Kantenzügen.

Optimale Kantenhervorhebung ist eine Filterung mit einer 1-D abgeleiteten Gaußfunktion orthogonal zur Kante.

Geringfügig schlechtere Ergebnisse erzielt man mit einem 2-D Gradientenoperator auf der Basis abgeleiteter Gaußfunktionen.

Canny Edge Operator

- Es wird immer das Pixel mit größter Gradientenlänge selektiert.
- Startpixel können nur Pixel sein, deren Gradientenlänge oberhalb einer Signifikanzschwelle T1 liegt.
- Neue Kantenpixel werden in den Kantenzug eingefügt, wenn ihre Gradientenlänge größer als T2 ist (T2<T1).
- Das Verfahren endet, wenn keine neuen Startpixel gefunden werden.

Nulldurchgänge



Die Orte der Nulldurchgänge der zweiten Ableitung sind Ränder zusammenhängender Gebiete.

Methode:

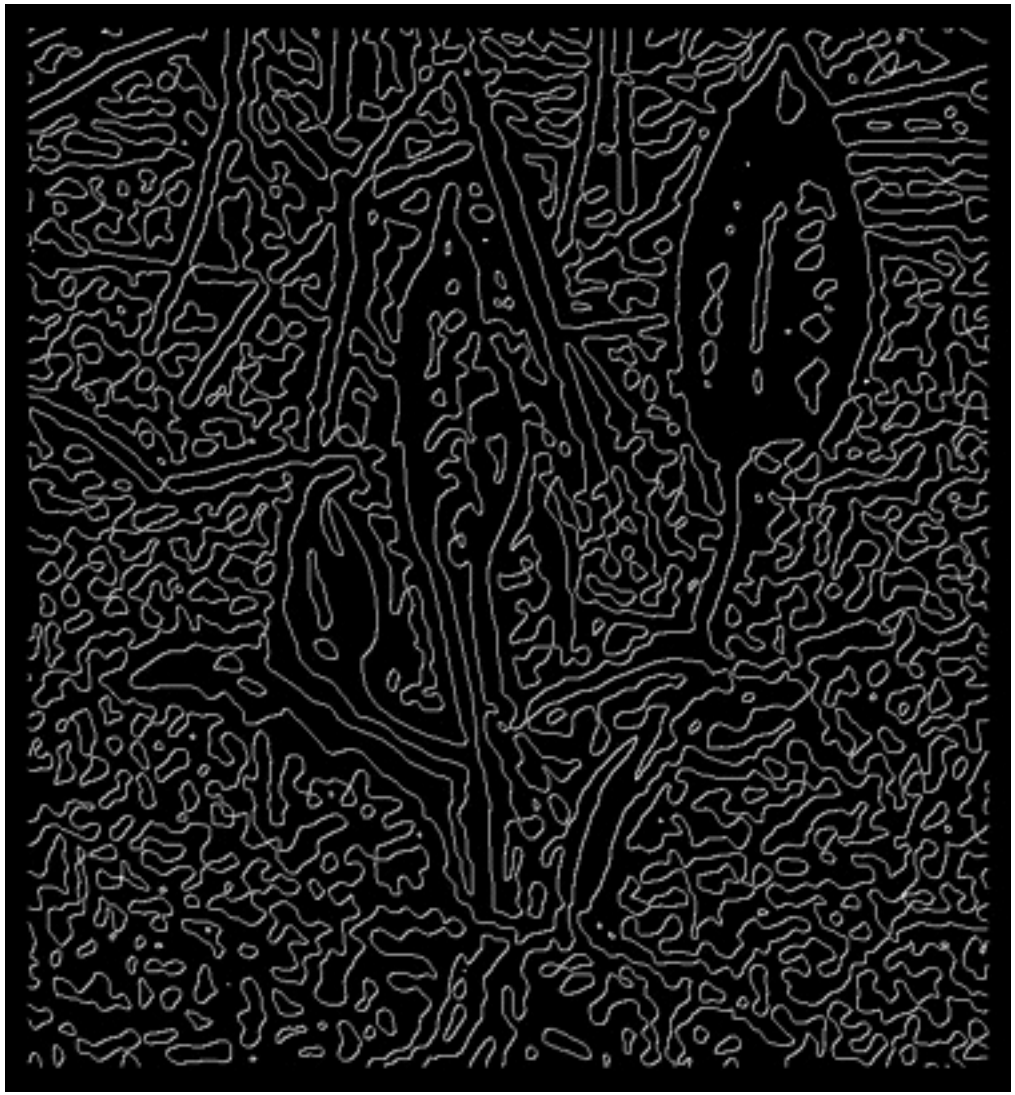
- Laplace-Operator
- Nulldurchgänge bestimmen:

  $\nabla^2(f(i,j)) \cdot \nabla^2(\text{shift}(f(i,j))) \leq 0$

  (shift: Verschiebung des Bilds um ein Pixel in jede Richtung)

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} + \frac{\partial^2 f(x,y)}{\partial x \partial y} + \frac{\partial^2 f(x,y)}{\partial y \partial x} : \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Nulldurchgänge



Kombination des Laplace-Operators mit Glättungsoperator (z.B. als LoG-Operator) reduziert die Anzahl der Nulldurchgänge

Wasserscheidentransformation

- Wasserscheide: Menge aller Orte, die die Grenzen der Entwässerung in unterschiedliche Senken sind.
- Beispiel: Wasserscheide zwischen Nordsee und Mittelmeer verläuft entlang des Kamms der Berner Alpen.
- Wasserscheide in der Segmentierung: Generiere Höhenprofil so, dass Wasserscheiden gerade die gesuchten Segmentgrenzen sind.



Wasserscheiden

- Wasserscheiden sollen an Kanten verlaufen.
- Wasserscheiden sind „Gebirgskämme“

► Wasserscheiden sind die Längen der Grauwertgradienten.



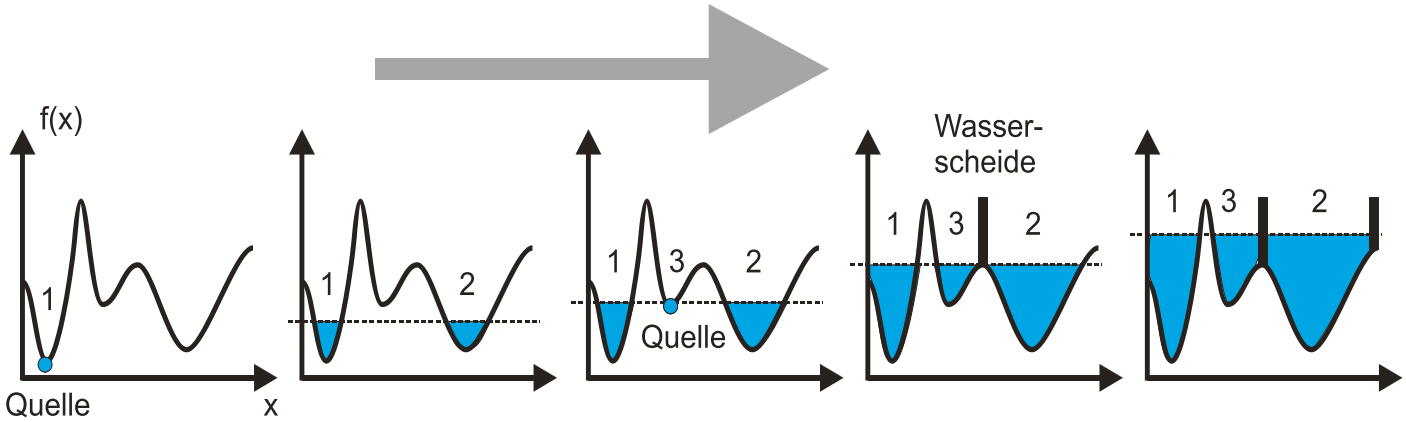
Wasserscheidentransformation

- Beregnung

  Es fällt „Regen“ auf jedes Pixel. Anhand des Gradienten wird entschieden, wohin der Regen entwässert wird.

- Flutung

  Die „Welt“ wird von den Senken her geflutet. Immer wenn Wasser aus zwei Senken zusammen fließt, entsteht eine Wasserscheide.



Flutungsalgorithmus (Skizze)

Jedes neu überflutete Pixel (m_f,n_f) ist

- **in Isolation**:

  Es nicht zu anderen überfluteten Pixeln der Höhen $h<h_{aktuell}$ benachbart.

  Isolierte Pixel sind Kerne von neuen Segmenten.

- **Erweiterung**:

  Es ist zu anderen überfluteten Pixeln der Höhen $h<h_{aktuell}$ mit gleichem Label benachbart.

  Das Pixel wird dem Segment mit diesem Label zugeordnet.

- **Wasserscheide**:

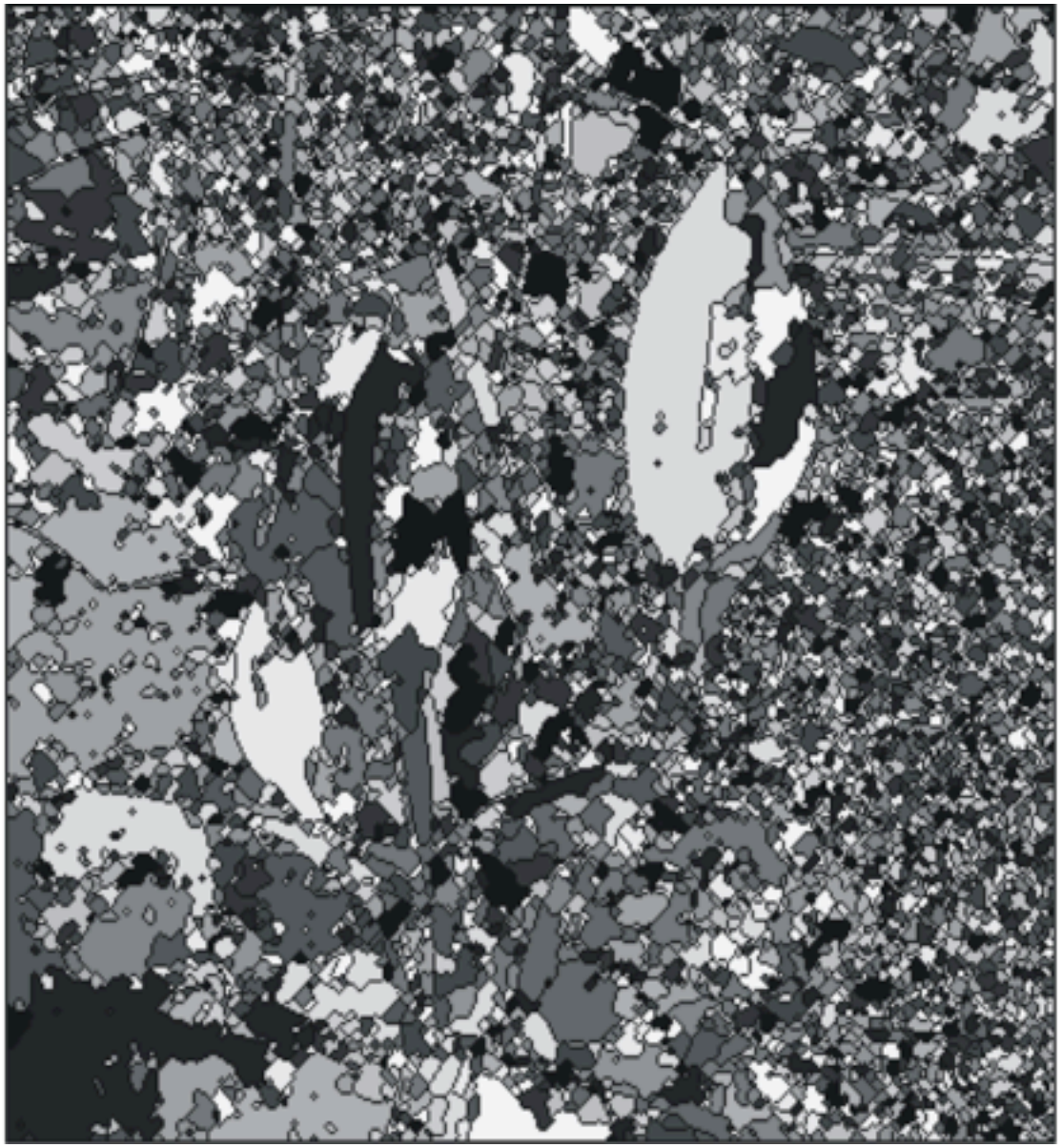
  Es ist zu überfluteten Pixeln von mindestens zwei Regionen benachbart.

  Dem Pixel wird das Label „Wasserscheide“ zugeordnet.

Resultat der WST

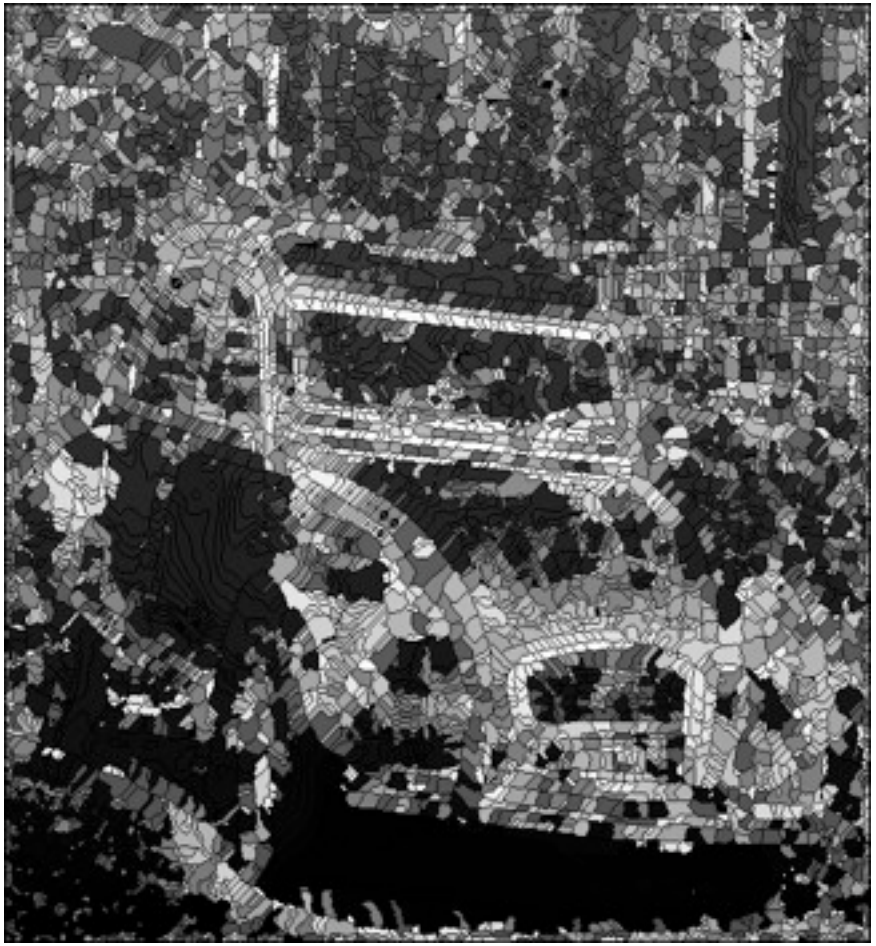






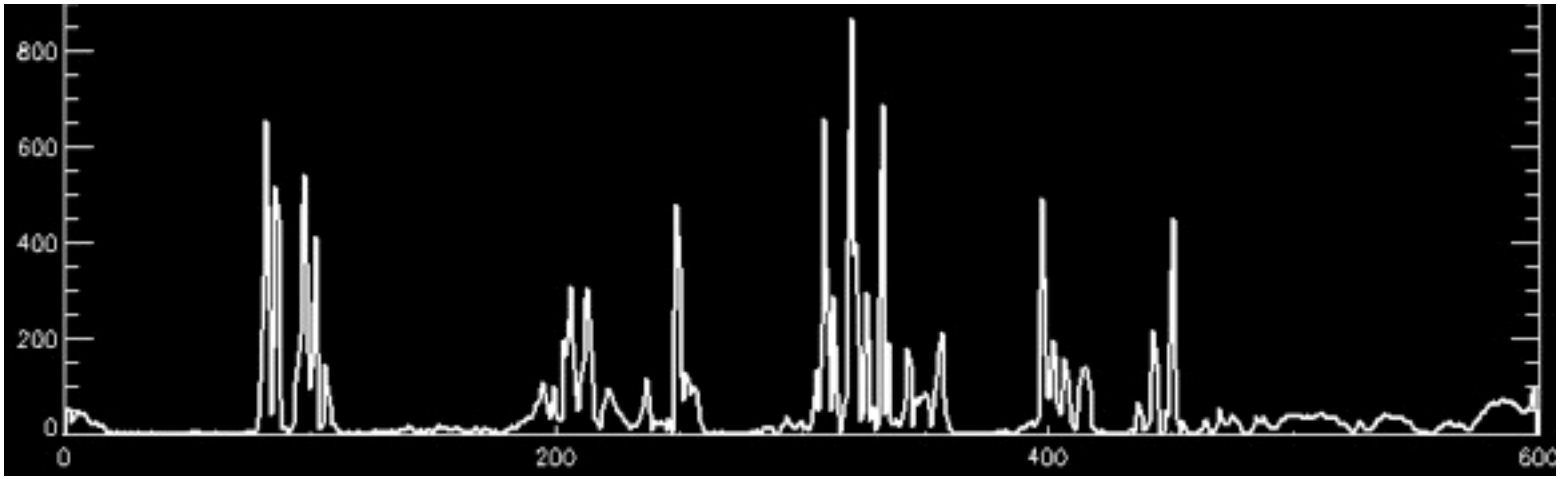
Resultat





Problem Übersegmentierung

- Für die WST ist jedes lokale Minimum eine Senke.
- Die meisten Senken werden durch Rauschen verursacht.
- Senken durch Rauschen sind weniger tief als die von Kanten.

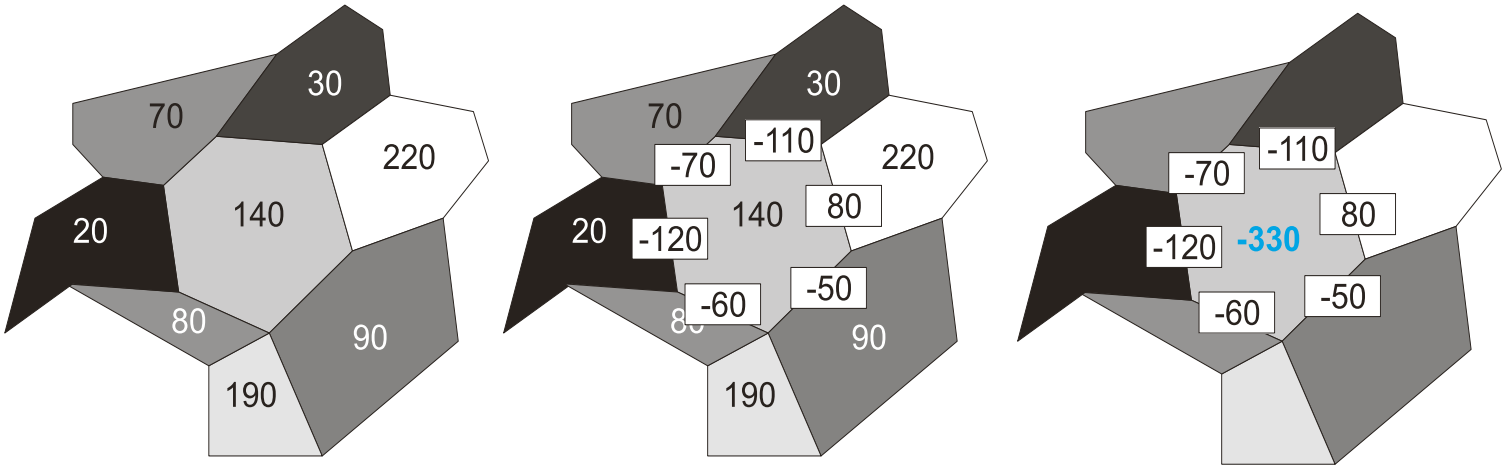


Hierarchische WST

Multiskalenstrategie:

- Wasserscheidentransformation auf dem WST-Resultat.
- Jede Region erhält ihren durchschnittlichen Grauwert als Funktionswert.
- Die erste WST wird hauptsächlich durch Rauschen verursachte Senken finden.
- „Wahre“ Senken sollten über mehrere Stufen der Hierarchie erhalten bleiben.

Gradienten für die hWST



- Zu jeder der benachbarten Regionen wird die Differenz berechnet.
- Die Länge des Gradienten ist die durchschnittliche Differenz zu allen Regionen.
- Die Richtung ergibt sich aus den (mit der Regionengröße gewichteten) Vektoren zwischen dem Schwerpunkt der Region und den Schwerpunkten aller benachb. Regionen.

…nochmal zum langsam mitmeißeln…

- Edge Linking und Canny Edge Operator
- Nulldurchgänge zur Regionensegmentierung
- Wasserscheidentransformation
  - Ausführung
  - Probleme
  - Multiskalenstrategie

Literatur

- Klaus D. Tönnies: "Grundlagen der Bildverarbeitung", ISBN 3-8273-7155-4
- http://www.pearson-studium.de/main/main.asp?page=booksites/selectchapter&isbn=3827371554&PSZielgruppe=Student

Computergraphik 1
2. Teil: Bildverarbeitung

Modellbasierte Segmentierung,
Morphologische Operationen

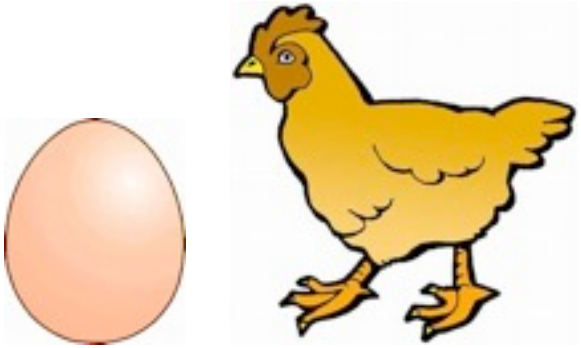
Modellbasierte Segmentierung

- Interaktive Suche
  - Region Growing
  - Kantenverfolgung
  - Markerbasierte WST
- Vollständige Suche
  - Template Matching
  - Hough Transformation

Modellbasierte Segmentierung

- Segmentierung: Generierung von Symbolen (Bedeutungsträgern) aus Pixeln.
- Modell: Erwartete Bedeutung

► Henne-Ei-Problem??



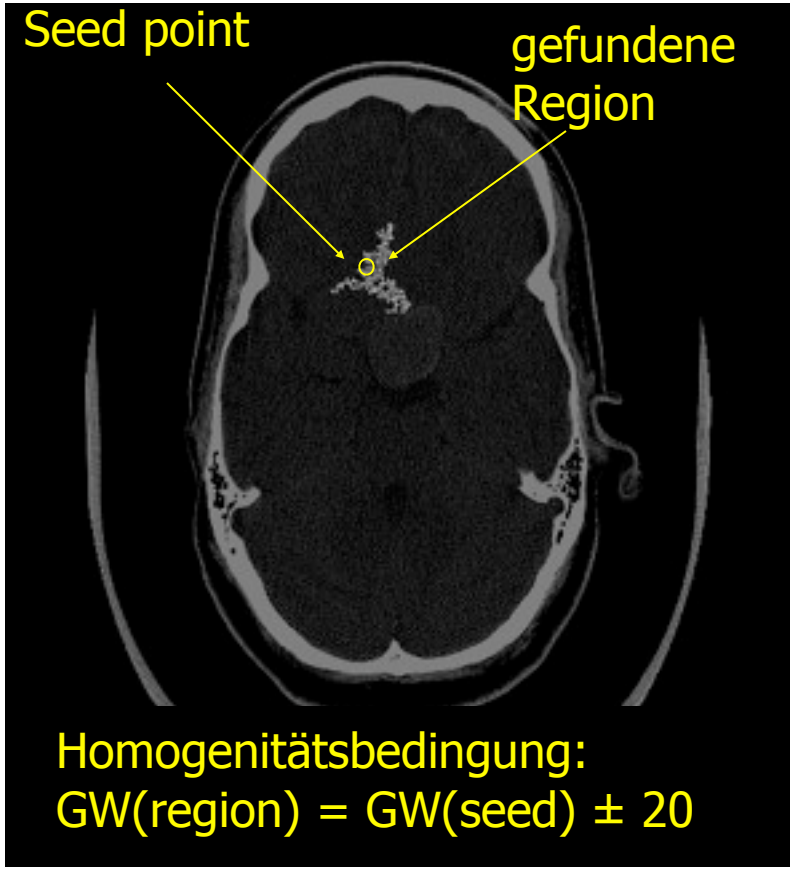
Modellbasierte Segmentierung: Mix aus Segmentierung und Analyse. Mit Vorwissen über ein bestimmtes Objekt wird nach Instanzen gesucht

- Interaktive Suche: Benutzer gibt Vorwissen interaktiv ein
- Vollständige Suche: Instanzen eines Modells mit wenigen Parametern werden gesucht
- Iterative Suche: Instanzen eines Modells mit vielen Parametern werden gesucht.

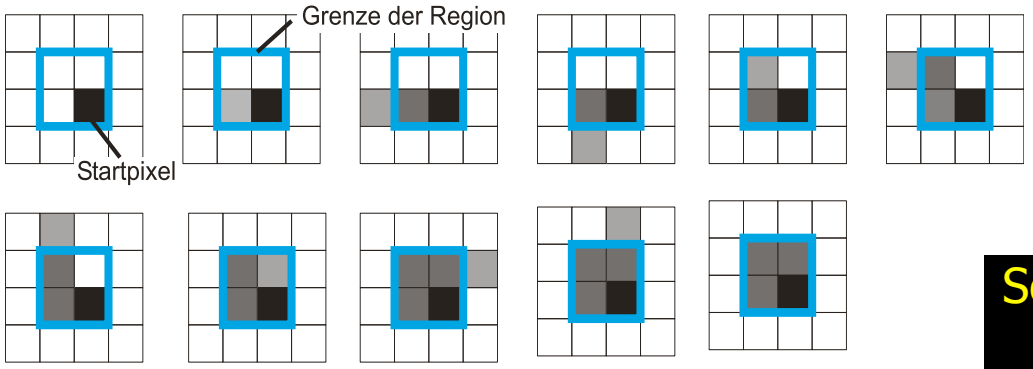
Region Growing

- = Flood Fill auf Grauwertbild für einen einzigen (vorgegebenen) Saatpunkt.

- Homogenitätsbedingungen:
  - Grauwertbereich
  - Grauwertschwankung
- Annahmen und Ziel:
  - Gebiet ist intern von größerer Homogenität als an den Grenzen.
  - Selektion eines einzigen Gebiets.
  - Homogenitätsverhältnisse an anderen Orten interessieren nicht.

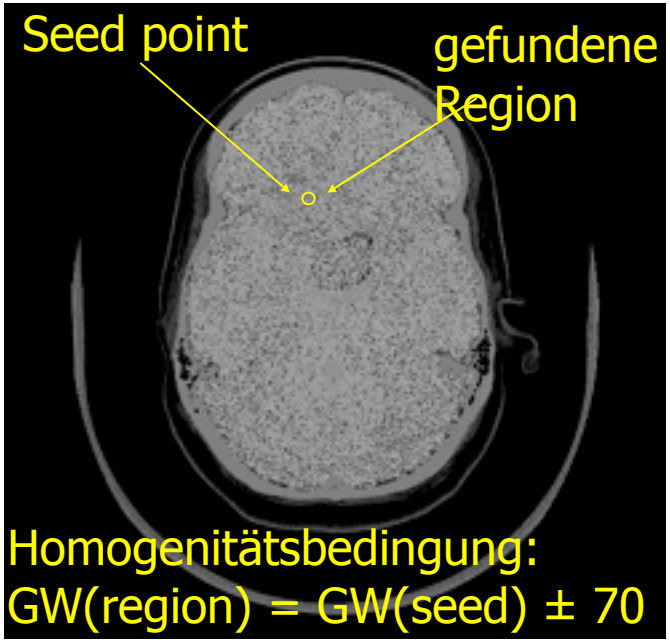


Region Growing

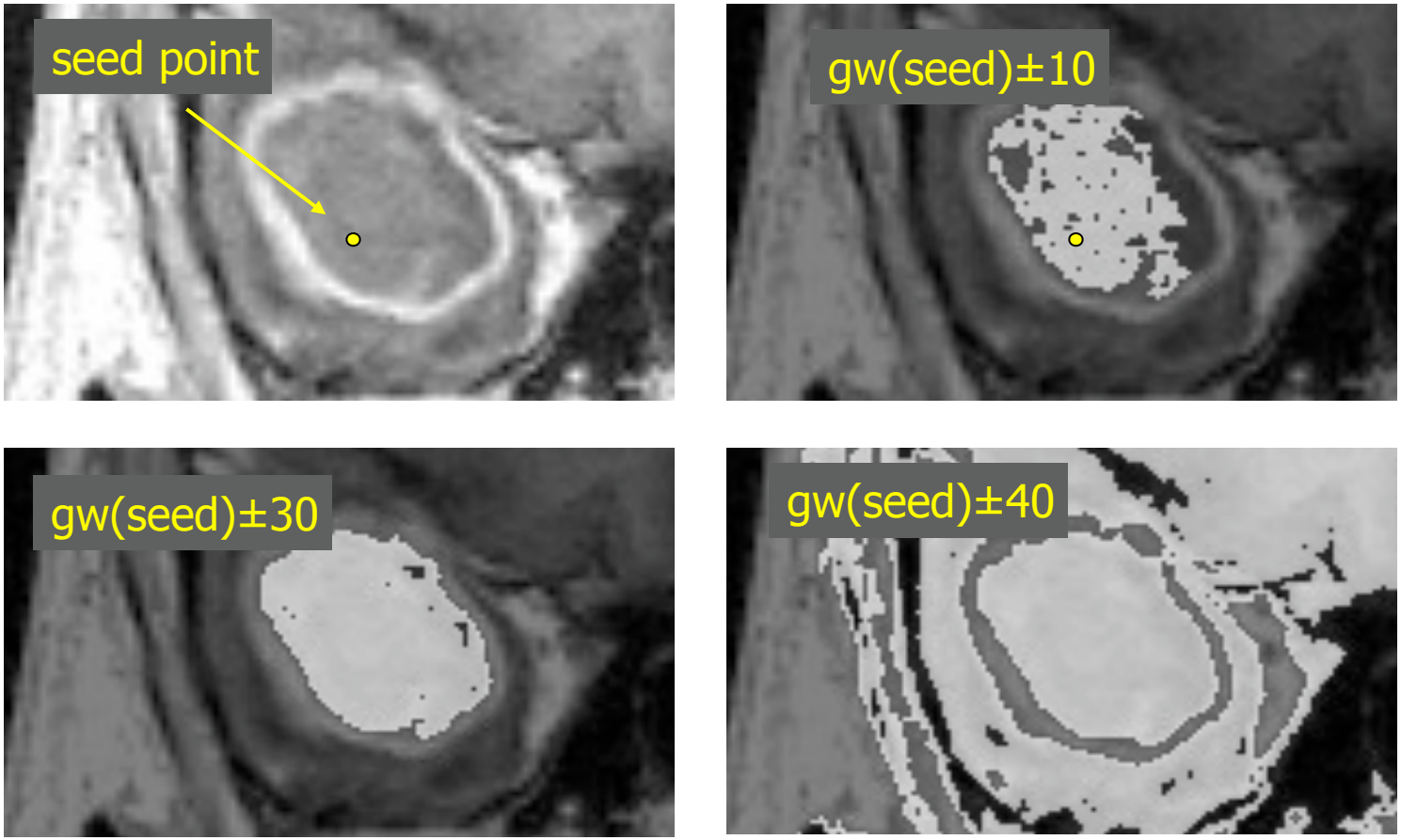


Probleme:

- „Auslaufen“ der Regionen
- zu kleine Regionen
- Rauschanfälligkeit
- Shading im Bild



Auslaufen von Regionen



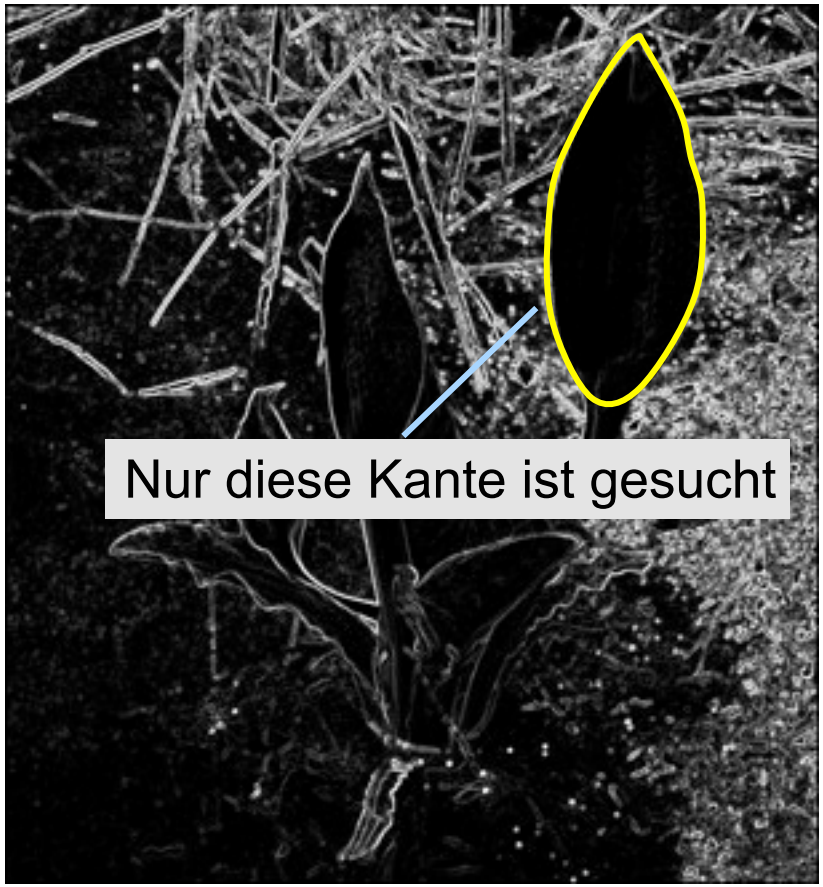
Region Growing und Rauschunterdrückung

Rauschen führt zu Anfälligkeit der Methode

➔ vorherige Rauschunterdrückung oder Nachverarbeitung



Gezielte Kantensuche



Resultat der Kantenfilter:

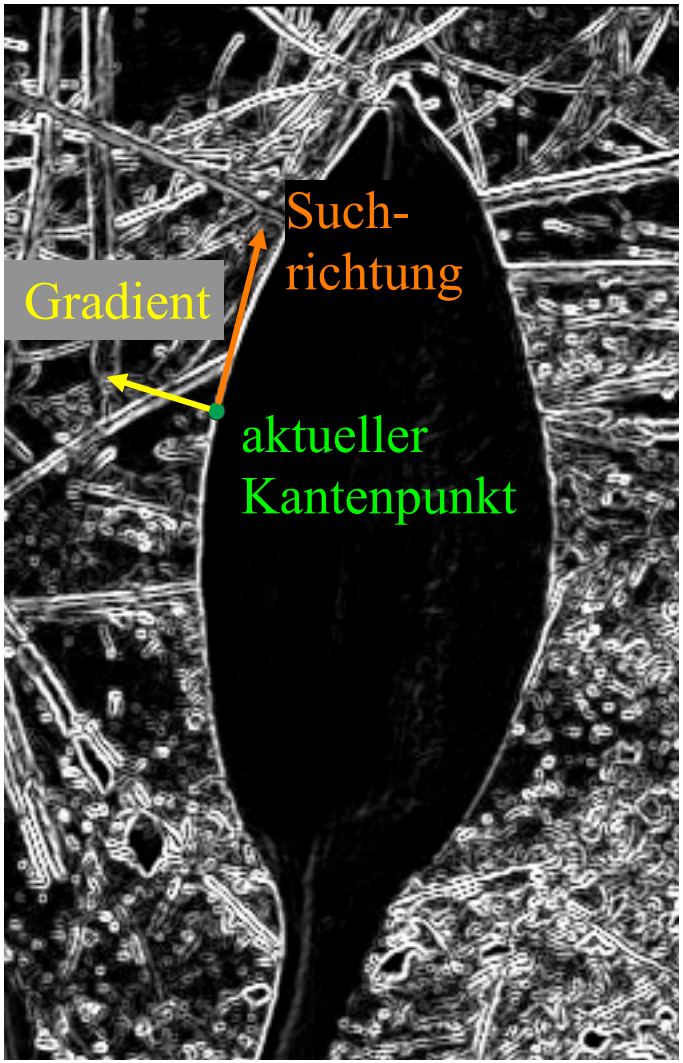
Liste von Kantenpunktkandidaten

- nicht alle Kandidaten gehören zur gesuchten Regionengrenze.
- die Regionengrenze kann Lücken aufweisen.

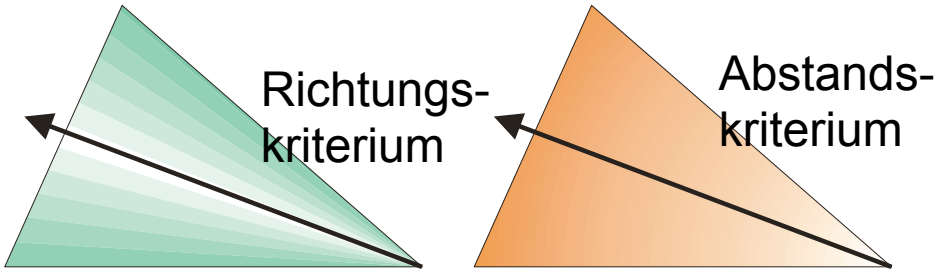
Strategien mit Nutzerinteraktion

- Kantenverfolgung
- Optimale Kantenzüge
- Hough Transformation

Kantenverfolgung



- Anfangspunkt bestimmen
- suche senkrecht zur aktuellen Gradientenrichtung
- Kantenpunkt ist gefunden, wenn
  - in hinreichend kurzer Entfernung,
  - mit hinreichend geringer Richtungsabweichung
  - ein hinreichend großer Gradient existiert.



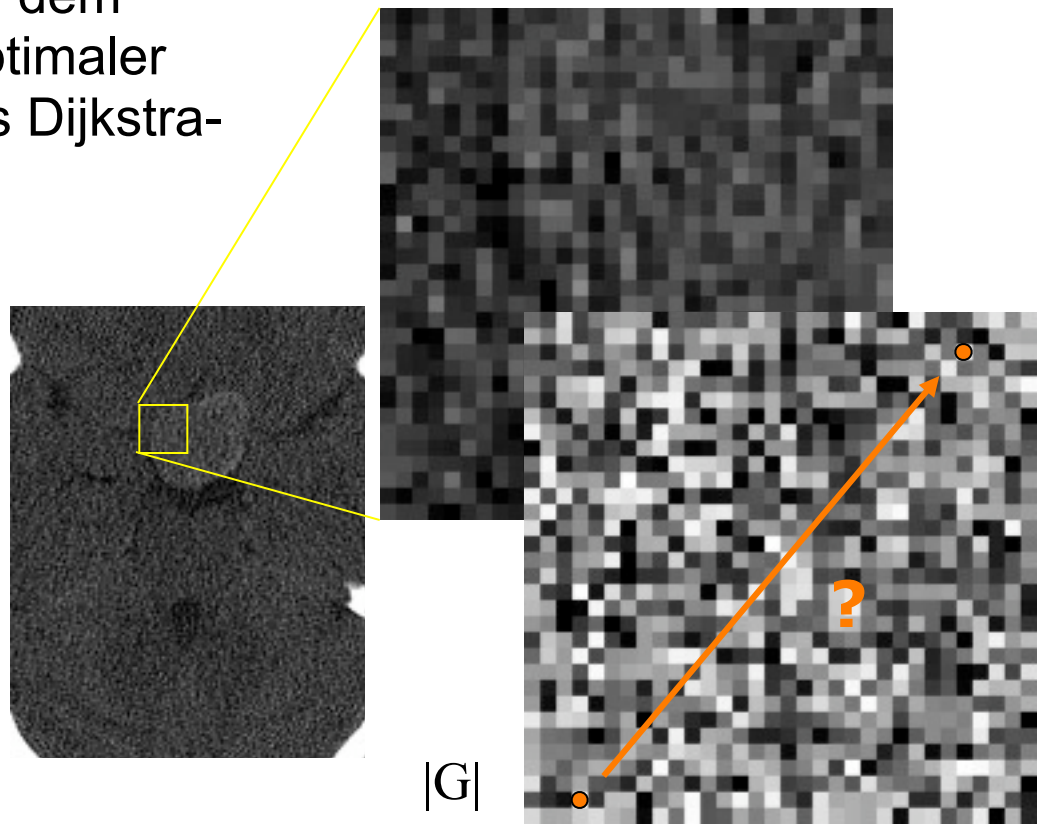
Optimale Kantenzüge (Graphensuche)

Das Resultat des Gradientenoperators wird als Graph aufgefasst, auf dem zwischen zwei Punkten ein optimaler Pfad gesucht wird (z.B. mittels Dijkstra-Algorithmus).

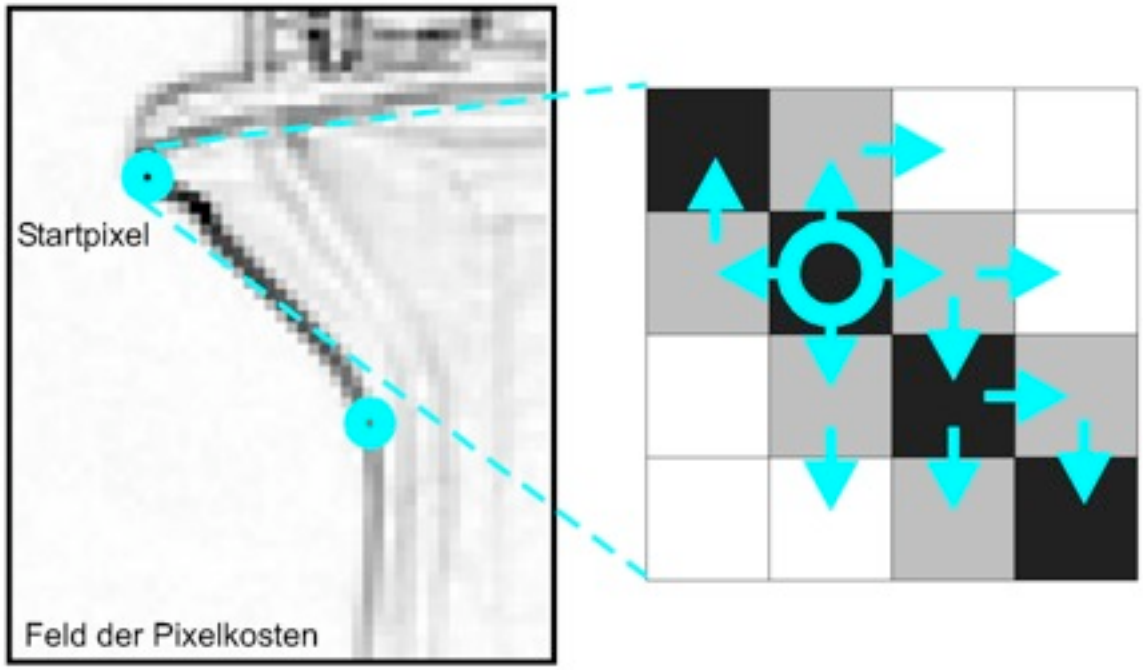
Vorteil:

Globale Aspekte des Kantenzugs können eingebracht werden. (z.B. Glattheitsbedingungen)

Nachteil:
Aufwand ist größer als bei Kantenverfolgung.



Optimale Kantenzüge



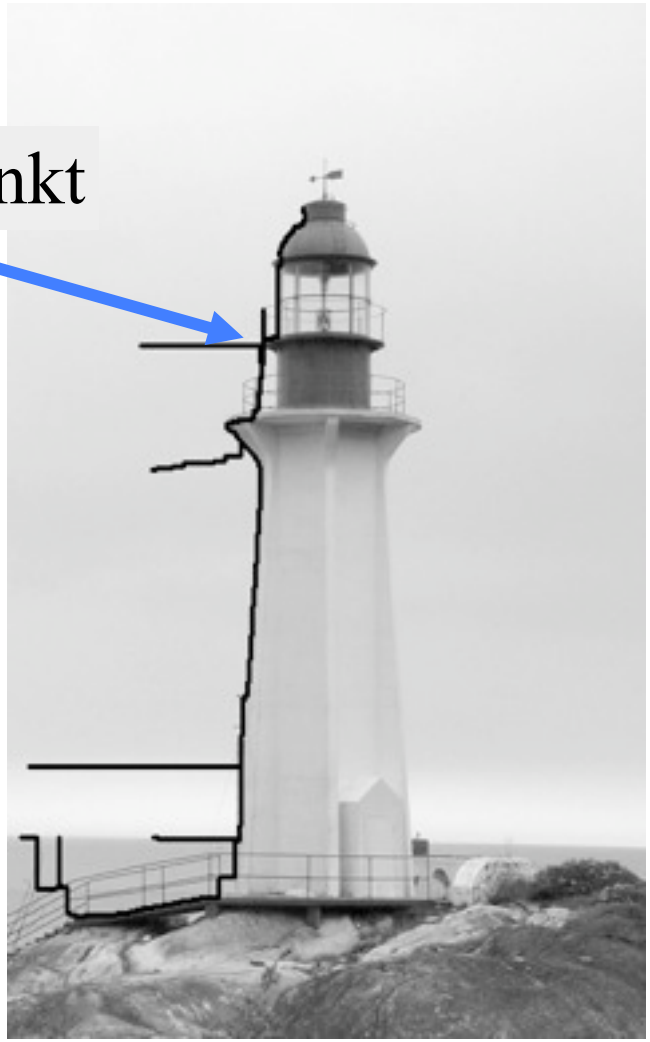
Optimalitätskriterien:

- Maximierung der (durchschnittlichen) Gradientenlänge
- Minimierung der Pfadlänge
- Minimierung der Richtungsänderungen
- Minimierung der Grauwertänderungen

Resultate



Startpunkt



Hough Transformation

Modellbasierte Suche:

- Modell einer Regionengrenze ist vorgegeben.
- Information über mögliche Kantenorte ist gegeben (z.B. durch Gradientenlänge).
- Zielsetzung: Orte finden, an denen Modell und Bild optimal übereinstimmt.

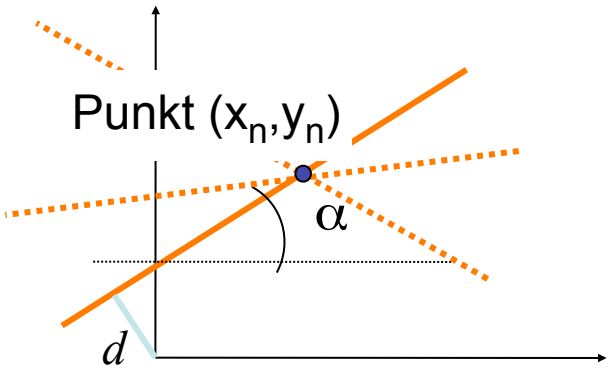
Hough-Transformation:

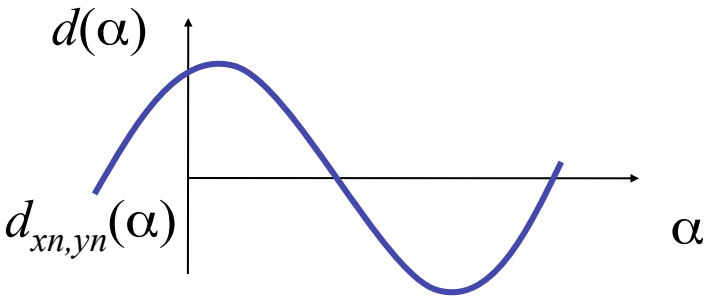
- Voting(Abstimmungs)-Mechanismus, bei dem jeder Ort in Abhängigkeit der lokalen Information für das Modell stimmt.
- entwickelt für Geraden, erweiterbar für beliebige Formen.

Hough Transformation (HT)

Suche von Geraden in einem Binärbild.

Geradenrepräsentation: $x \cos(\alpha) + y \sin(\alpha) - d = 0$





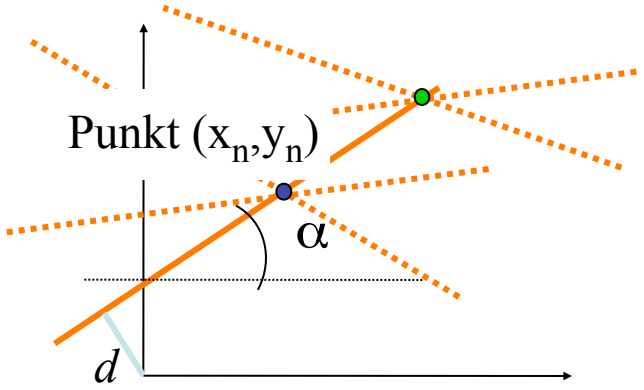
Hough-Transformation:

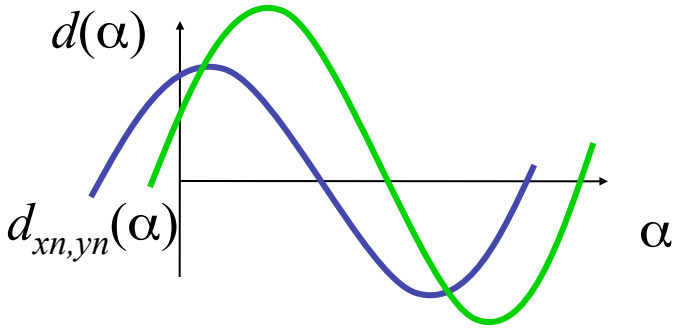
Suche alle Parameter (α,d) für Geraden, die durch einen Punkt (x_n,y_n) gehen

$$d(\alpha) = x_n \cos(\alpha) + y_n \sin(\alpha)$$

Der Raum, der durch (α,d) aufgespannt wird, heißt Hough-Raum.

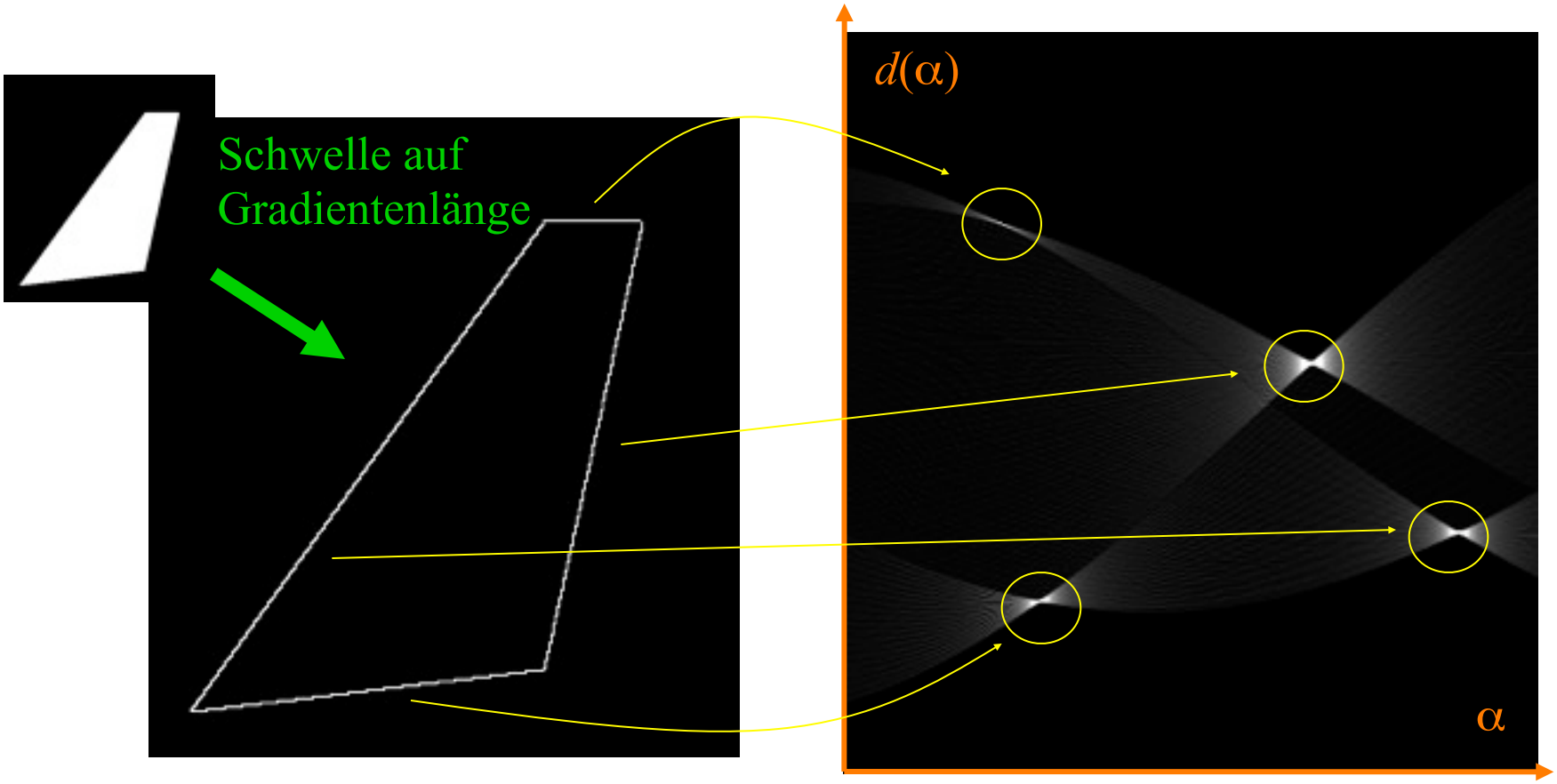
Berechnung der HT





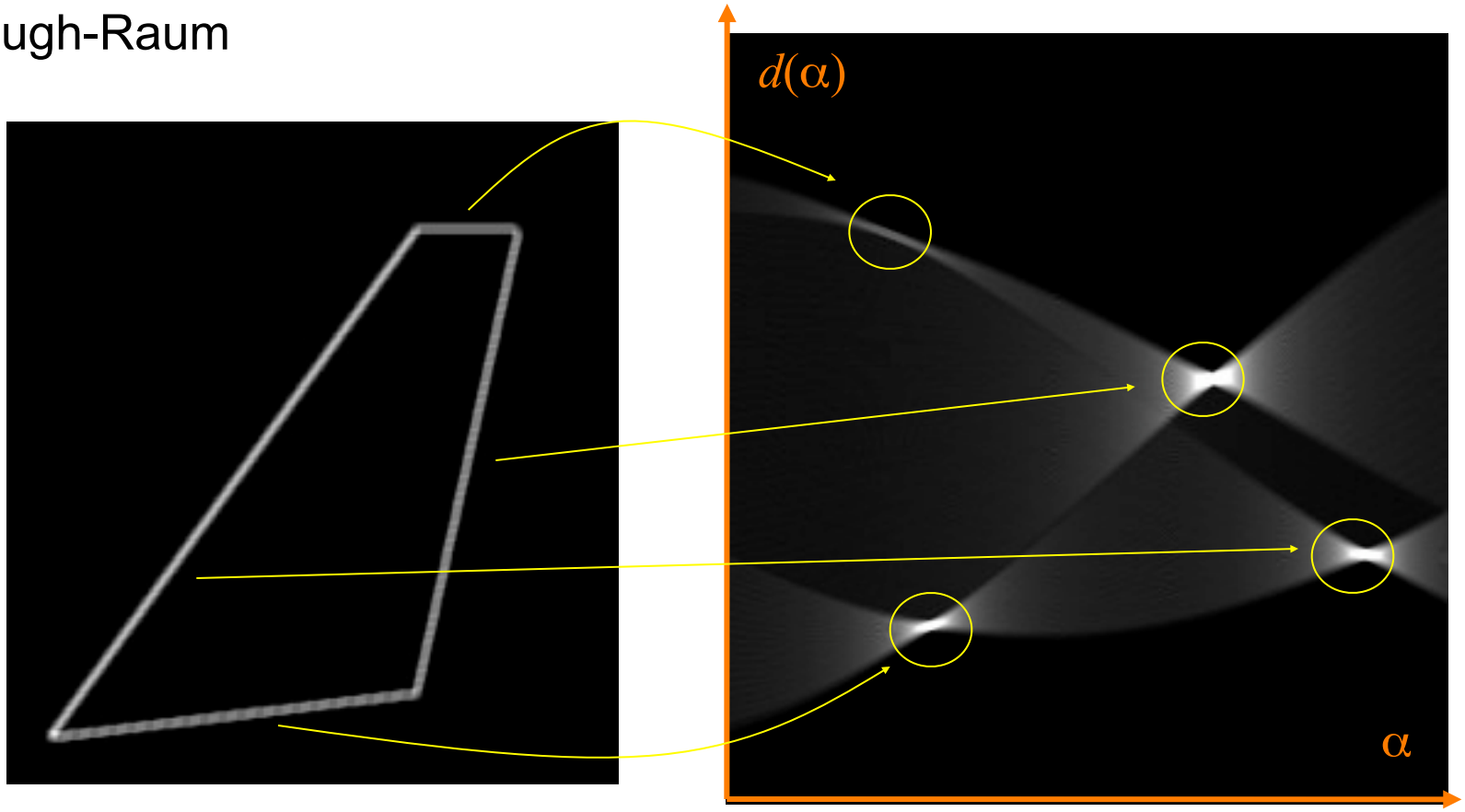
- Erzeugung eines Kantenbilds durch Schwellenwertsetzung auf Gradientenlängen.
- Diskretisierung des (α,*d*)-Raums (Zerlegung in Akkumulatoren)
- Für jeden Punkt x_n,y_n wird eine Kurve im (α,*d*)-Raum diskretisiert.
- Jeder Akkumulator wird inkrementiert, sobald eine Kurve durch in verläuft.
- Parameter von Linien im Ortsraum sind durch (α,*d*)-Kombinationen gegeben, deren Wert (Stimmenanzahl, votes) nach Ausführung der Transformation am höchsten sind.

Hough Transformation



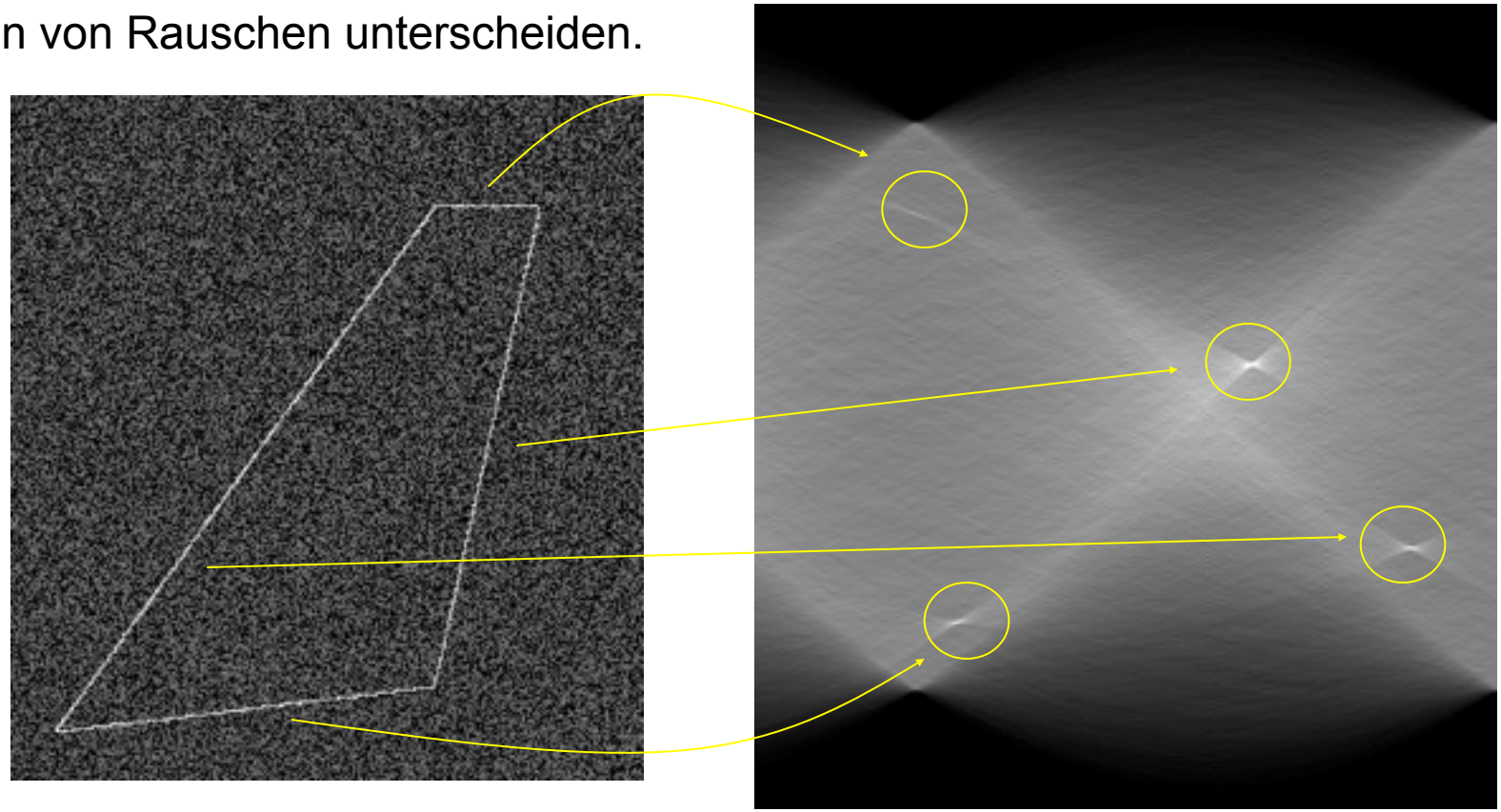
Hough Transformation

Breite Kanten führen zu flacheren Maxima im Hough-Raum



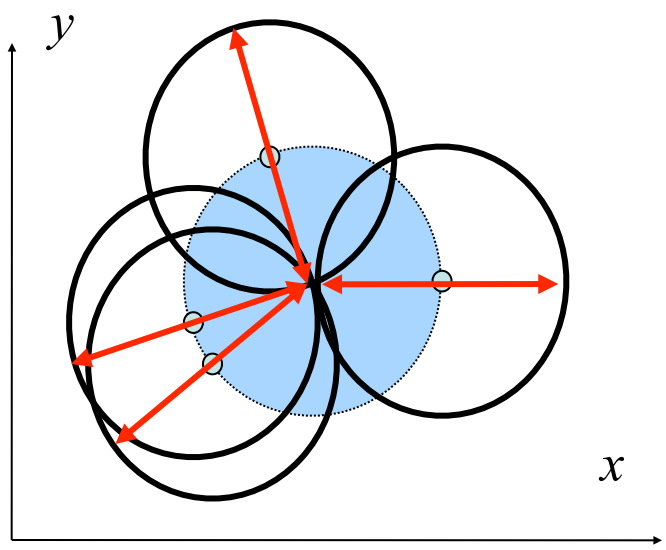
Hough Transformation in nicht-binären Bildern

Bei Gewichtung der Stimmanzahl durch die Kantenstärke lassen sich Kanten von Rauschen unterscheiden.



Hough Transformation für Kreise

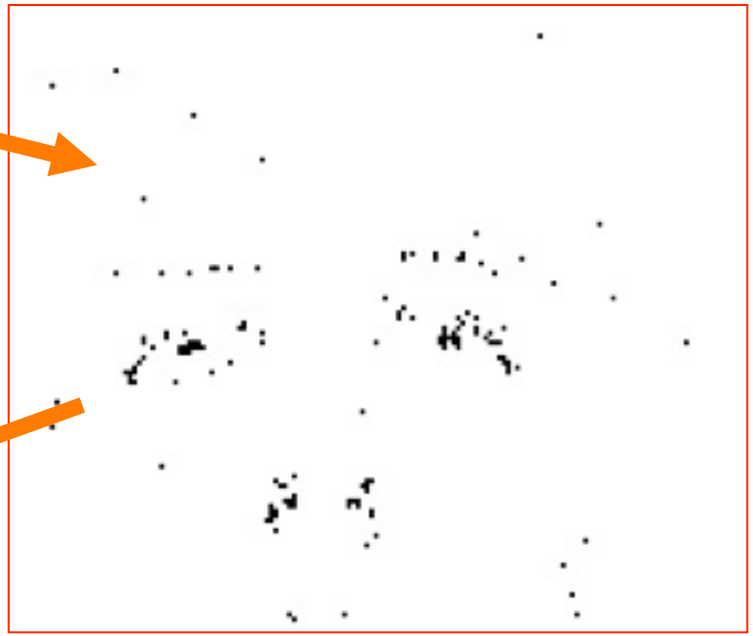
- Kreisgleichung für Kreis mit Mittelpunkt (x_c, y_c) und Radius r: $(x-x_c)^2+(y-y_c)^2-r^2=0$.
- Falls der Radius bekannt ist, ist nur der Verschiebevektor (x_c, y_c) gesucht
  - Hough-Raum = Ortsraum
  - Um jeden Kantenpunkt wird ein Kreis mit Radius r diskretisiert.
- Beschleunigung: Akkumulator wird nur in Distanz r in und entgegen der Gradientenrichtung inkrementiert.



Hough Transformation für Kreise



Akkumulatorzellen mit mehr als 33% der maximalen Stimmenanzahl.





selektierte Kreise mit r=4.5mm

Zusammenfassung bis hier hin:

- Rolle des Modellwissens bei der Segmentierung
- Methoden mit interaktivem Einbringen von Modellwissen
  - Welche Art von Modellwissen
  - Wo und warum erfolgreich
- Methoden, die eine vollständige Suche verwenden
  - Welche Art von Modellen
  - Voraussetzungen

Morphologische Operationen

- Erosion und Dilatation
- Opening und Closing
- Ränder und Distanzen, Morphing
- Hit-or-Miss-Operator
- Skelettierung

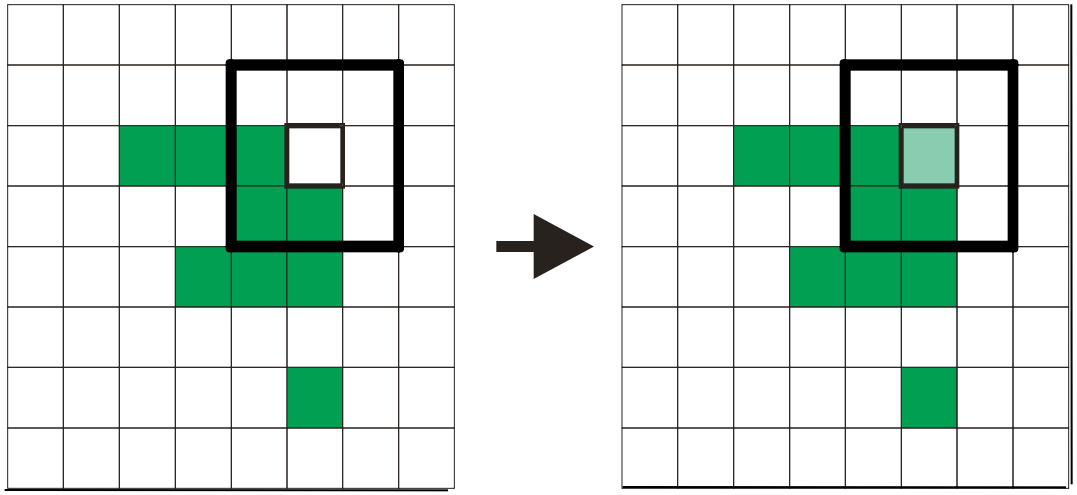
Morphologische Operationen

- Morphologisch: die äußere Gestalt betreffend
- morphologische Operationen:
  - Operationen auf der Gestalt von Objekten
  - ➔ setzt die Extraktion einer Gestalt voraus
  - also: in erster Linie Operation auf Segmenten (d.h., auf Binärbildern)
- Ziel von morphologischen Operationen:
  - Veränderung der Gestalt, um Störungen nach einer Segmentierung zu beseitigen
  - Berechnung von Formmerkmalen
  - Suche nach bestimmten Formen (also: Analyse)

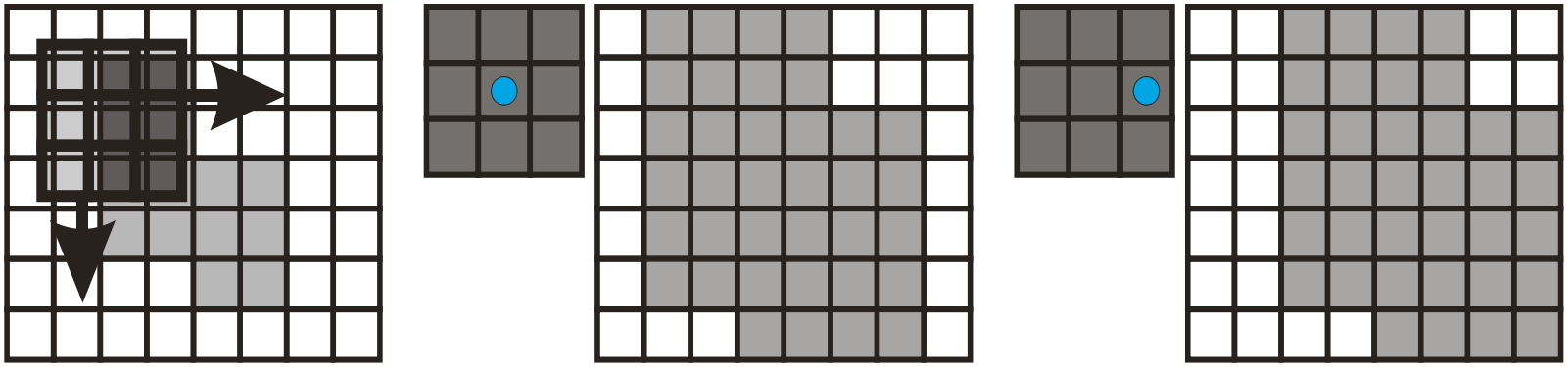
Dilatation

Dilatation (Ausdehnung): $G \oplus S$ mit Strukturelement S

$$g(m,n) = \bigvee_{(m_k,n_k) \in s} b(m + m_k, n + n_k)$$



Dilatation

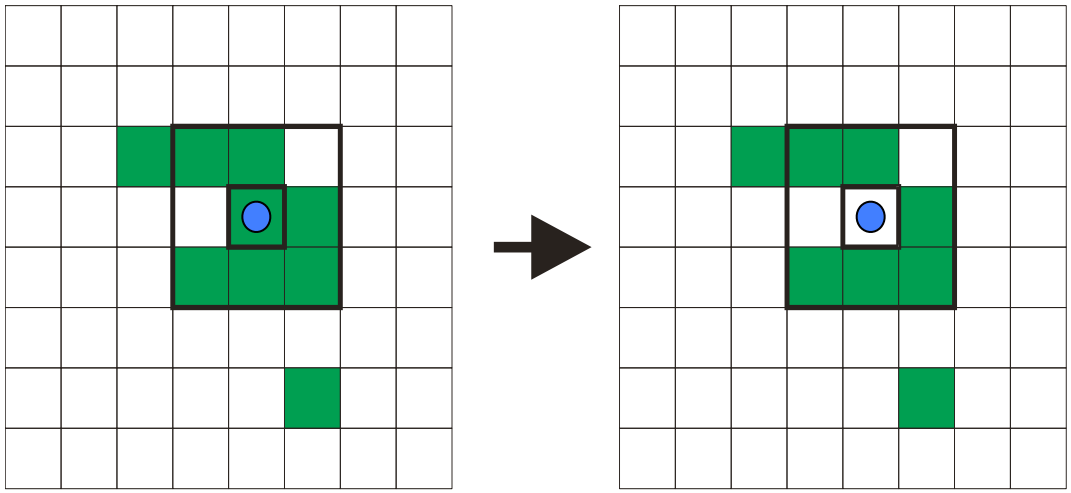


Dilatation wird (wie jede morphologische Operation) für einen **Ankerpunkt** ausgeführt.

Dilatation: - verbindet Strukturen
- füllt Löcher
- vergrößert

Erosion

$$g(m,n) = \bigwedge_{(m_k,n_k)\in s} b(m+m_k, n+n_k)$$

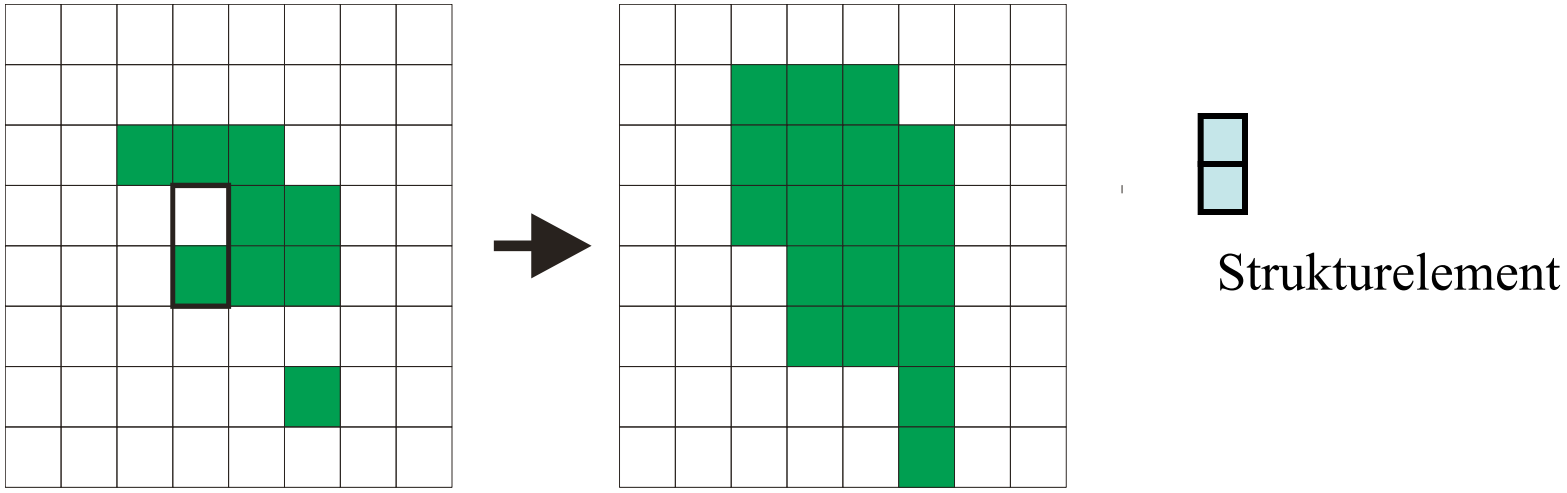


Erosion: $G \ominus S$ mit Strukturelement S

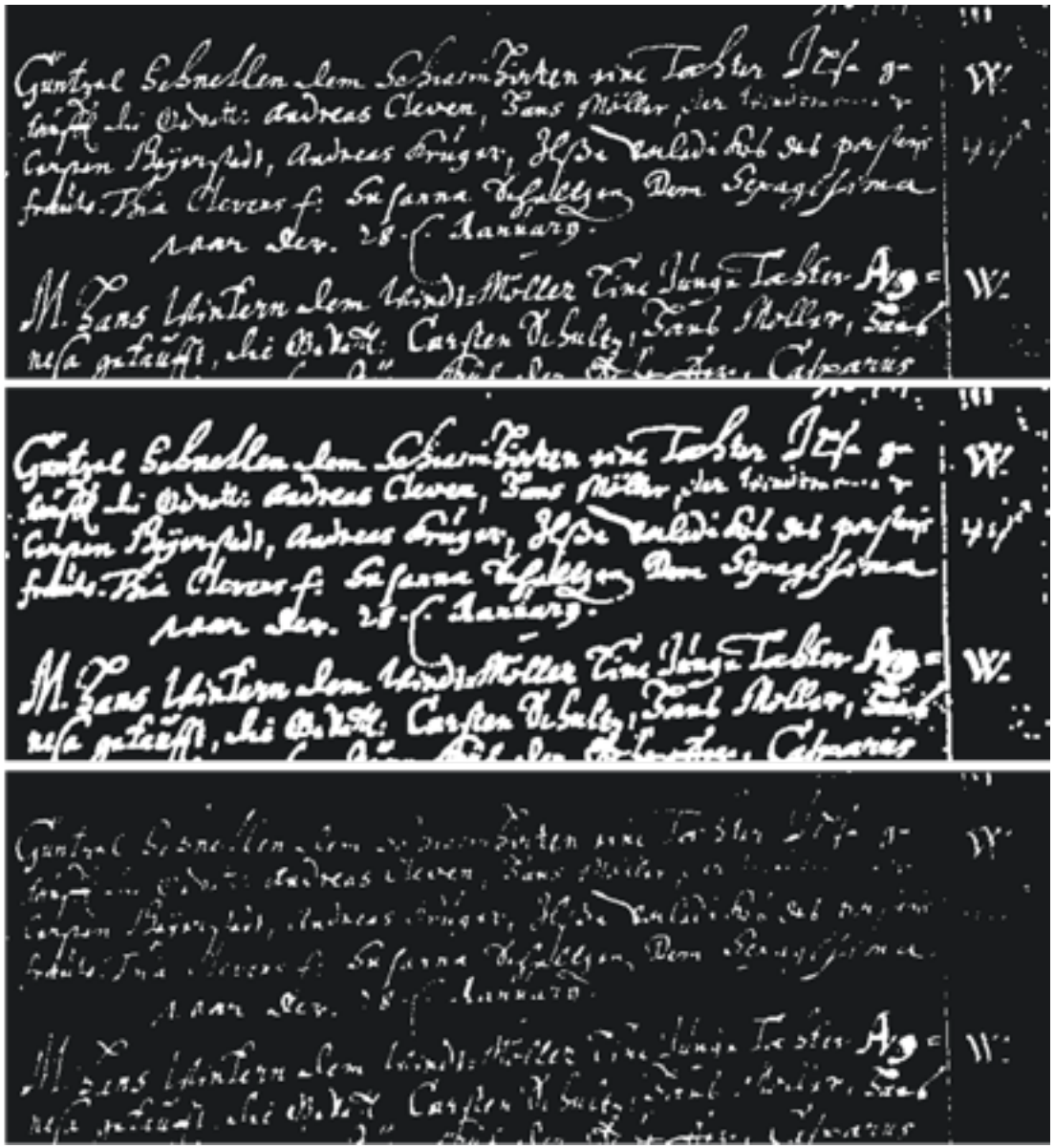
Erosion:
- löst Strukturen auf
- entfernt Details
- verkleinert

Strukturelemente

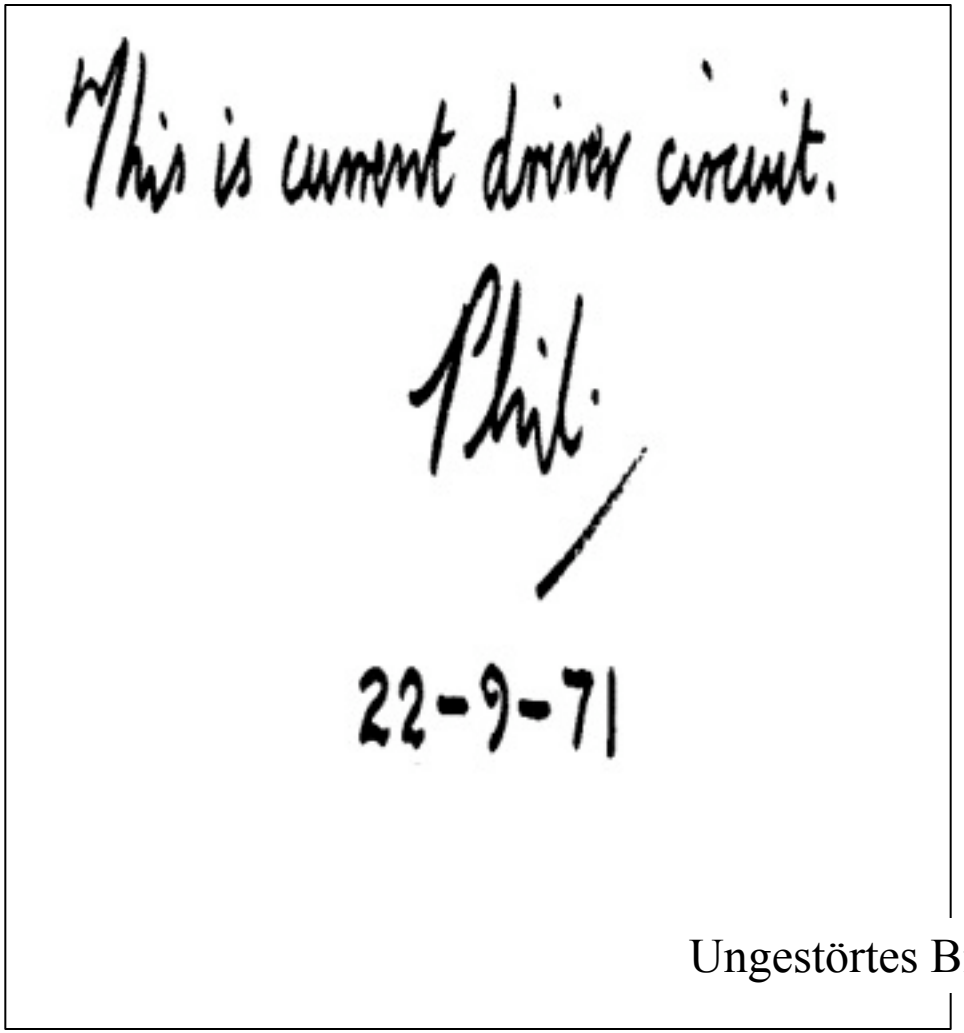
- Ein Strukturelement einer morphologischen Operation entspricht dem Faltungskern bei einer Konvolution.
- Mit einem gezielt geformten Strukturelement können genau definierte Formveränderungen erzeugt werden.



Beispiel

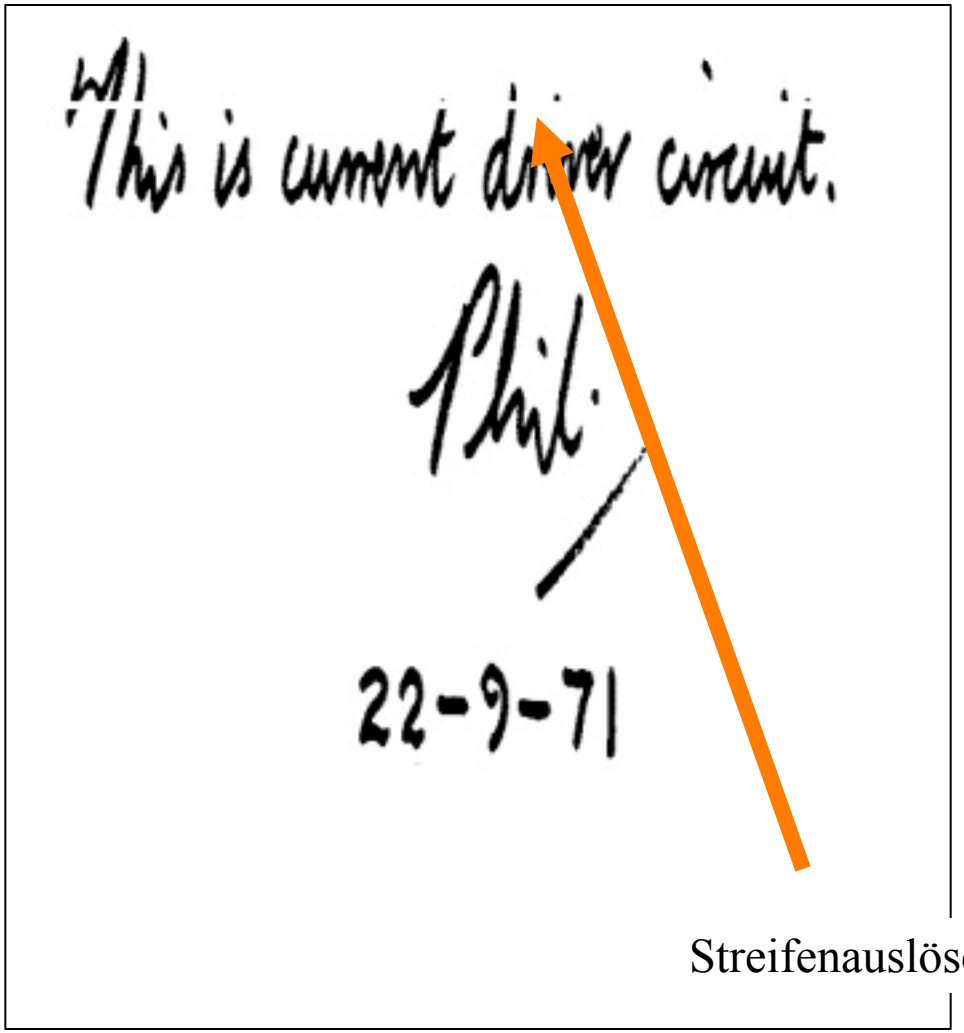


Gezielter Einsatz

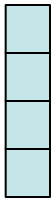


Ungestörtes Binärbild

Gezielter Einsatz

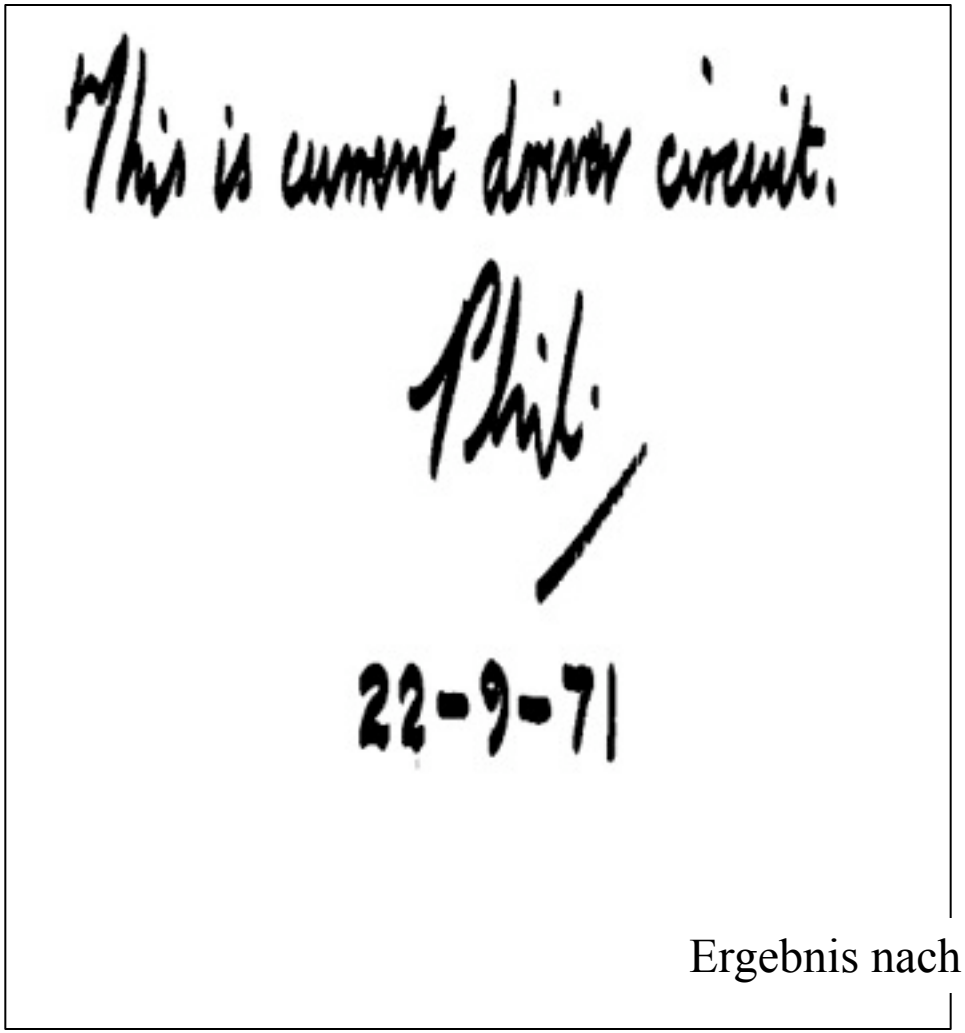


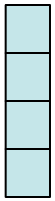
Streifenauslöschung



Strukturelement zum Schließen des Streifens

Gezielter Einsatz

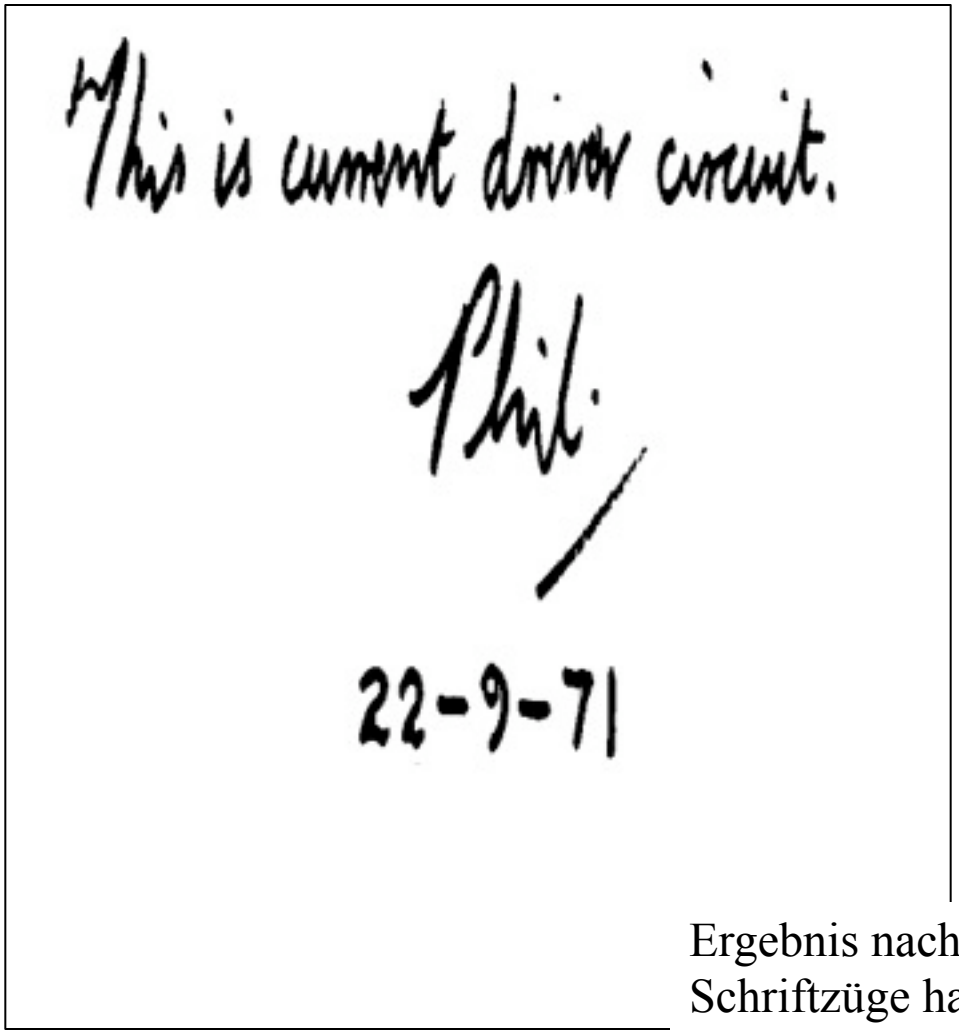


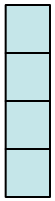


Strukturelement zum Schließen des Streifens

Ergebnis nach Dilatation: Streifen ist geschlossen

Gezielter Einsatz





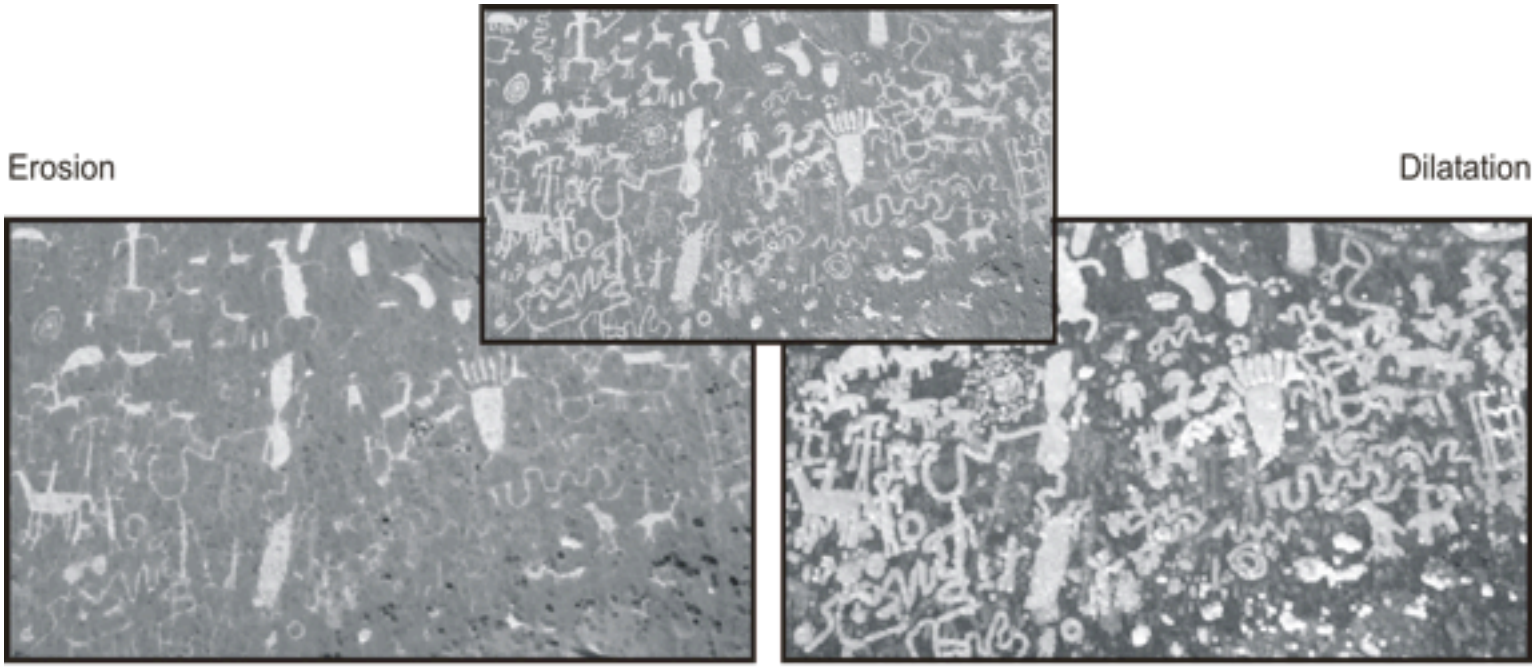
Strukturelement zur Erosion des zu breiten Schriftzugs

Ergebnis nach nachfolgender Erosion:
Schriftzüge haben ihre Ursprungsstärke

Morphologische Operationen auf Grauwertbildern

$$g(m,n) = \max_{(m_k,n_k)\in s}\left(b(m+m_k, n+n_k)\right)$$

$$g(m,n) = \min_{(m_k,n_k)\in s}\left(b(m+m_k, n+n_k)\right)$$



Opening and closing

Opening (Öffnen): Kombination von Erosion gefolgt von einer Dilation mit dem am Ankerpunkt gespiegelten Strukturelement S‘

$$G \circ S = (G \ominus S) \oplus S'$$

Ziel: Erosion - Entfernung aller (Teil-)strukturen, die kleiner als das Strukturelement sind

Dilatation - Wiederherstellung der ursprünglichen Größe des Objekts mit Ausnahme der vollständig entfernten Teilstrukturen

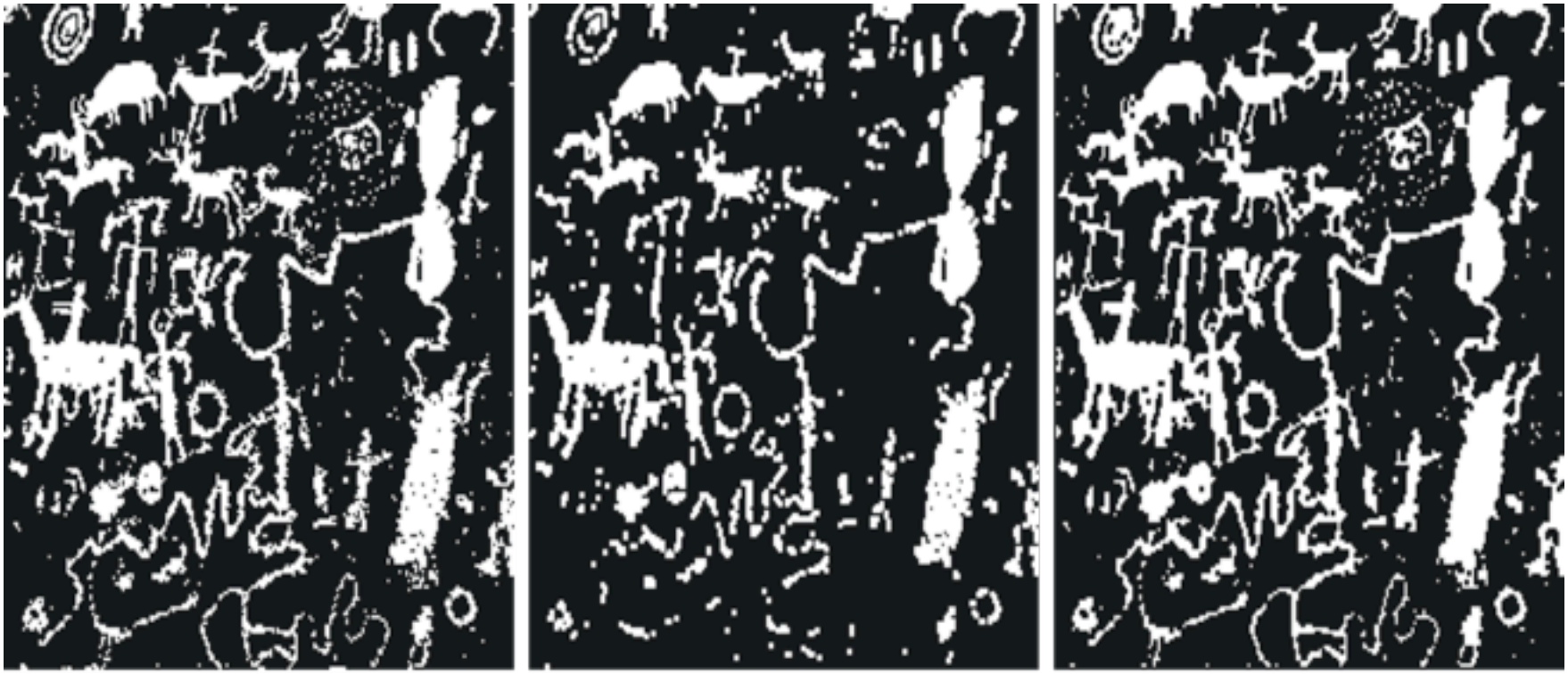
Closing (Schließen): Kombination von Dilatation gefolgt von einer Erosion mit einem am Ankerpunkt gespiegelten Strukturelement S‘

$$G \bullet S = (G \oplus S) \ominus S'$$

Ziel: Dilatation - Schließen von kleinen Löchern (kleiner als das Strukturelement)

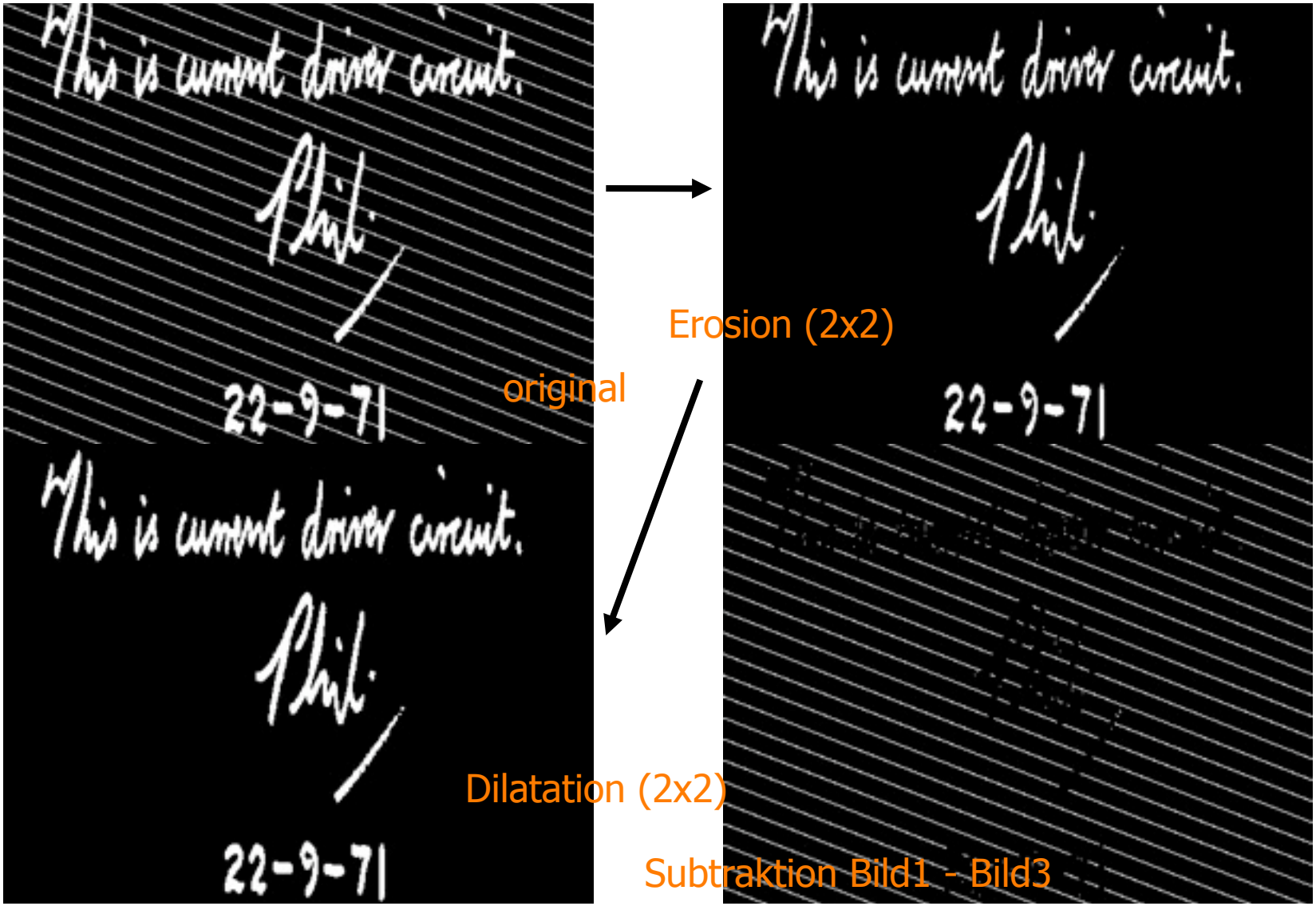
Erosion - Wiederherstellung der ursprünglichen Größe des Objekts

Beispiele Opening and closing



Original opening closing

Entfernung von Linien

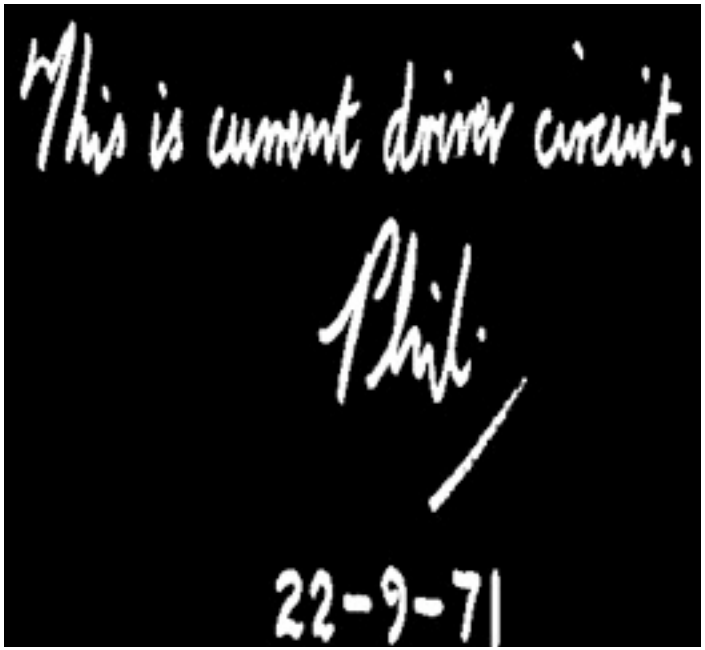


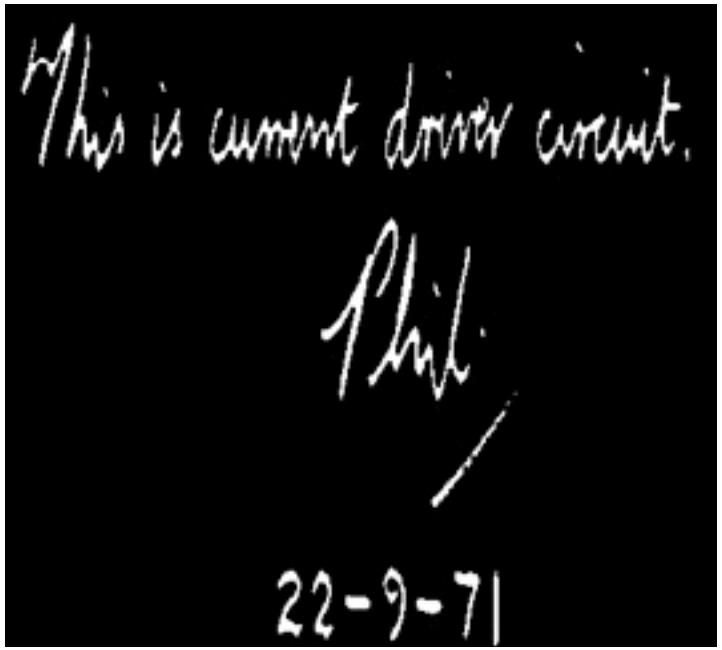
Extraktion von Rändern

$$S_{b4} = \begin{matrix} 0\ 1\ 0 \\ 1\ 1\ 1 \\ 0\ 1\ 0 \end{matrix} \qquad S_{b8} = \begin{matrix} 1\ 1\ 1 \\ 1\ 1\ 1 \\ 1\ 1\ 1 \end{matrix}$$

Erosion mit S_{b4} bzw. S_{b8} entfernt alle Objektpixel, in deren 4- bzw. 8-Nachbarschaft sich Hintergrundpixel befinden.

Der Rand kann nun durch Differenzbildung zwischen Ursprungsbild und erodiertem Bild erzeugt werden: $\partial G = G \setminus (G \ominus S_b)$





Extraktion von Rändern

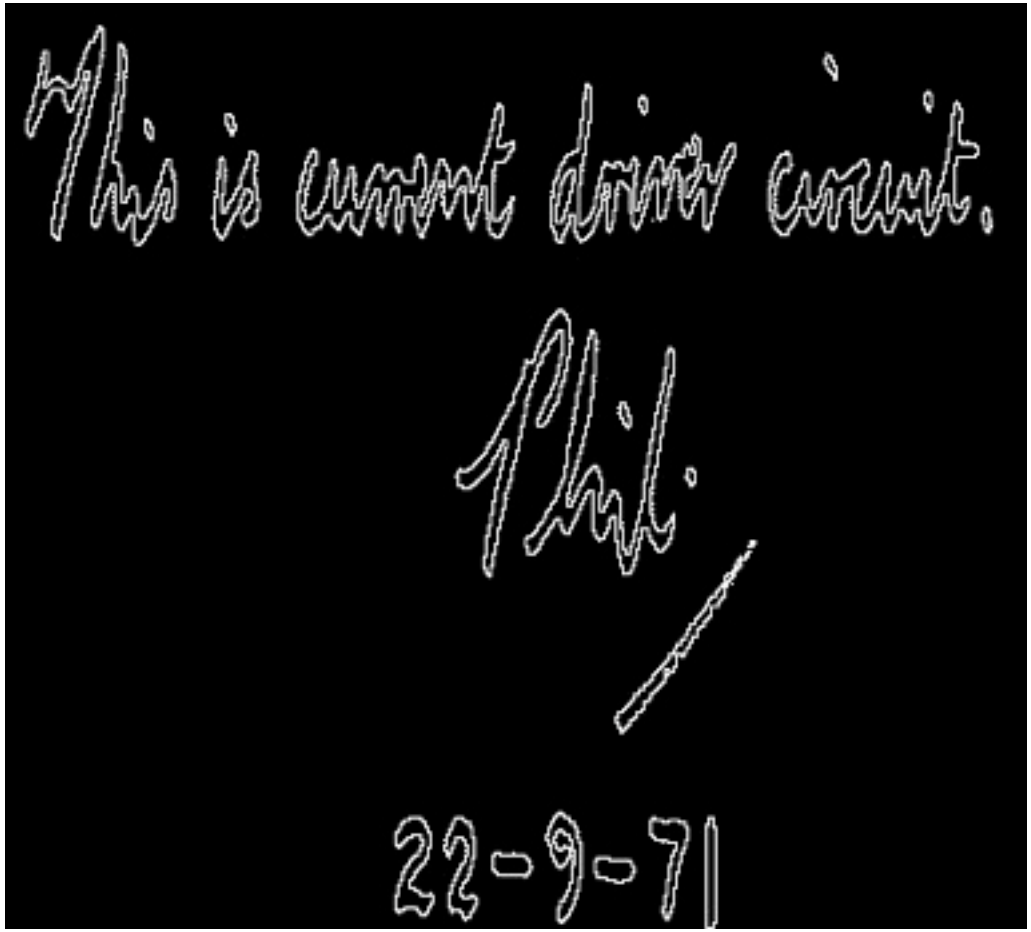
$$\partial G = G \setminus (G \ominus M_b)$$

$$= G \cap \overline{(G \ominus M_b)}$$

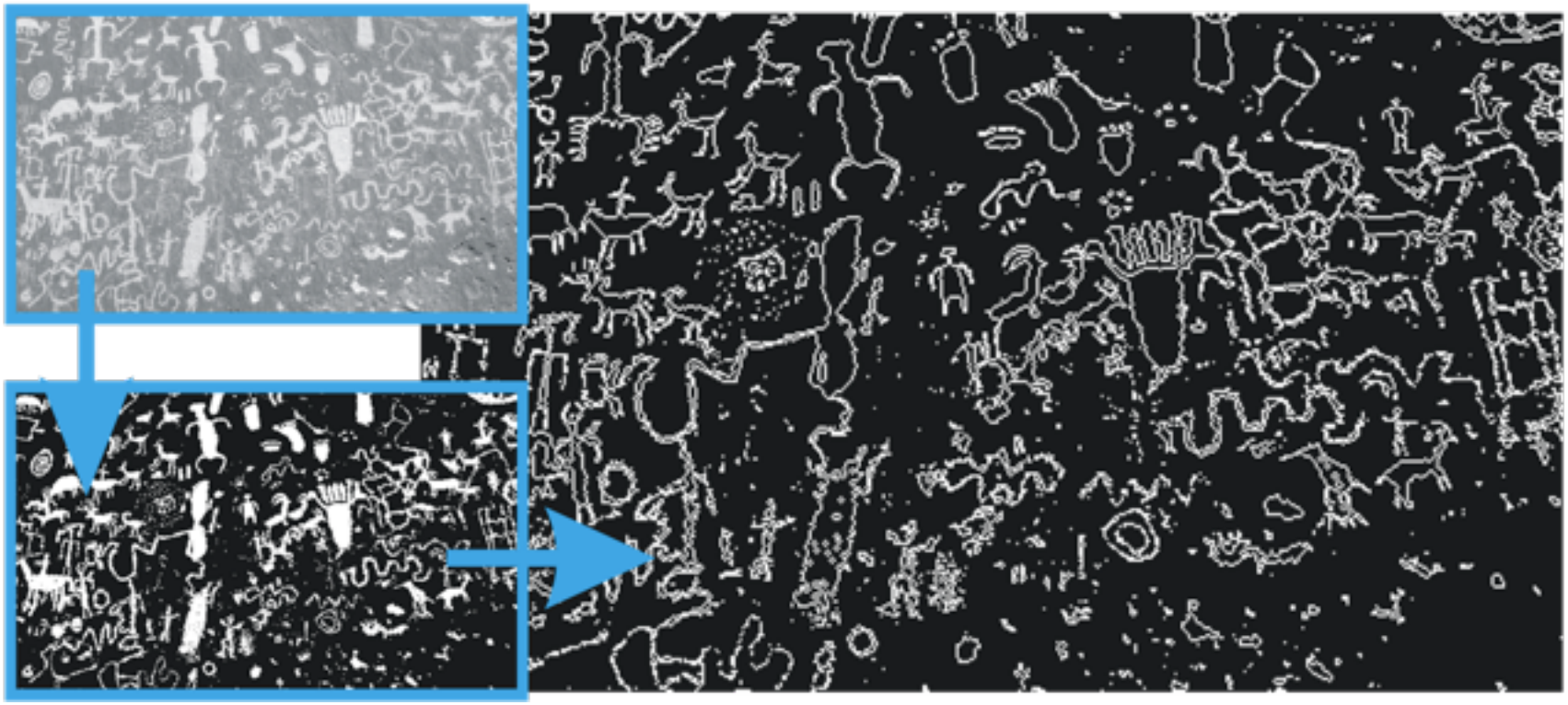
$$= G \cap (\overline{G \oplus M_b})$$

Hintergrundrand:

$$\partial G_B = (G \oplus M_b) \setminus G$$



Beispiel



Distanztransformation

Resultat der Randoperation $\partial G_0 = G \setminus (G \ominus S_b)$:

Menge aller Pixel, die den Abstand 0 zum Rand haben.

Falls die gleiche Operation auf dem um den Rand verminderten Bild nochmals angewendet wird: $\partial G_1 = (G \ominus S_b) \setminus (G \ominus S_b \ominus S_b)$

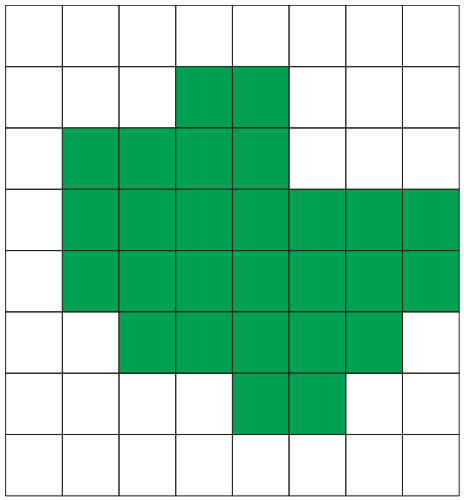
Menge aller Pixel, die den Abstand 1 zum Rand haben.

Fortgesetzte Extraktion von immer weiter vom Rand entfernten Linien und Multiplikation der jeweiligen Resultate mit der aktuellen Entfernung überführt das Binärbild in ein Distanzbild D:

$$D = \bigcup_{n=1,\infty} [\, (G \ominus S_b^{n-1}) \setminus (G \ominus S_b^{n}) \cdot n \,] ,$$

wobei die Operation $\cdot$ die punktweise Multiplikation der n-ten Randkurve mit der Zahl n (dem aktuellen Abstand) darstellt.

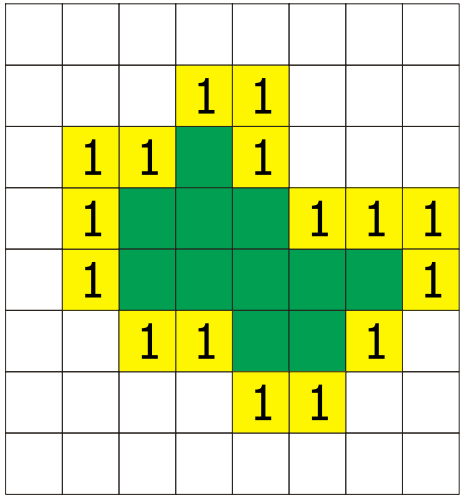
Beispiel



Originalbild

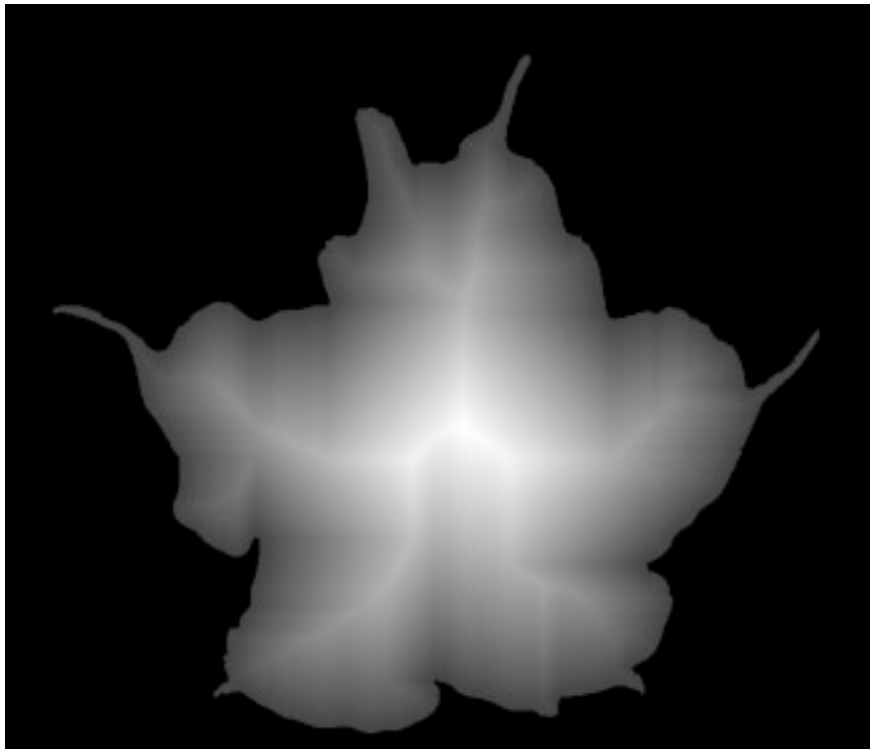
Objektinneres (nach fortgesetzter Erosion)

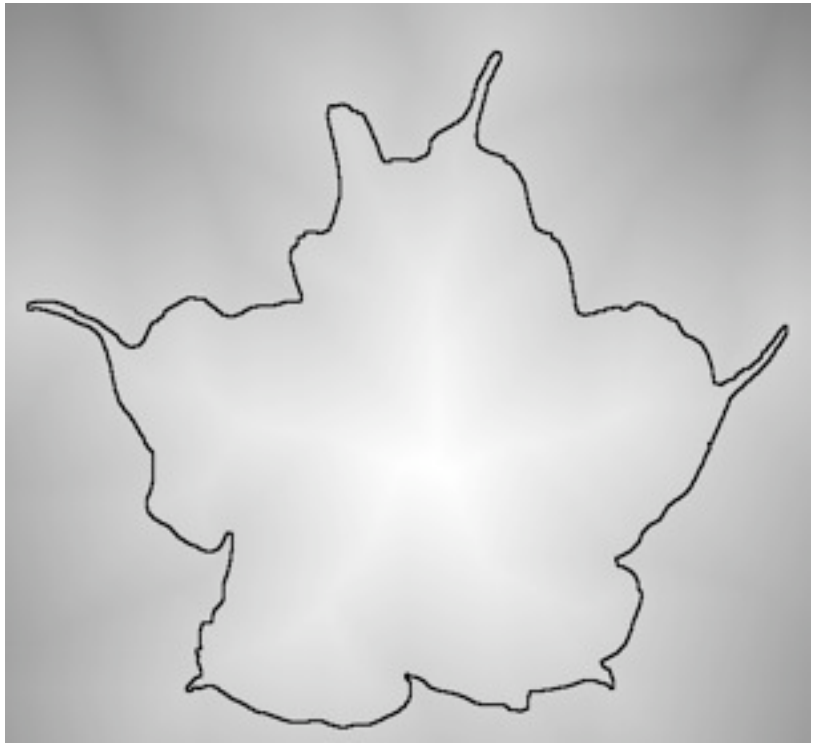
Randpixel nach der n-ten Erosion einschließlich Distanz



Beispiel







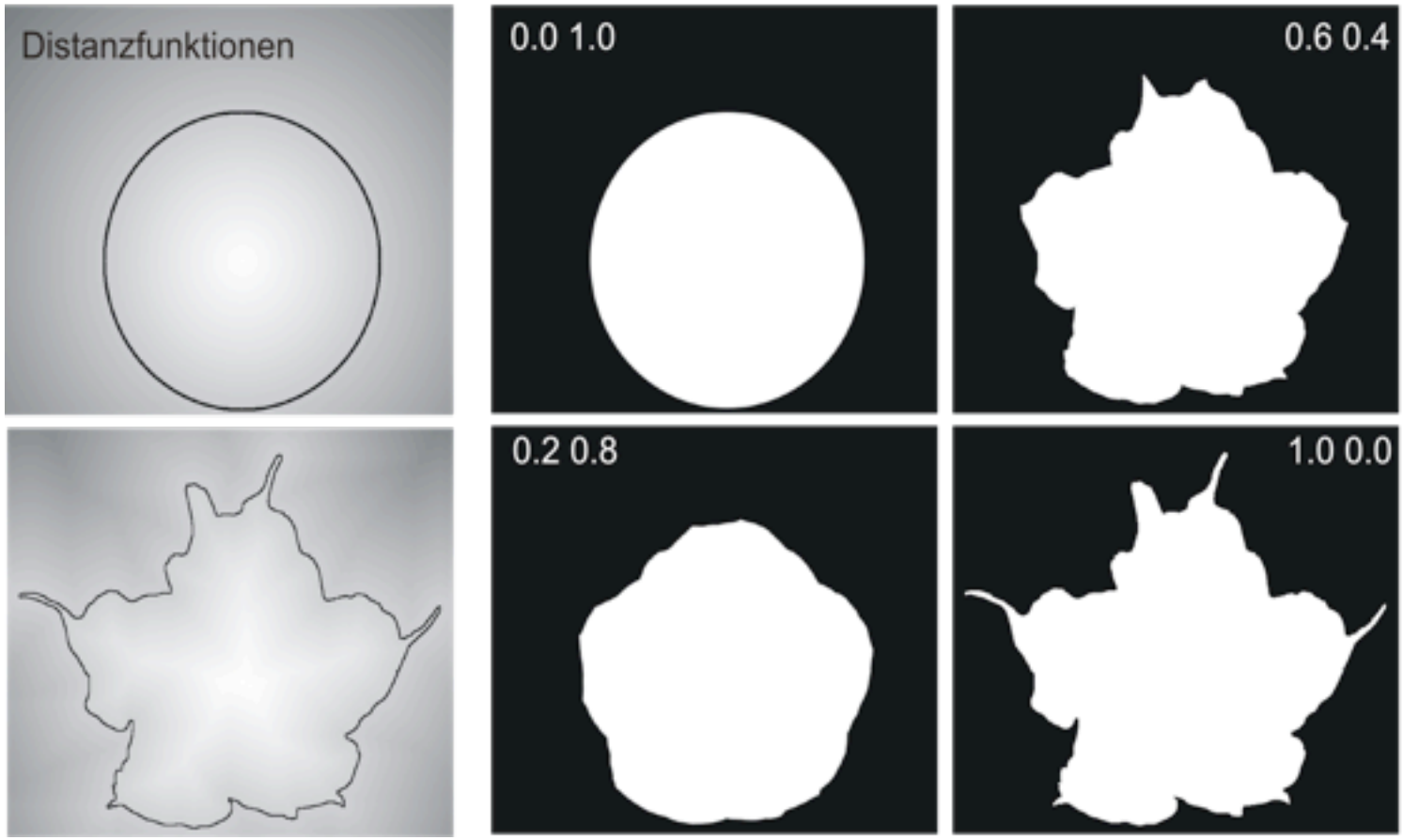
Morphing

- Vorzeichenbehaftete Distanztransformation auf Binärbildern b_A und b_B durchführen.
- Für i=0,N-1 Distanzbilder linear aus den Distanzbildern A_A und A_B interpolieren

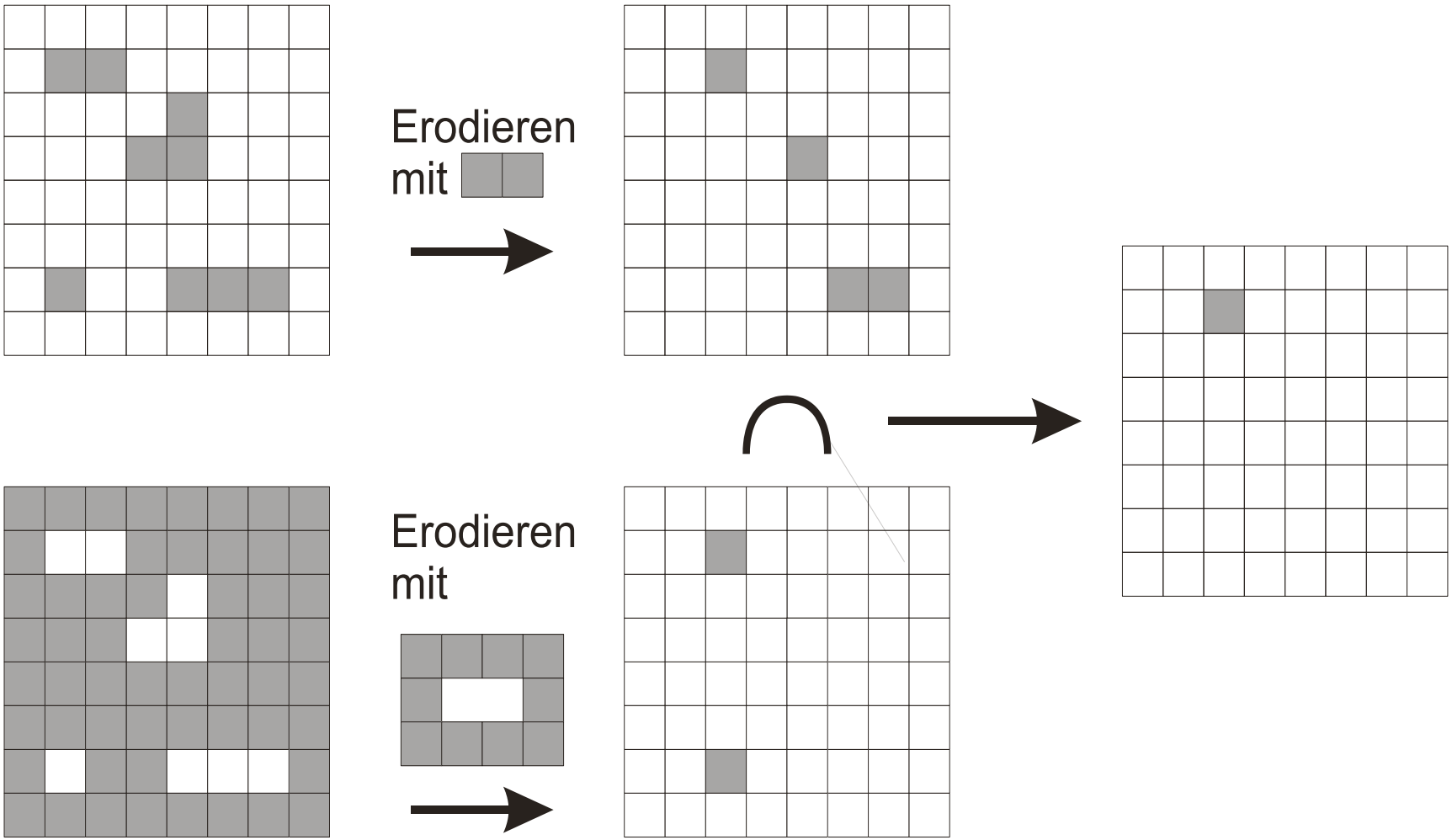
$$A_i = \frac{i \times A_B + (L - i) \times A_A}{L}$$

- Objekt einer Zwischenstufe i sind diejenigen Pixel, für die im i-ten Distanzbild A_i die Distanzen positiv sind.

Beispiel



Hit-or-Miss Operator



Hit-or-Miss Operator

Hit-or-Miss Operator: $G \otimes (S_1,S_2) = (G \ominus S_1) \cap (\overline{G} \ominus S_2)$

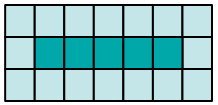
$= (G \overline{\ominus} S_1) \cap \overline{(G \oplus S_2)}$

mit $M_1 \cap M_2 = \emptyset$ (sonst wäre das Resultat der Operation die leere Menge)

Hit-or-Miss-Operator für variable Strukturgrößen, z.B.:



Hit



Miss

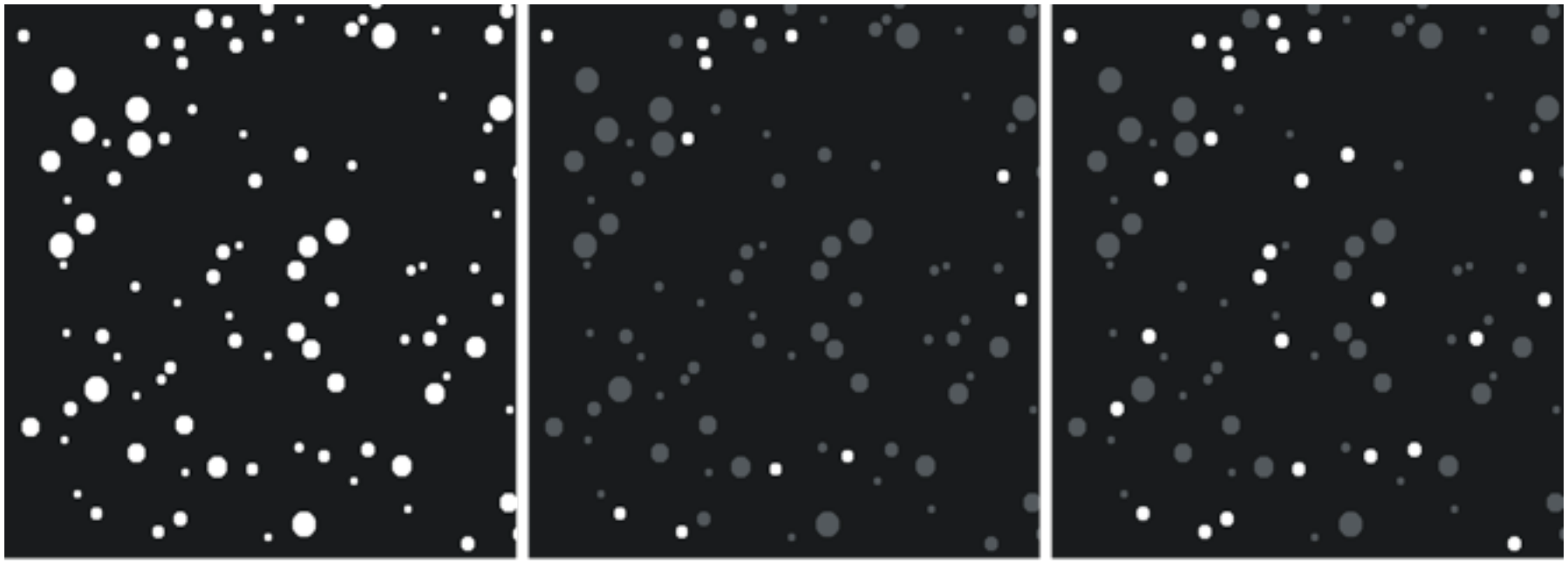
führt zur Akzeptanz von horizontalen Linien von 3,4, und 5 Pixeln Länge.

Notation für Hit-or-Miss-Operator:

0 - Miss
1 - Hit
x - weder Miss noch Hit

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x & 1 & 1 & 1 & x & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Beispiel



Kreise mit Radius von 6 Pixel

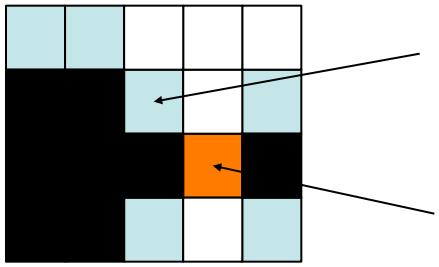
Kreise mit Radius 6-7 Pixel

Hit-or-Miss-Operatoren

M_I = 0 0 0 / 0 1 0 / 0 0 0 — Entfernung einzelner Pixel

M_C = x 1 0 / 1 1 0 / 0 0 0 — detektiert untere, rechte Ecken eines Objekts

M_{T1} = 0 0 0 / x 1 x / 1 1 1 — findet alle Randpunkte von oben, die ein Objekt nicht teilen würden, wenn sie entfernt würden.



Diese Punkte würden gefunden werden.

Diese Punkte würden nicht gefunden werden.

Zusammenfassung 2. Teil

- Morphologische Operationen: Formverändernde oder formauswertende Operationen auf Segmenten.
- Morphologische Filter zur:
  - Unterdrückung von Artefakten nach einer Segmentierung
  - Suche nach vorgegebenen Formen
  - Randbestimmung, Distanztransformation und Morphing