

Bildverarbeitung

Jan Fässler

5. Semester (HS 2013)

Inhaltsverzeichnis

1 Einführung	1
1.1 Elektromagnetische Strahlung	1
1.2 Sensor-Arrays	1
1.3 Farbbilderzeugung	1
1.4 Digitales Rasterbild	2
1.5 Raster- vs. Vektordaten	2
1.6 Wertebereiche von Pixeln	3
1.7 Region of Interest	3
2 Bildspeicherung	4
2.1 Bildformate	4
2.2 PGM-Beispiel	4
2.3 TIFF-Struktur	4
2.4 JPEG-Prozesskette	5
2.5 Bildkompression	5
2.6 Codierung	5
2.6.1 mittlere Codelänge einer Symbolfolge	5
2.6.2 Huffman-Code	6
2.7 Unterabtastung	6
2.8 quantisierung	6
3 Punktoperationen	7
3.1 Definition	7
3.1.1 homogen	7
3.1.2 nicht homogen	7
3.2 Kontrast und Helligkeit	7
3.3 Automatische Kontrastanpassung	7
3.4 Schwellwertoperation	8
3.5 Histogrammausgleich	8
3.6 Gammakorrektur	8
3.7 Bildqualität	9
4 Farben	10
4.1 Farbsysteme	10
4.1.1 lineare Farbsysteme	10
4.1.2 nicht lineare Farbsysteme	10
4.2 Farträume	11
4.2.1 RGB	11
4.2.2 HSV und HLS	11
4.2.3 CIExyz	11
4.2.4 CIExy	11
4.3 Umwandlungen	12
4.3.1 CMY und CMYK	12
4.3.2 CIExy	12
4.3.3 Transformation sRGB → XYZ	12
4.4 Weisspunkte	12
5 Filter	14
5.1 Anwendung	14
5.2 Gauss-Filter	14
5.3 Laplace-Filter	14
5.4 Faltung	15
5.4.1 diskrete Faltung	15
5.4.2 Eigenschaften der Faltung	15
5.5 Separierbarkeit	15

6 Morphologie	16
6.1 Schrumpfen und Wachsen	16
6.2 Grundoperationen	16
6.3 Zusammengesetzte Operationen	16
6.4 Grauwert-Morphologie	17
7 Binärbildanalyse	18
7.1 Prozesskette	18
7.2 Flood filling	18
7.3 Rekursives Flood Filling	18
7.4 Hough-Transformation	18
8 Mustererkennung (Pattern Matching)	19
8.1 Definitionen	19
8.2 Ähnlichkeitsmasse	19
8.3 Chamfer-Algorithmus	19
9 Fourier-Transformation	20
9.1 Frequenz, Amplitude, Phase	20
9.2 Fourierreihe	20
9.3 Fourierintegral und -spektrum	20
9.4 Fouriertransformation	20
9.5 Fourier-Transformationspaare	21
9.6 FT von diskreten Signalen	21
9.7 Aliasing und Abtasttheorem	21
9.8 Diskrete Fouriertransformation	22
9.9 FFT und DCT	23
9.10 DFT in 2D	23
9.11 Implementierung der 2D-DFT	23
9.12 SD Algorithmus	24
10 Kanten und Ecken	25
10.1 Übersicht	25
10.2 Gradient	25
10.2.1 Ableitungsfilter	25
10.2.2 Kantendetektierung	26
10.2.3 1. und 2. Ableitung	26
10.3 Laplacian of Gaussian	26
10.3.1 Kantenschärfung	27
10.3.2 Anwendung des Laplace-Filters	27
10.4 Eckpunkte	27
10.4.1 Anforderungen	28
10.4.2 Strukturmatrix	28
10.4.3 Eigenwerte und Eigenvektoren	28
10.4.4 Ähnliche Matrizen	28
10.4.5 Diagonalisierte Strukturmatrix	28
10.4.6 Spezifische Interpretation	29
10.4.7 Corner Response Function (CRF)	29
11 Wavelet-Transformation	30
11.1 Einsatz der WT	30
11.2 Idee der Bildpyramide	30
11.3 L_2 -Norm	30
11.4 Mother Wavelet und Wavelet-Familien	31
11.5 Continuous Wavelet Transform	31
11.6 CWT, DWT und FWT	31
11.7 FWT	32
11.8 FWT als Filteroperation	32

11.9 Rücktransformation	33
11.10FWT Filterbank	33
11.11FWT in zwei Dimensionen	33
11.12Bilddaten-Dekorrelation	34
11.13Quantisierung	34
12 PGF	35
12.1 Features	35
12.2 Aufbau	35
12.3 Farbtransformation	35
12.4 5/3 Wavelet Filterbank	35
12.5 Skalare Quantisierung	36
12.6 Progressive Kodierung	36
12.7 Bitplane-Kodierung	37
12.8 Lauflängenkodierung (RLE)	37
12.9 Adaptives RLE	37

1 Einführung

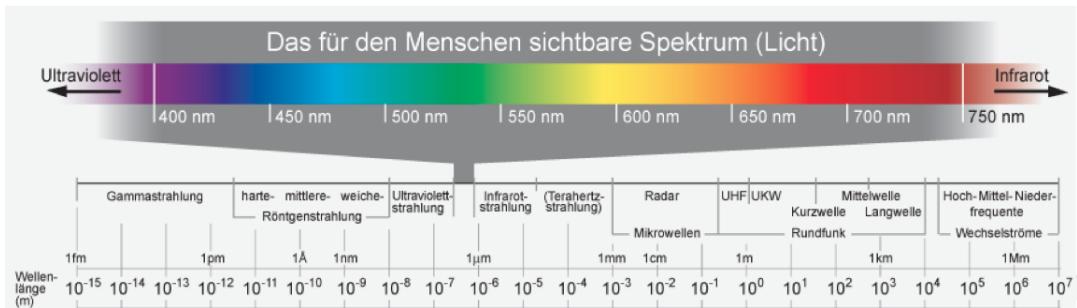
1.1 Elektromagnetische Strahlung

Wellenlänge

$$\lambda = \frac{c}{f} [m], \text{ mit } c = \text{Lichtgeschwindigkeit und } f = \text{Frequenz}$$

Energie der elektromagnetischen Strahlung

$$E = h * f [eV], \text{ mit } h = 6.626 * 10^{-34} \text{ Js} = 4.136 * 10^{-15} \text{ eVs}$$



1.2 Sensor-Arrays

Funktionsprinzip

- Ladungen werden in einem Kondensator gesammelt
- die Ladungsmenge ist proportional zur eingestrahlten Lichtmenge, wenn rechtzeitig ausgelesen wird, bevor die Leerlaufspannung der Fotodiode erreicht ist

CCD

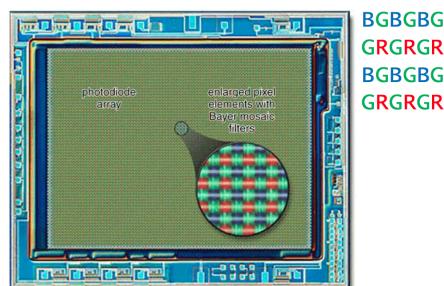
- Zeilen werden der Reihe nach ausgelesen
- pro Zeile werden die Ladungen serialisiert
- alle Ladungsmengen werden durch einen einzigen Ausleseverstärker geleitet

CMOS

- zu jeder Fotodiode gehört ein eigener Verstärker, welcher die Kondensatorspannung dem Analogsignalprozessor direkt zur Verfügung stellt

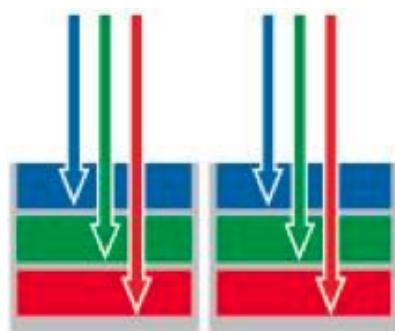
1.3 Farbbilderzeugung

Bayer-Maske

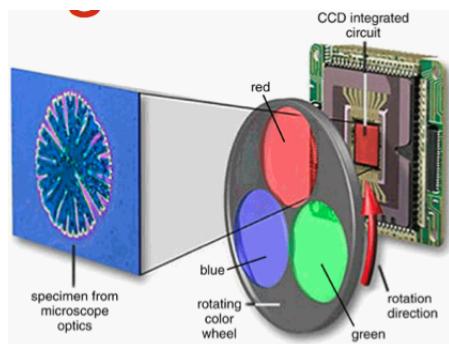


Andere

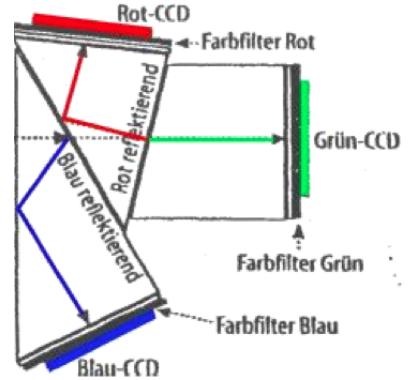
Foveon



Filterrad mit RGB-Filter



Spektralzerlegung



1.4 Digitales Rasterbild

Entstehung eines digitalen Rasterbildes

1. räumliche Abtastung

Übergang von einer räumlichen zu einer diskreten Lichtverteilung durch Geometrie des Aufnahmesensors (z.B. CCD-Chip)

2. zeitliche Abtastung

Steuerung der Zeit, über die die Lichtmessung stattfindet

3. Quantisierung der Pixelwerte

gemessene Lichtwerte werden auf eine endliche Menge von Zahlenwerten abgebildet

1.5 Raster- vs. Vektordaten

Rasterbilder

- regelmässige Matrix (mit diskreten Koordinaten) von Pixelwerten
- typische Formate: BMP, JFIF, TIFF, PNG, PGF usw.

Vektorgrafiken

- geometrische Beschreibung mit kontinuierlichen Koordinaten
- Rasterung (falls notwendig) erfolgt erst bei der Ausgabe
- typische Formate: SVG, DXF usw.

Metaformate

- oft werden Vektorgrafiken und Rasterbilder kombiniert
- typische Formate: CGM, AI, PICT, WMF, PS, EPS, PDF

1.6 Wertebereiche von Pixeln

Grauwertbilder (Intensitätsbilder):

Kanäle	Bit/Pixel	Wertebereich	Anwendungen
1	1	[0...1]	Binärbilder: Dokumente, Illustration, Fax
1	8	[0...255]	Universell: Foto, Scan, Druck
1	12	[0...4095]	Hochwertig: Foto, Scan, Druck
1	14	[0...16383]	Professionell: Foto, Scan, Druck
1	16	[0...65535]	Höchste Qualität: Medizin, Astronomie

Farbbilder:

Kanäle	Bits/Pixel	Wertebereich	Anwendungen
3	24	[0...255] ³	RGB, universell: Foto, Scan, Druck
3	36	[0...4095] ³	RGB, hochwertig: Foto, Scan, Druck
3	42	[0...16383] ³	RGB, professionell: Foto, Scan, Druck
4	32	[0...255] ⁴	CMYK, digitale Druckvorstufe

Spezialbilder:

Kanäle	Bits/Pixel	Wertebereich	Anwendungen
1	16	-32768...32767	Pos./neg. short integers, interne Verarbeitung
1	32	$\pm 3.4 \cdot 10^{38}$	Gleitkomma: Medizin, Astronomie
1	64	$\pm 1.8 \cdot 10^{308}$	Gleitkomma: interne Verarbeitung

1.7 Region of Interest

Grundidee

- nicht immer interessieren alle Teile eines Bild gleichermassen
- der oder die Bildbereiche, welche von Interesse sind, werden als ROIs bezeichnet

Bildformate mit ROI-Unterstützung

- JPEG 2000 (ROIs werden weniger stark komprimiert)
- PGF (nur ROI wird dekomprimiert)

Bestimmung von ROIs

- manuell (Benutzerin wählt den ROI selber aus)
- automatisch
 - Regionen mit Pixel einer bestimmten Intensität/Farbe
 - Regionen mit erhöhter Dichte
 - Regionen mit bestimmten Mustern (Mustererkennung)
 - Regionen die visuell hervorragen (Saliency Detection)

2 Bildspeicherung

2.1 Bildformate

ohne Kompression

- PNM: Portable Bitmap/Grayscale/Color Format
- RAS: Sun Raster Format
- BMP: Windows Bitmap

mit verlustloser Kompression

- GIF: Graphics Interchange Format
- PNG: Portable Network Graphics
- TIFF: Tagged Image File Format

nur mit verlustloser Kompression

- JFIF: JPEG File Interchange Format
- EXIF: Exchangeable Image File Format (Variante von JFIF)

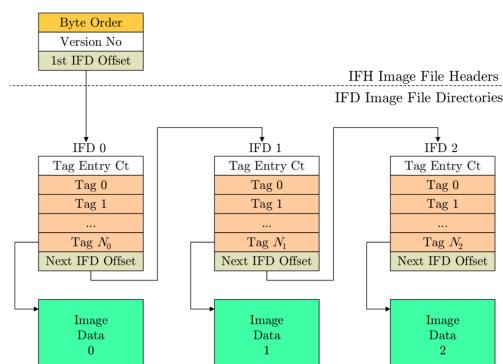
mit verlustloser/verlustbehafteter Kompression

- JPEG-2000
- PGF: Progressive Graphics File

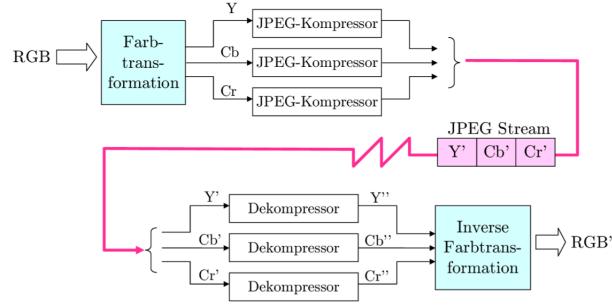
2.2 PGM-Beispiel

```
P2
# oie.pgm
17 7
255
0 13 13 13 13 13 13 13 0 0 0 0 0 0 0 0 0
0 13 0 0 0 0 0 13 0 7 7 0 0 81 81 81 81
0 13 0 7 7 7 0 13 0 7 7 0 0 81 81 0 0 0
0 13 0 7 0 7 0 13 0 7 7 0 0 81 81 81 0
0 13 0 7 7 7 0 13 0 7 7 0 0 81 0 0 0
0 13 0 0 0 0 0 13 0 7 7 0 0 81 81 81 81
0 13 13 13 13 13 13 13 0 0 0 0 0 0 0 0 0
```

2.3 TIFF-Struktur



2.4 JPEG-Prozesskette



2.5 Bildkompression

Das Ziel ist ein Bild auf dem Speichermedium so zu verdichten, dass es möglichst wenig Speicher benötigt, unter der Nebenbedingung, dass die Bildqualität möglichst gut ist.

$$\text{Kompressionsrate } r = \frac{\text{mem}_{\text{orig}}}{\text{mem}_{\text{comp}}}$$

verlustlos (Codierung)

- Entropiecodierung (Huffman, arithmetisch usw.)
- Präcodierung (RLE, LZ usw.)

verlustbehaftet (Datenreduktion)

- Farbreduktion (skalare Quantisierung / Vektorquantisierung)
- Transformation (Fourier / Wavelet) und Quantisierung der Koeffizienten

2.6 Codierung

$$\text{Codierungsredundanz } \Delta R_{\text{cod}} = S - H$$

N_A	Anzahl Symbole im Signal
N_B	gesamte Datenmenge des Signals in Bit
$S = N_B/N_A$	mittlere Anzahl Bits pro Symbol
K	Anzahl verschiedener Symbole
p_i	Symbol-Wahrscheinlichkeit
$H = -\sum_{i=1}^K p_i * ld(p_i)$	Entropie (mittlerer Symbolinformationsgehalt)
$H_{\text{tot}} = n * H$	Totaler Informationsgehalt bei n Symbolen

2.6.1 mittlere Codelänge einer Symbolfolge

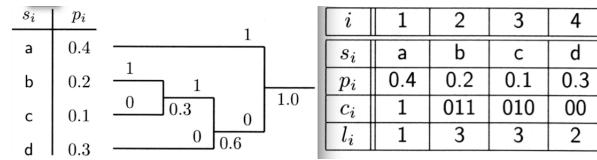
- Alphabet $\{s_1, s_2, \dots, s_K\}$
- mit $p_i = \text{Auftretenswahrscheinlichkeit des Symbols } s_i$

$$l = \sum_{i=1}^K p_i * ||c_i|| \text{ [Bit/Symbol]}$$

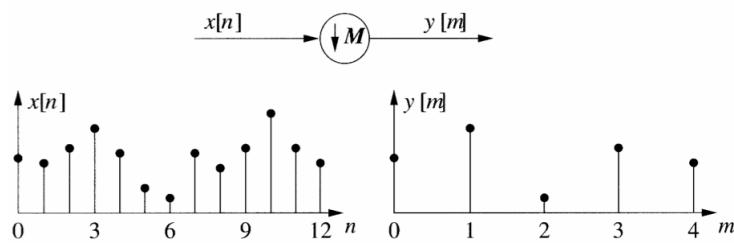
Untere Schranke $H \leq l \leq H + 1$ Obere Schranke

2.6.2 Huffman-Code

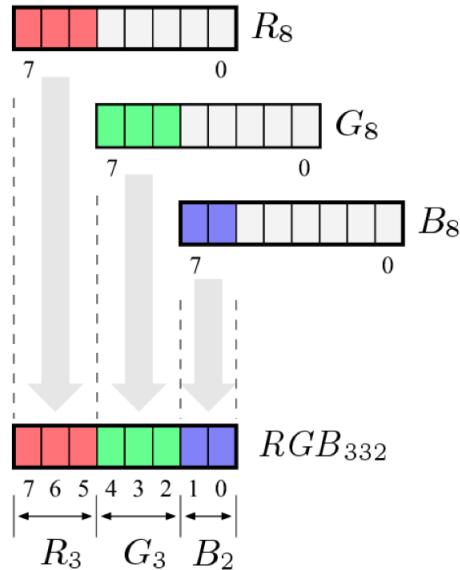
1. Betrachte alle Symbole als Blätter eines Codebaums und trage ihre Wahrscheinlichkeiten ein.
2. Fasse die beiden geringsten Wahrscheinlichkeiten zu einem Knoten zusammen und weise ihre Summe dem Knoten zu.
3. Beschrifte die neuen Äste mit 0 und 1.
4. Fahre bei Schritt 2 fort, so lange die Wurzel mit Wahrscheinlichkeit $p = 1$ noch nicht erreicht worden ist.



2.7 Unterabtastung



2.8 quantisierung



3 Punktoperationen

3.1 Definition

3.1.1 homogen

$$I'(u, v) \leftarrow f(I(u, v))$$

Typische Beispiele:

- Änderungen von Kontrast und Helligkeit
- Invertieren von Bildern
- Quantisieren der Helligkeit (Poster-Effekt)
- Schwellwertbildung
- Gammakorrektur
- Farbtransformation

3.1.2 nicht homogen

$$I'(u, v) \leftarrow g(I(u, v), u, v)$$

Typisches Beispiel:

- selektive Kontrast - oder Helligkeitsanpassung

3.2 Kontrast und Helligkeit

Erhöhung des Kontrasts um 50%:

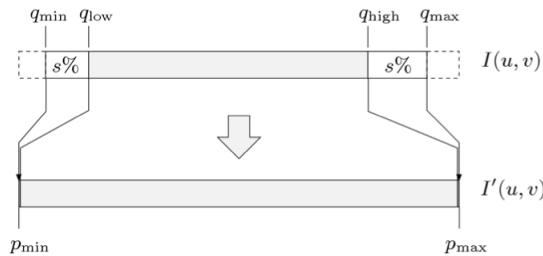
$$I'(u, v) \leftarrow f(I(u, v)) * 1.5$$

Anheben der Helligkeit um 10 Stufen:

$$I'(u, v) \leftarrow f(I(u, v)) + 10$$

3.3 Automatische Kontrastanpassung

$$I'(u, v) \leftarrow \begin{cases} p_{min}, & \text{für } I(u, v) \leq q_{low} \\ p_{min} + (I(u, v) - q_{low}) * \frac{p_{max} - p_{min}}{q_{high} - q_{low}}, & \text{für } q_{low} < I(u, v) < q_{high} \\ p_{max}, & \text{für } I(u, v) \geq q_{high} \end{cases}$$

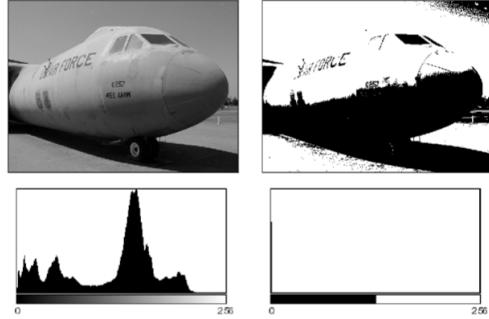


3.4 Schwellwertoperation

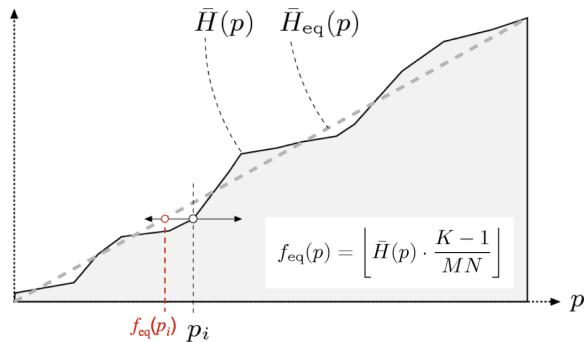
Thresholding

- Reduktion auf zwei Intensitätswerte → Binarisierung
- Spezialfall der Quantisierung (Graustufenreduktion)

$$I'(u, v) \leftarrow f_{th}(I(u, v)) \begin{cases} p_0, & \text{für } I(u, v) < p_{th} \\ p_1, & \text{für } I(u, v) \geq p_{th} \end{cases}$$



3.5 Histogrammausgleich



3.6 Gammakorrektur

einfache Gammakorrektur

$$q = GC(p, \gamma) = \left(\frac{p}{p_{max}} \right)^\gamma * p_{max}$$

Probleme

- Anstieg der Gammafunktion in der Nähe des Nullpunktes
- starke Rauschanfälligkeit wegen der extrem hohen Verstärkung

Modifizierte Gammafunktion

$$f_{(\gamma, x_0)}(x) = \begin{cases} s * x, & \text{für } 0 \leq x \leq x_0 \\ (1 + d) * x^\gamma - d, & \text{für } x_0 < x \leq 1 \end{cases}$$

$$s = \frac{\gamma}{x_0(\gamma - 1) + x_0^{1-\gamma}} \quad d = \frac{1}{x_0^\gamma(\gamma - 1) + 1} - 1$$

3.7 Bildqualität

- technische Bildqualität ist ein Mass für die Abweichung zwischen Original (o) und Kopie (k)
- eine subjektive Qualitätsmessung eines Betrachters könnte anders ausfallen

root mean squared error

Bildkanal mit n Pixeln:

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (o[i] - k[i])^2}{n}}$$

peak signal-to-noise ratio in dB

Bildkanal mit maximaler Intensität 255:

$$PSNR = 20 * \log_{10} \left(\frac{255}{RMSE} \right)$$

4 Farben

4.1 Farbsysteme

4.1.1 lineare Farbsysteme

RGB:

- Rot, Grün, Blau
- additive Farbmischung

CMY:

- Cyan (Türkis), Magenta (Purpur), Yellow (Gelb)
- subtraktive Farbmischung

CMY:

- Cyan (Türkis), Magenta (Purpur), Yellow (Gelb)
- subtraktive Farbmischung

YUV, YIQ und YC_bC_r :

- Fernseh-Komponentenfarbräume

CIEXYZ:

- CIE Standard-Farbraum

4.1.2 nicht lineare Farbsysteme

CMYK:

- Farbmischung für den 4-Farbendruck: CMY und Schwarz (K)

HSV/HSB/HSI/HLS:

- Hue (Farbton), Saturation (Sättigung), Value/Brightness/Intensity/Luminance (Helligkeit)

CIExy:

- Koordinatensystem des CIE-Farbdigramms
- einfache Umrechnung zwischen CIEXYZ und CIExy
- nicht intuitiv zu verstehen

CIE L*a*b*:

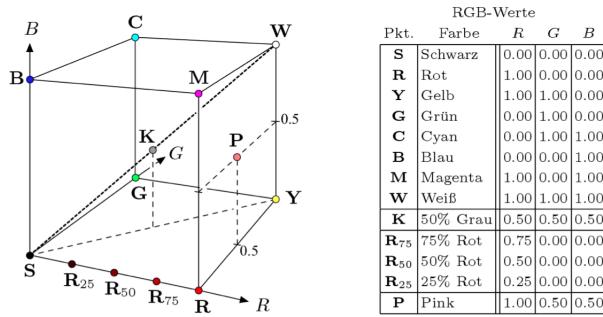
- intuitiv verständlich
- möglichst linear gegenüber dem menschlichen Sehen

sRGB:

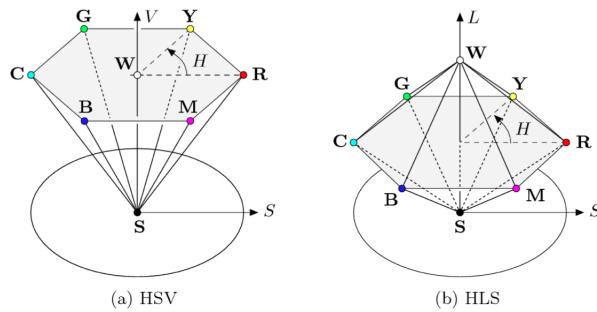
- Standard RGB (präzise definiert, basierend auf CIE xy)
- relativ kleines Gamut (für digitale Anwendungen genügend)

4.2 Farbräume

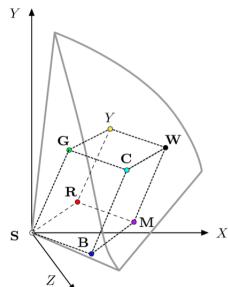
4.2.1 RGB



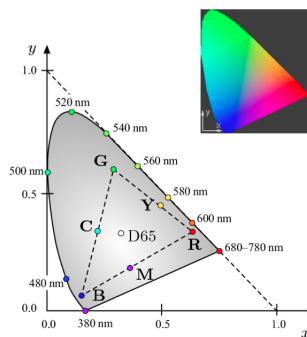
4.2.2 HSV und HLS



4.2.3 CIExyz



4.2.4 CIExy



4.3 Umwandlungen

4.3.1 CMY und CMYK

RGB → CMY:

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

CMY → CMYK:

$$\begin{aligned} C' &= C - K \\ M' &= M - K \\ Y' &= Y - K \\ K' &= K \end{aligned}$$

4.3.2 CIExy

XYZ → xy:

$$x = \frac{X}{X + Y + Z} \quad y = \frac{Y}{X + Y + Z}$$

xy → XYZ:

$$X = \frac{x}{y} \quad Y = 1 \quad Z = \frac{z}{y} = \frac{1 - x - y}{y}$$

4.3.3 Transformation sRGB → XYZ

Nichtlineare sRGB-Komponenten:

$$R = f_2(R') \quad G = f_2(G') \quad B = f_2(B')$$

Inverse modifizierte Gammakorrektur:

$$f_2(c) = \begin{cases} \left(\frac{c+0.055}{1.055}\right)^{2.4}, & \text{wenn } c > 0.03928 \\ \frac{c}{12.92}, & \text{wenn } c \leq 0.03928 \end{cases}$$

Lineare Transformation von RGB nach CIE XYZ:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.4121 & 0.3576 & 0.1805 \\ 0.2126 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

4.4 Weisspunkte

D50

- Farbtemperatur von 5000 Kelvin
- ähnlich dem direkten Sonnenlicht
- Referenzbeleuchtung für die Betrachtung von reflektierenden Bildern wie z.B. von Druckern

D65

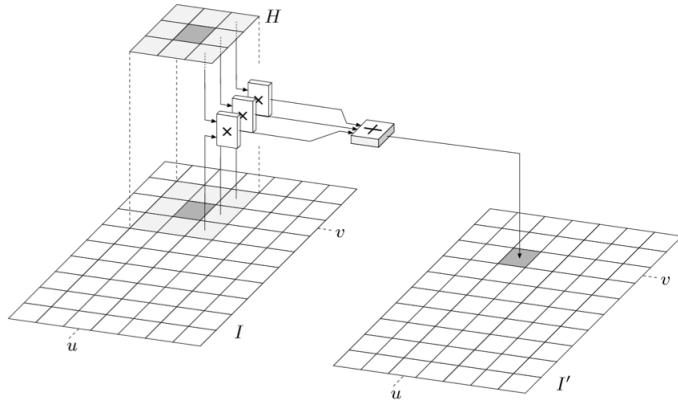
- Farbtemperatur von 6500 Kelvin
- durchschnittliche indirekte Tageslichtbeleuchtung
- Normweisslicht für emittierende Wiedergabegeräte z.B. von Bildschirme

Chromatische Adaptierung:

$$\begin{pmatrix} X_{65} \\ Y_{65} \\ Z_{65} \end{pmatrix} = M_{65|50} * \begin{pmatrix} X_{50} \\ Y_{50} \\ Z_{50} \end{pmatrix} = \begin{pmatrix} 0.955556 & -0.023049 & 0.063197 \\ -0.028302 & 1.009944 & 0.021018 \\ 0.012305 & -0.020494 & 1.330084 \end{pmatrix} * \begin{pmatrix} X_{50} \\ Y_{50} \\ Z_{50} \end{pmatrix}$$

5 Filter

5.1 Anwendung



$$I'(u, v) \leftarrow \sum_{(i,j) \in \mathbb{R}} I(u+i, v+j) * H(i, j)$$

5.2 Gauss-Filter

diskrete, zweidimensionale Gauss-Funktion:

$$G_\sigma(r) = e^{-\frac{r^2}{2\sigma^2}} \quad G_\sigma(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Glättungsfilter (Tiefpassfilter):

- weil alle Filterkoeffizienten positiv sind
- annähernd isotrop ab Filtergrößen von 5 x 5 (nach allen Richtungen hin gleichförmig arbeiten)
- gutmütiges Frequenzverhalten (stetig, differenzierbar)

5.3 Laplace-Filter

Interpretation als Differenz von zwei Summen:

$$I'(u, v) = \sum_{(i,j) \in \mathbb{R}^+} I(u+i, v+j) * |H(i, j)| - \sum_{(i,j) \in \mathbb{R}^-} I(u+i, v+j) * |H(i, j)|$$

$\mathbb{R}^+/\mathbb{R}^-$ Teil der Filterregion mit positiven/negativen Koeffizienten

Verstärken von Kanten und Konturen:

- örtliche Intensitätsunterschiede werden verstärkt
- richtungsunabhängige Detektion von Kanten
- Hochpassfilter

5.4 Faltung

5.4.1 diskrete Faltung

$$\begin{aligned}
 I'(u, v) &= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(u - i, v - j) \cdot H(i, j) \\
 I' &= I * H \\
 I'(u, v) &= \sum_{(i,j) \in \mathbb{R}} I(u - i, v - j) \cdot H(i, j) \\
 &= \sum_{(i,j) \in \mathbb{R}} I(u + i, v + j) \cdot H(-i, -j)
 \end{aligned}$$

5.4.2 Eigenschaften der Faltung

Kommutativität:

$$I * H = H * I$$

Linearität:

$$\begin{aligned}
 (a \cdot I) * H &= I * (a \cdot H) = a \cdot (I * H) \\
 (I_1 + I_2) * H &= (I_1 * H) + (I_2 * H)
 \end{aligned}$$

Assoziativität:

$$A * (B * C) = (A * B) * C$$

5.5 Separierbarkeit

$$I' \leftarrow (I * H_x) * H_y = I * \underbrace{(H_x * H_y)}_{H_{xy}}$$

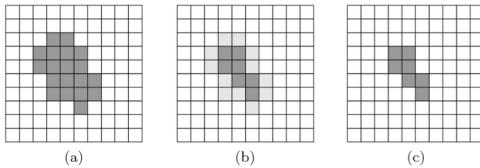
$$H_x^P = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad H_y^P = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$H_x^P = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad H_y^P = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

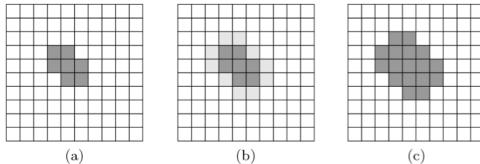
6 Morphologie

6.1 Schrumpfen und Wachsen

Schrumpfen



Wachsen lassen



Art der Nachbarschaft

$$\begin{array}{|c|c|c|} \hline & & \\ \hline & N_2 & \\ \hline N_1 & \times & N_4 \\ \hline & N_3 & \\ \hline \end{array}$$

N_2	N_3	N_4
N_1	\times	N_5
N_8	N_7	N_6

6.2 Grundoperationen

Strukturelement: binäre Matrix mit Hot-Spot als Ursprung des Koordinatensystems (analog zu einem Filter mit binären Koeffizienten)

Strukturelement als Punktmenge:

$$P_H = \{(i, j) | H(i, j)\}1\}$$

Bild als Punktmenge:

$$P_I = \{(u, v) | I(u, v)\}1\}$$

Erosion (Schrumpfen):

$$I \ominus H = \{(x, y) | (x + i, y + j) \in P_i, \forall (i, j) \in PH\}$$

$$\begin{array}{c} \begin{array}{|c|c|c|c|} \hline & 0 & 1 & 2 & 3 \\ \hline 0 & & \bullet & \bullet & \\ \hline 1 & & & & \\ \hline 2 & & & \bullet & \\ \hline 3 & & & & \\ \hline \end{array} & \oplus & \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline 0 & \bullet & \bullet \\ \hline 1 & & \\ \hline \end{array} & = & \begin{array}{|c|c|c|c|} \hline & 0 & 1 & 2 & 3 \\ \hline 0 & & \bullet & & \\ \hline 1 & & & \bullet & \\ \hline 2 & & & & \\ \hline 3 & & & & \\ \hline \end{array} \\ I & & H & & I \oplus H \end{array}$$

Dilation (Wachsen lassen):

$$I \oplus H = \{(x, y) = (u + i, v + j) \mid (u, v) \in P_i, \forall (i, j) \in PH\}$$

$$\begin{array}{c} \text{Table } I \\ \oplus \\ \text{Table } H \\ = \\ \text{Table } I \oplus H \end{array}$$

6.3 Zusammengesetzte Operationen

Opening

zuerst schrumpfen, dann wachsen lassen:

$$I \circ H = (I \ominus H) \oplus H$$

- sehr kleine Vordergrundstrukturen werden eliminiert

- glättet Kanten von grösseren Elementen
- Elemente mit kleinem Berührungs punkt werden getrennt

Closing

zuerst wachsen lassen, dann schrumpfen:

$$I \bullet H = (I \oplus H) \ominus H$$

- füllt kleine Löcher
- glättet Kanten von grösseren Elementen
- Elemente mit kleinem Berührungs punkt werden stärker verbunden

6.4 Grauwert-Morphologie

Grauwert-Dilation

$$(I \oplus H)(u, v) = \max_{(i,j) \in H} \{I(u+i, v+j) + H(i, j)\}$$

$$\begin{array}{c}
 I \\
 \begin{array}{|c|c|c|c|} \hline
 6 & 7 & 3 & 4 \\ \hline
 5 & 6 & 6 & 8 \\ \hline
 6 & 4 & 5 & 2 \\ \hline
 6 & 4 & 2 & 3 \\ \hline
 \end{array}
 \end{array}
 \oplus
 \begin{array}{c}
 H \\
 \begin{array}{|c|c|c|} \hline
 1 & 1 & 1 \\ \hline
 1 & 2 & 1 \\ \hline
 1 & 1 & 1 \\ \hline
 \end{array}
 \end{array}
 =
 \begin{array}{c}
 I \oplus H \\
 \begin{array}{|c|c|c|} \hline
 & & \\ \hline
 & 8 & 9 \\ \hline
 & 7 & 9 \\ \hline
 & & \\ \hline
 \end{array}
 \end{array}$$

$$\begin{array}{c}
 I + H \\
 \begin{array}{|c|c|c|} \hline
 7 & 8 & 4 \\ \hline
 6 & 8 & 7 \\ \hline
 7 & 5 & 6 \\ \hline
 \end{array}
 \end{array}
 \text{ max}$$

Grauwert-Erosion

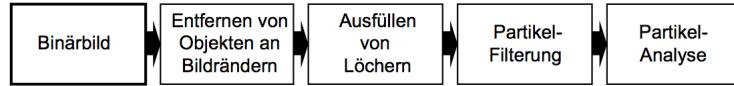
$$(I \ominus H)(u, v) = \min_{(i,j) \in H} \{I(u+i, v+j) - H(i, j)\}$$

$$\begin{array}{c}
 I \\
 \begin{array}{|c|c|c|c|} \hline
 6 & 7 & 3 & 4 \\ \hline
 5 & 6 & 6 & 8 \\ \hline
 6 & 4 & 5 & 2 \\ \hline
 6 & 4 & 2 & 3 \\ \hline
 \end{array}
 \end{array}
 \ominus
 \begin{array}{c}
 H \\
 \begin{array}{|c|c|c|} \hline
 1 & 1 & 1 \\ \hline
 1 & 2 & 1 \\ \hline
 1 & 1 & 1 \\ \hline
 \end{array}
 \end{array}
 =
 \begin{array}{c}
 I \ominus H \\
 \begin{array}{|c|c|c|} \hline
 & & \\ \hline
 & 2 & 1 \\ \hline
 & 1 & 1 \\ \hline
 & & \\ \hline
 \end{array}
 \end{array}$$

$$\begin{array}{c}
 I - H \\
 \begin{array}{|c|c|c|} \hline
 5 & 6 & 2 \\ \hline
 4 & 4 & 5 \\ \hline
 5 & 3 & 4 \\ \hline
 \end{array}
 \end{array}
 \text{ min}$$

7 Binärbildanalyse

7.1 Prozesskette



7.2 Flood filling

```

1: REGIONLABELING( $I$ )
2: Initialize  $m \leftarrow 2$  (the value of the next label to be assigned).
3: Iterate over all image coordinates  $(u, v)$ .
4:   if  $I(u, v) = 1$  then
5:     FLOODFILL( $I, u, v, m$ )
6:   Increment  $m$ .
7: return
  
```

- **3 Varianten**

- **rekursiv**

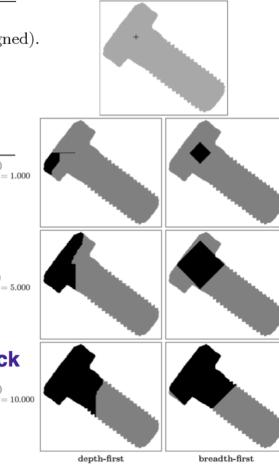
- meist untauglich, weil die Stackgrösse nicht ausreicht

- **iterativ mit Stack: depth-first**

- wie rekursiv, aber mit eigenem Stack

- **iterativ mit Queue: breadth-first**

- oft am speichereffizientesten



7.3 Rekursives Flood Filling

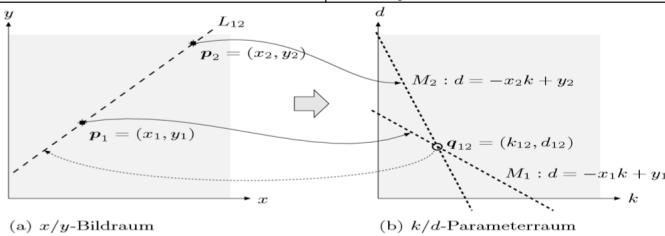
```

FloodFill( $I, u, v, label$ )
if  $(u, v)$  is within image boundaries  $\wedge I(u, v) = 1$  then
   $I(u, v) := label$ 
  FloodFill( $I, u + 1, v, label$ )
  FloodFill( $I, u, v + 1, label$ )
  FloodFill( $I, u, v - 1, label$ )
  FloodFill( $I, u - 1, v, label$ )
endif
end
  
```

7.4 Hough-Transformation

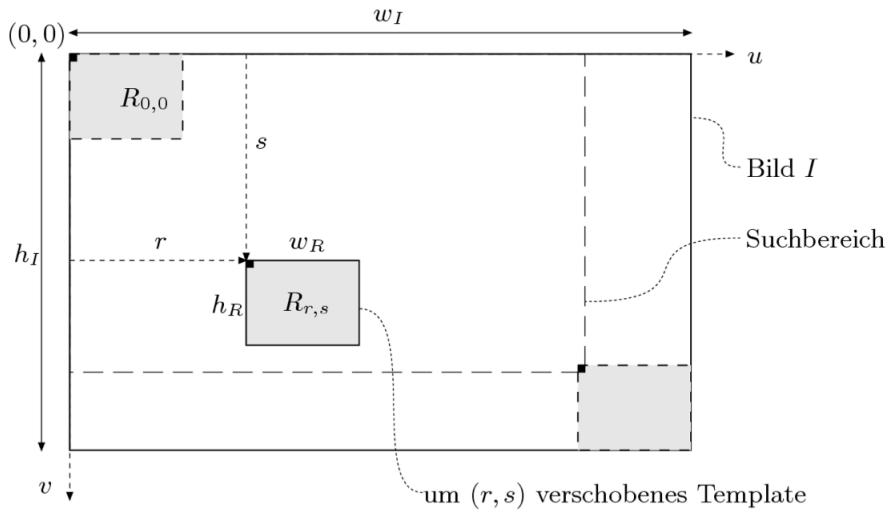
Die Grundidee ist, dass wir durch alle Kantenpixen in einem binären Kantenbild gehen. Ein solches Kantenpixel entspricht einem Punkt im Bildraum. Durch einen solchen Punkt im Bildraum können unendliche viele verschiedenen Geraden im Bildraum laufen. Diese können durch eine einzige Gerade im Parameterraum repräsentiert werden. Liegen mehrere Punkte im Bildraum auf derselben Geraden im Bildraum, so schneiden sich die entsprechenden Geraden im Parameterraum in einem einzigen Punkt.

Bildraum (x, y)		Parameterraum (k, d)	
Punkt	$p_i = (x_i, y_i)$	$M_i : d = -x_i k + y_i$	Gerade
Gerade	$L_j : y = k_j x + d_j$	$q_j = (k_j, d_j)$	Punkt



8 Mustererkennung (Pattern Matching)

8.1 Definitionen



8.2 Ähnlichkeitsmasse

Summe der Differenzbeträge:

$$d_A(r, s) = \sum_{(i,j) \in \mathbb{R}} |I(r+i, s+j) - R(i, j)|$$

Maximaler Differenzbetrag

$$d_M(r, s) = \max_{(i,j) \in \mathbb{R}} |I(r+i, s+j) - R(i, j)|$$

Summe der quadratischen Differenzbeträge (euklidische Distanz):

$$d_E(r, s) = \sqrt{\sum_{(i,j) \in \mathbb{R}} (I(r+i, s+j) - R(i, j))^2}$$

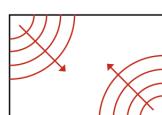
globale lineare Kreuzkorrelation:

$$(I \otimes R)(r, s) = \sum_{(i,j) \in \mathbb{R}} I(r+i, s+j) \cdot R(i, j)$$

8.3 Chamfer-Algorithmus

Idee

- wellenförmige Ausbreitung der Distanzwerte



Initialisierung

- $D(u, v) := (\#u, v) = 1 ? 0 : \infty$

Algorithmus

- erste Welle breitet sich von links oben nach rechts unten aus

if $D(u, v) > 0$ then $D(u, v) := \min(D(u+i, v+j) + M^L(i, j))$

$$M^L = \begin{bmatrix} m_2^L & m_3^L & m_4^L \\ m_1^L & \times & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$$

- zweite Welle breitet sich von rechts unten nach links oben aus

if $D(u, v) > 0$ then $D(u, v) := \min(D(u+i, v+j) + M^R(i, j))$

$$M^R = \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & 0 & m_1^R \\ m_4^R & m_3^R & m_2^R \end{bmatrix}$$

$$\text{Ähnlichkeitsmasse } Q(r, s) = \frac{1}{K} \sum_{(i,j) \in FG(R)} D(r+i, s+j)$$

```

1: CHAMFERMATCH ( $I, R$ )                                ▷ binary image  $I(u, v)$  of size  $w_I \times h_I$ 
   ▷ binary template  $R(u, v)$  of size  $w_R \times h_R$ 
2: STEP 1 – INITIALIZE:
3:  $D \leftarrow \text{DISTANCETRANSFORM}(I)$                   ▷ Alg. 17.1
4:  $K \leftarrow$  number of foreground pixels in  $R$ 
5: Set up a match map  $Q$  of size  $(w_I - w_R + 1) \times (h_I - h_R + 1)$ 
6: STEP 2 – COMPUTE MATCH FUNCTION:
7: for  $r \leftarrow 0 \dots (w_I - w_R)$  do                   ▷ set origin of model to  $(r, s)$ 
8:   for  $s \leftarrow 0 \dots (h_I - h_R)$  do
9:     EVALUATE MATCH FOR TEMPLATE POSITIONED AT  $(r, s)$ :
10:     $q \leftarrow 0$ 
11:    for  $i \leftarrow 0 \dots (w_R - 1)$  do
12:      for  $j \leftarrow 0 \dots (h_R - 1)$  do
13:        if  $R(i, j) = 1$  then                            ▷ foreground pixel in model
14:           $q \leftarrow q + D(r+i, s+j)$ 
15:     $Q(r, s) \leftarrow q/K$ 
16:  return  $Q$ 

```

9 Fourier-Transformation

9.1 Frequenz, Amplitude, Phase

$$\begin{aligned}
 \text{Beispiel: } & a \cdot \sin(\omega \cdot x - \varphi) \\
 \text{Kreisfrequenz: } & \omega = 2 \cdot \pi \cdot f \\
 \text{Frequenz: } & f = \frac{1}{T} = \frac{\omega}{2 \cdot \pi} \\
 \text{Periodenlänge: } & T \\
 \text{Amplitudde: } & a \\
 \text{Phase: } & \varphi
 \end{aligned}$$

9.2 Fourierreihe

Jede periodische Funktion $g(x)$ mit einer Grundfrequenz ω_0 kann als unendliche Summe von harmonischen Schwingungen dargestellt werden:

$$g(x) = \sum_{k=0}^{\infty} [A_k \cdot \cos(k \cdot \omega_0 \cdot x) + B_k \cdot \sin(k \cdot \omega_0 \cdot x)]$$

Fourieranalyse: Berechnung der Fourerkoeffizienten (A_k, B_k) aus einer gegebenen Funktion $g(x)$.

9.3 Fourierintegral und -spektrum

Eine nicht periodische Funktion $g(x)$ kann als Summe von unendlich vielen Sinus- und Kosinusschwingungen dargestellt werden. Das bedarf nicht nur Vielfache von der Grundfrequenz (\rightarrow Fourierintegral) sondern auch unendlich viele dicht aneinander liegende Frequenzen.

$$g(x) = \int_0^{\infty} A_{\omega} \cdot \cos(\omega \cdot x) + B_{\omega} \cdot \sin(\omega \cdot x) d\omega$$

Bestimmung des Fourierspektrums (Fourerkoeffizientenfunktionen):

$$\begin{aligned}
 A_{\omega} = A(\omega) &= \frac{1}{\pi} \cdot \int_{-\infty}^{\infty} g(x) \cdot \cos(\omega \cdot x) dx \\
 B_{\omega} = B(\omega) &= \frac{1}{\pi} \cdot \int_{-\infty}^{\infty} g(x) \cdot \sin(\omega \cdot x) dx
 \end{aligned}$$

9.4 Fouriertransformation

Übergang von Fourierintegral zu Fouriertransformation:

$$\begin{aligned}
 G(\omega) &= \sqrt{\frac{\pi}{2}} \cdot [A(\omega) - i \cdot B(\omega)] \\
 &= \sqrt{\frac{\pi}{2}} \cdot \left[\frac{1}{\pi} \cdot \int_{-\infty}^{\infty} g(x) \cdot \cos(\omega \cdot x) dx - i \cdot \frac{1}{\pi} \cdot \int_{-\infty}^{\infty} g(x) \cdot \sin(\omega \cdot x) dx \right] \\
 &= \frac{1}{\sqrt{2\pi}} \cdot \int_{-\infty}^{\infty} g(x) \cdot [\cos(\omega \cdot x) - i \cdot \sin(\omega \cdot x)] dx
 \end{aligned}$$

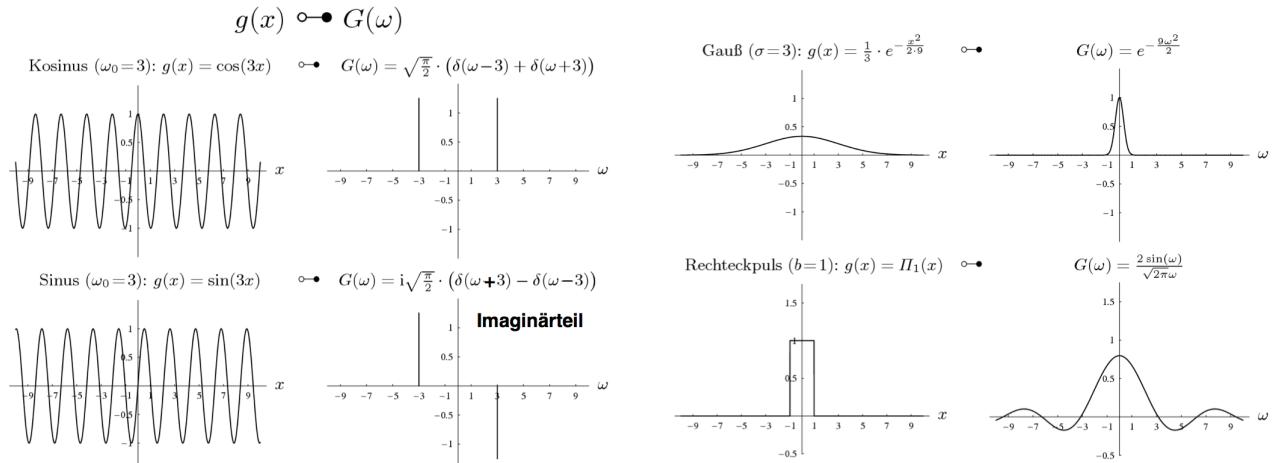
Vorwärtstransformation

$$G(\omega) = \frac{1}{\sqrt{2\pi}} \cdot \int_{-\infty}^{\infty} g(x) \cdot e^{-i\omega x} dx$$

Rücktransformation

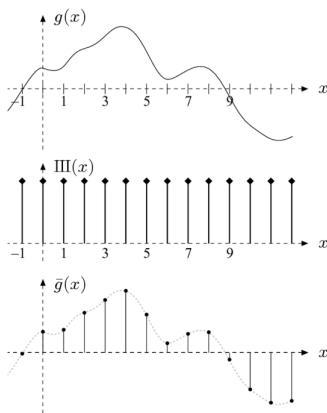
$$g(x) = \frac{1}{\sqrt{2\pi}} \cdot \int_{-\infty}^{\infty} G(\omega) \cdot e^{i\omega x} d\omega$$

9.5 Fourier-Transformationspaare



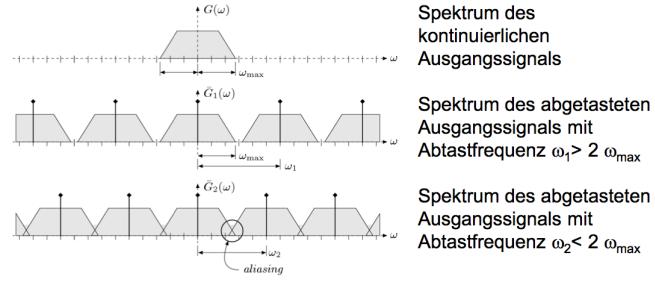
9.6 FT von diskreten Signalen

- Abtastung (Sampling)**
- **Abtastung = Multiplikation mit Kämmfunktion**
 - **durch Abtastung wird aus einer kontinuierlichen Ausgangsfunktion $g(x)$ eine diskrete Funktion**
- Auswirkungen**
- **Diskretisierung im Ortsraum führt zu Periodizität im Frequenzraum (Frequenzraum)**
 - **Invers zu: Periodizität im Ortsraum führt zu diskretem Frequenzspektrum (→ Fourierreihe)**



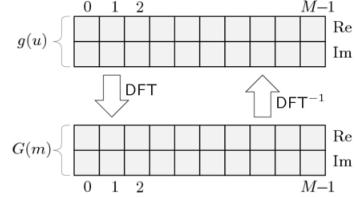
9.7 Aliasing und Abtasttheorem

- Diskretisierung im Ortsraum → Periodizität im Frequenzraum
- falls die sich wiederholenden Spektralkomponenten im Frequenzraum nicht überschneiden, so ist eine verlustlose Rücktransformation möglich
- maximal zulässige Signalfrequenz ω_{max} ist von der Abtastfrequenz ω_s abhängig



9.8 Diskrete Fouriertransformation

$$\begin{aligned}
 \text{Periodenlänge } t_0: & M \text{ Abtastwerte im Abstand } t_s \\
 \text{Frequenz } f_0 = & \frac{1}{M \cdot t_s} \\
 \text{Abtastfrequenz } f_s = & \frac{1}{t_s} = M \cdot f_0 \\
 \text{Wellenzahl } m: & 0 \leq m < M \\
 \text{Kreisfrequenz } \omega & = m \cdot \omega_0 = 2\pi \cdot m \cdot f_0
 \end{aligned}$$



Leistungsspektrum

$$|G(m)| = \sqrt{G_{Re}^2(m) + G_{Im}^2(m)}$$

Phasenspektrum

$$Pha(m) = \arctan \left(\frac{G_{Im}(m)}{G_{Re}(m)} \right)$$

Implementation

```

Complex[] DFT(Complex[] g, boolean forward) {
    int M = g.length;
    double s = 1 / Math.sqrt(M); //common scale factor
    Complex[] G = new Complex[M];
    for (int m = 0; m < M; m++) {
        double sumRe = 0;
        double sumIm = 0;
        double phim = 2 * Math.PI * m / M;
        for (int u = 0; u < M; u++) {
            double gRe = g[u].re;
            double gIm = g[u].im;
            double cosw = Math.cos(phim * u);
            double sinw = Math.sin(phim * u);
            if (!forward) // inverse transform
                sinw = -sinw;
            //complex mult: (gRe + i gIm) * (cosw + i sinw)
            sumRe += gRe * cosw - gIm * sinw;
            sumIm += gIm * cosw + gRe * sinw;
        }
        G[m] = new Complex(s * sumRe, s * sumIm);
    }
    return G;
}

```

9.9 FFT und DCT

Zeitkomplexität der DFT

- zwei verschachtelte for-Schleifen von 0 bis M
- $O(M^2)$

Fast Fourier Transform (FFT)

- z.B. Algorithmus von Cooley und Tukey
- Optimierung auf Signallängen von $M = 2^k$
- Reduktion der Zeitkomplexität auf $O(M \cdot \log M)$

Discrete Cosine Transform (DCT)

- nur für reelle Signale geeignet
- Spektrum ist auch reell
- Transformation verwendet nur Kosinusfunktionen

9.10 DFT in 2D

Vorwärtstransformation

$$\begin{aligned} G(m, n) &= \frac{1}{\sqrt{MN}} \cdot \sum_{u=0}^{M-1} g(u, v) \cdot e^{-i2\pi \frac{m \cdot u}{M}} \cdot e^{-i2\pi \frac{n \cdot v}{N}} \\ &= \frac{1}{\sqrt{MN}} \cdot \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} g(u, v) \cdot e^{-i2\pi \left(\frac{m \cdot u}{M} + \frac{n \cdot v}{N} \right)} \end{aligned}$$

Rücktransformation

$$\begin{aligned} g(u, v) &= \frac{1}{\sqrt{MN}} \cdot \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(u, v) \cdot e^{i2\pi \frac{m \cdot u}{M}} \cdot e^{i2\pi \frac{n \cdot v}{N}} \\ &= \frac{1}{\sqrt{MN}} \cdot \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(u, v) \cdot e^{i2\pi \left(\frac{m \cdot u}{M} + \frac{n \cdot v}{N} \right)} \end{aligned}$$

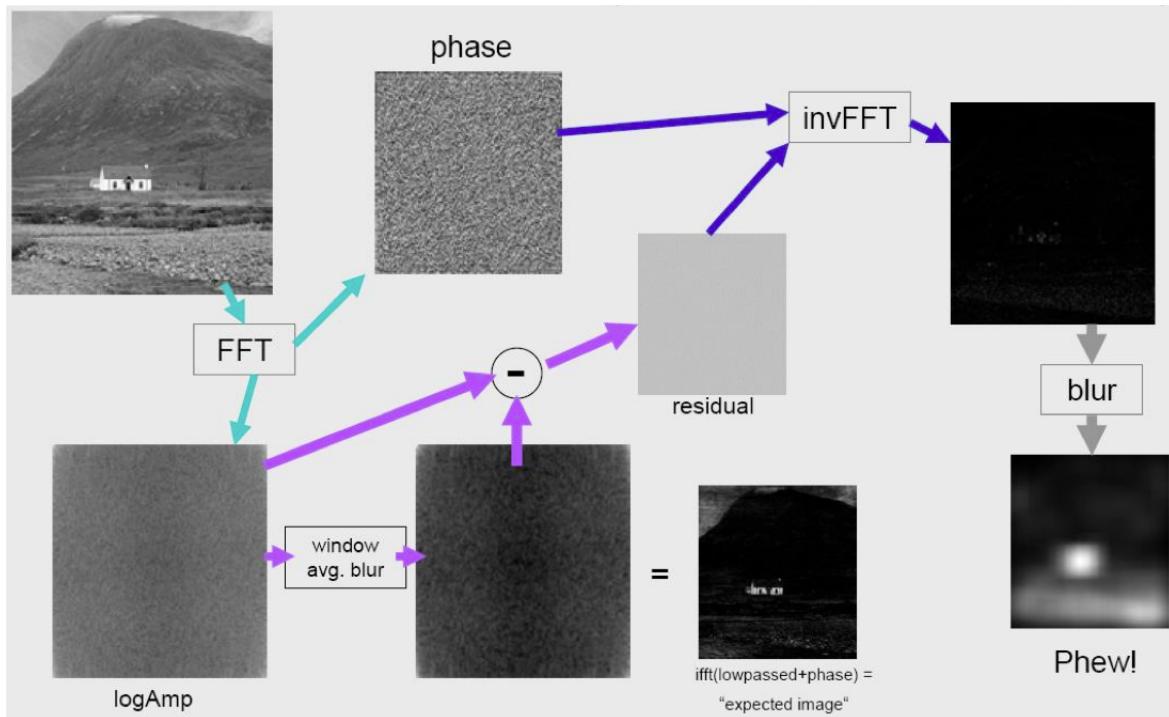
9.11 Implementierung der 2D-DFT

$$G(m, n) = \frac{1}{\sqrt{N}} \cdot \underbrace{\sum_{v=0}^{N-1} \left[\frac{1}{\sqrt{M}} \sum_{u=0}^{M-1} g(u, v) \cdot e^{-i2\pi \frac{m \cdot u}{M}} \right]}_{\text{1-dim. DFT der Zeile } g(\cdot, u)} \cdot e^{-i2\pi \frac{n \cdot v}{N}}$$

eindimensionale DFT genügt:

- zuerst alle Zeilen eines Bildes mit der DFT transformieren
- dann alle transformierten Zeilen spaltenweise mit DFT transformieren

9.12 SD Algorithmus



10 Kanten und Ecken

10.1 Übersicht

Gradientbasierte Kantendetektion

- Faltungsoperationen basierend auf der 1. Ableitung
- typische Vertreter: Prewitt, Sobel

Verfahren basierend auf der 2. Ableitung

- verbesserte Kantenlokalisierung gegenüber der Gradientverfahren
- typische Vertreter: Laplacian of Gaussian

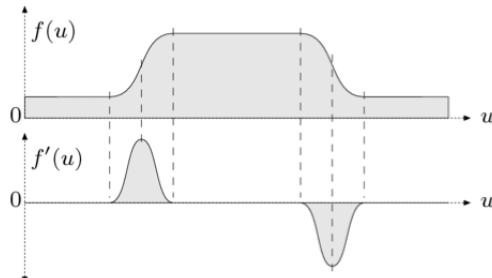
Canny Edge Detector

- basierend auf Gradientverfahren unter Ausnutzung der 2. Ableitung für Kantenlokalisierung
- gilt als eins der besten Verfahren

Morphologische Verfahren

- Ermittlung inneren oder äusseren Kontur

10.2 Gradient



Diskrete Approximation der Ableitung:

$$\frac{df}{du}(u) \approx \frac{f(u+1) - f(u-1)}{2} = 0.5 * (f(u+1) - f(u-1))$$

Partielle Ableitungen:

$$\frac{df}{du}(u, v) \quad \frac{df}{dv}(u, v)$$

Gradient:

$$\nabla I(u, v) = \begin{bmatrix} \frac{df}{du}(u, v) \\ \frac{df}{dv}(u, v) \end{bmatrix} \quad |\nabla I(u, v)| = \sqrt{\left(\frac{df}{du}(u, v)\right)^2 + \left(\frac{df}{dv}(u, v)\right)^2}$$

10.2.1 Ableitungsfilter

Die Diskrete Approximation der ersten Ableitung lässt sich einfach mit einer diskreten Faltung realisieren:

$$\frac{df}{du}(u) \approx \frac{f(u+1) - f(u-1)}{2} = 0.5 * (f(u+1) - f(u-1)) = f(u) * \left[-\frac{1}{2} \quad 0 \quad \frac{1}{2} \right]$$

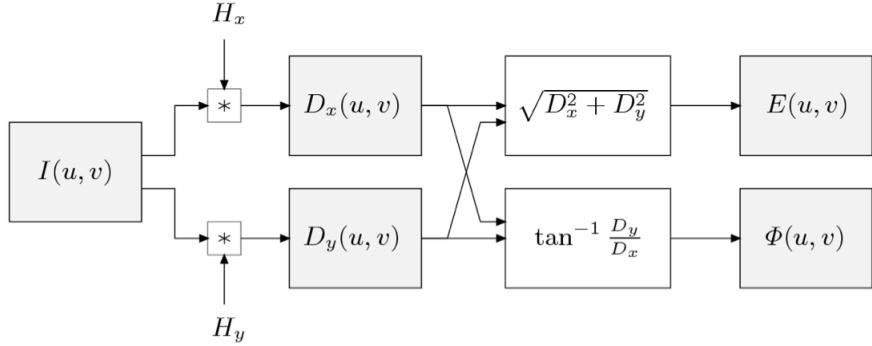
Resultierende Ableitungsfilter:

$$H_x^D = \begin{bmatrix} -0.5 & 0 & 0.5 \end{bmatrix} = 0.5 \cdot \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

$$H_y^D = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix} = 0.5 \cdot \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

10.2.2 Kantendetektierung

Mit den Gradientenfiltern H_x und H_y werden zwei Gradientenbilder D_x und D_y erzeugt. Anschliessend wird die Kantenstärke E und die Kantenrichtung Φ pro Bildposition (u, v) berechnet:

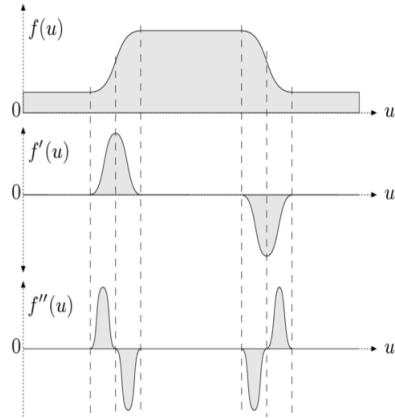


10.2.3 1. und 2. Ableitung

Problematik der Gradientenverfahren:

- detektierte Kanten sind so breit wie die Steigerungsverläufe
- schlechte Kantenlokalisierung

Mit der Verwendung der zweiten Ableitung können die Übergänge zwischen verschiedenen Steigerungsverläufen detektiert werden:



10.3 Laplacian of Gaussian

Grundidee:

- Bild zuerst mit Gauss-Filter glätten (kleine scharfe Störungen verwischen)
- Zweite Ableitung des geglätteten Bildes bestimmen
- Pder: Bild mit 2. Ableitung von Gauss falten

Gauss-Funktion:

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

1. partielle Ableitung:

$$\frac{\partial G(x, y)}{\partial x} = -\frac{x}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = -\frac{x}{\sigma^2} G(x, y)$$

2. partielle Ableitung:

$$\frac{\partial^2 G(x, y)}{\partial^2 x} = -\frac{x^2 - \sigma^2}{\sigma^4} G(x, y)$$

LoG:

$$-\nabla^2 G(x, y) = -\frac{\partial^2 G(x, y)}{\partial^2 x} - \frac{\partial^2 G(x, y)}{\partial^2 y}$$

10.3.1 Kantenschärfung

Verwendung des Laplace-Operators:

$$(\nabla^2 f)(x, y) = \frac{\partial^2 f(x, y)}{\partial^2 x} + \frac{\partial^2 f(x, y)}{\partial^2 y}$$

Approximation der 2. Ableitungen:

$$\frac{\partial^2 f}{\partial^2 x} \approx H_x^L = [1 \quad -2 \quad 1] \quad \frac{\partial^2 f}{\partial^2 y} \approx H_y^L = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

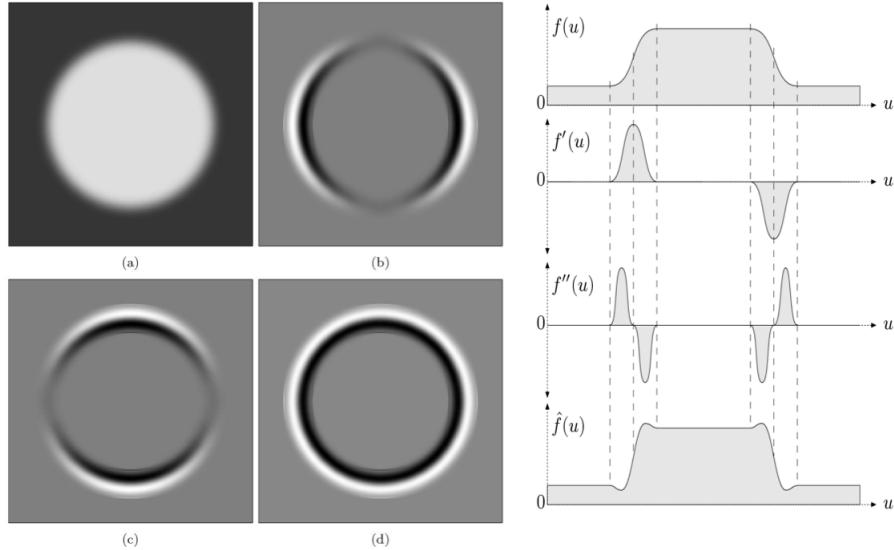
Laplace-Filter:

$$H^L = H_x^L + H_y^L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Kantenschärfung:

$$I' \leftarrow I - \omega \cdot (H^L * I)$$

10.3.2 Anwendung des Laplace-Filters



10.4 Eckpunkte

Eckpunkte weisen Intensitätsänderung in x- und y-Richtung auf.

10.4.1 Anforderungen

- lokale Einzigartigkeit
- invariant gegenüber linearen Abbildungen
- unempfindlich gegenüber Rauschen

10.4.2 Strukturmatrix

Autokorrelation:

$$cor(u, v) = \sum_{i,j \in \mathbb{R}} [I(u+i, v+j) - I(u+i+\Delta u, v+j+\Delta v)]^2$$

Taylor-Approximation des Verschobenen Bildes pro Bildpunkt berechnen:

$$A(u, v) = I_x^2(u, v) \quad B(u, v) = I_y^2(u, v) \quad c(u, v) = I_x^2(u, v) \cdot I_y^2(u, v)$$

Konzeptionell zusammengefasst zur Strukturmatrix:

$$M = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} = \begin{pmatrix} A & C \\ C & B \end{pmatrix}$$

Strukturmatrix glätten:

$$\bar{M} = \begin{pmatrix} A * H^{G,\sigma} & C * H^{G,\sigma} \\ C * H^{G,\sigma} & B * H^{G,\sigma} \end{pmatrix} = \begin{pmatrix} \bar{A} & \bar{C} \\ \bar{C} & \bar{B} \end{pmatrix}$$

10.4.3 Eigenwerte und Eigenvektoren

- Ein Vektor $x \neq 0$ heisst Eigenvektor einer quadratischen Matrix A zum reellen Eigenwert λ , falls $Ax = \lambda x$. $(A - \lambda E)x = 0$, wobei E die Einheitsmatrix ist.
- Die Eigenwerte der Matrix A sind alle reellen λ für die ein Eigenvektor existiert.
- Die Matrix A ist eine lineare Abbildung, welche den Eigenvektor x nicht rotiert, sondern nur um den Faktor λ skaliert.
- Wenn A eine Diagonalmatrix ist, dann sind die Diagonalelemente die Eigenwerte und die Einheitsvektoren sind die Eigenvektoren.

10.4.4 Ähnliche Matrizen

- 2 quadratische Matrizen A und D heissen ähnlich, falls es eine invertierbare Matrix S gibt, mit $A = S^{-1}DS$
- eine quadratische Matrix A heisst symmetrisch falls $A^T = A$
- Jede symmetrische Matrix A ist ähnlich zu einer Diagonalmatrix D . Die Diagonalelemente der Diagonalmatrix D sind die Eigenwerte von A .
- Die Transformationsmatrix S wird aus den orthonormierten Eigenvektoren x_i gebildet, $S = [x_1, x_2, \dots, x_n]$

10.4.5 Diagonalisierte Strukturmatrix

Ähnliche Matrix zur geglätteten Strukturmatrix:

- die Diagonalelemente sind gleich den Eigenwerten λ

$$\bar{M}' = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

Generelle Interpretation:

- Matrix entspricht einer linearen Abbildung basierend auf einem Koordinatensystem
- durch geschickte Variation des Koordinatensystems kann die Abbildungsmatrix vereinfacht (diagonalisiert) werden
- die Eigenvektoren entsprechen den Einheitsvektoren des neuen Koordinatensystems

10.4.6 Spezifische Interpretation

Eigenwerte:

$$\lambda_{1,2} = \frac{\text{trace}(\bar{M})}{2} \pm \sqrt{\left(\frac{\text{trace}(\bar{M})}{2}\right)^2 - \det(\bar{M})} = \frac{1}{2} \left(\bar{A} + \bar{B} \pm \sqrt{\bar{A}^2 - 2\bar{A}\bar{B} + \bar{B}^2 + 4\bar{C}^2} \right)$$

- beide sind null in flachen Bildregionen
- bei einer Kante in beliebiger Richtung ist der kleinere der beiden Eigenwerte fast null
- nur bei Eckpunkten sind beide Eigenwerte gross

Eigenvektoren:

- geben die Richtung der Kanten an

10.4.7 Corner Response Function (CRF)

Die Differenz der Eigenwerte kann als Gütemass für einen Eckpunkt gewertet werden: je kleiner die Differenz, desto eher ist es einen Eckpunkt:

$$\lambda_1 - \lambda_2 = 2 \cdot \sqrt{\frac{1}{4} \cdot (\text{trace}(\bar{M})^2 - \det(\bar{M}))}$$

CRF:

$$Q(u, v) = \det(\bar{M}) - \alpha \cdot (\text{trace}(\bar{M})^2) = (\bar{A}\bar{B} - \bar{C}^2) - \alpha \cdot (\bar{A} + \bar{B})^2$$

- mit dem Parameter α wird die Empfindlichkeit gesteuert (z.B. 0.05)
- $Q(u, v) >$ Schwellwert (z.B. 10'000 bis 1 Mio)

11 Wavelet-Transformation

11.1 Einsatz der WT

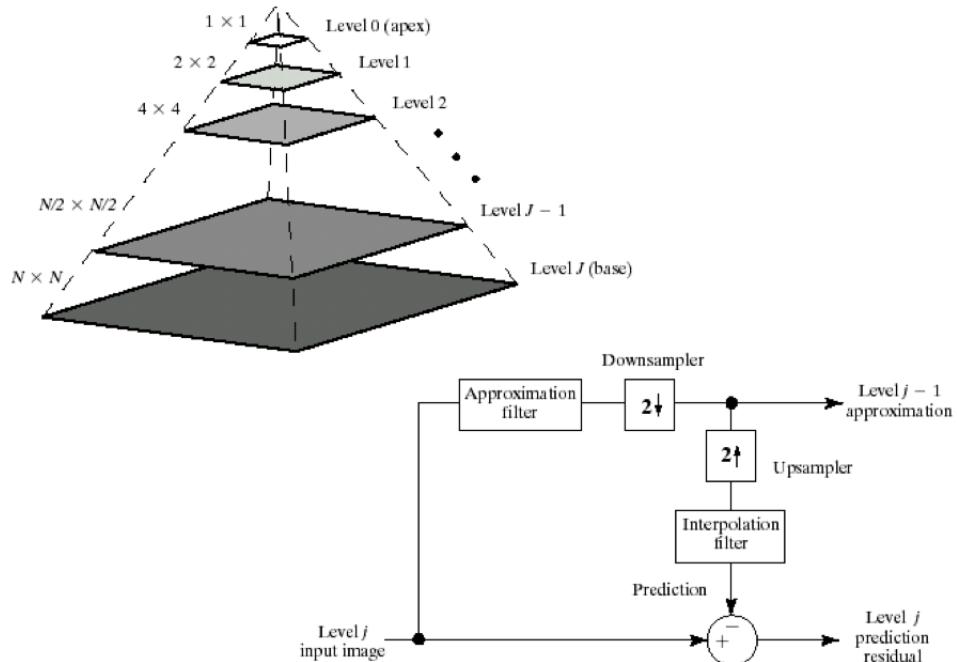
Datenverdichtung:

- Bilder: JPEG2000, PGF, MPEG-4
- Audio

Multi Resolution Models:

- Daten in verschiedenen Auflösungsstufen zur Verfügung stellen
- alle Auflösungsstufen in einem Kompakten Format zusammenhalten
- keine disjunkten Datensätze, sondern alle Daten werden benötigt, um die höchste Auflösungsstufe zu rekonstruieren

11.2 Idee der Bildpyramide



11.3 L_2 -Norm

Ist eine Vektornorm:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad |x| = \sqrt{\sum_{k=1}^n |x_k|^2}$$

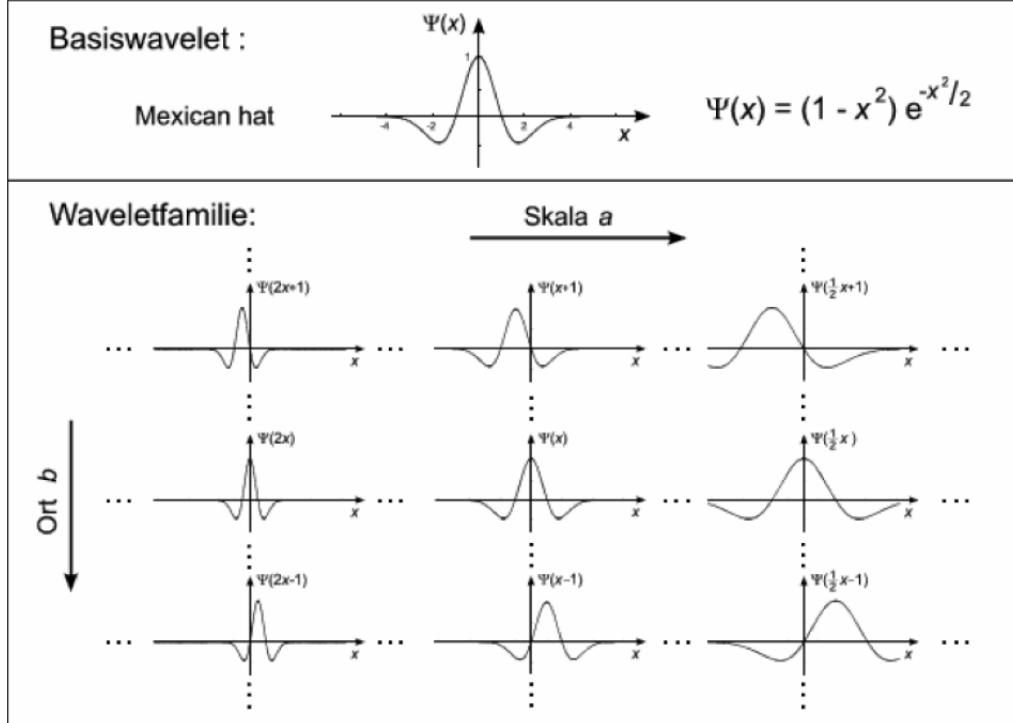
Ist eine Norm für Funktionen $\Phi(x)$:

$$\|\Phi\|^2 \equiv \Phi \circ \Phi \equiv \langle \Phi | \Phi \rangle \equiv \int |\Phi(x)|^2 dx$$

11.4 Mother Wavelet und Wavelet-Familien

Aus Mother Wavelets, auch Basis-Wavelets genannt, kann durch Skalierung ($a > 0$) und Verschiebung (b) eine ganze Wavelet-Familie erzeugt werden:

$$\Psi_{a,b}(x) = \frac{1}{\sqrt{a}} \Psi\left(\frac{x-b}{a}\right)$$



11.5 Continuous Wavelet Transform

- Ausgangssignal $f(x)$
- wähle beliebiges Wavelet Ψ
- berechne Skalarprodukt von $\Psi_{a,b}$ mit $f(x)$ für alle möglichen Paare von a und b
- die berechneten Werte sind die Wavelet-Koeffizienten $CWT_{a,b}$

Vorwärtstransformation

$$CWT_{a,b}(f) = \langle f(x) | \Psi_{a,b} \rangle \equiv \int_{-\infty}^{\infty} f(x) \overline{\Psi_{a,b}(x)} dx$$

Rücktransformation

$$f(x) = \langle CWT_{a,b}(f) | \Psi_{a,b} \rangle \equiv \int_{\mathbb{R}^* \times \mathbb{R}} CWT_{a,b}(f) \overline{\Psi_{a,b}(x)} \frac{da db}{|a|^2}$$

11.6 CWT, DWT und FWT

Problem der CWT:

- es gibt unendliche viele Parameterpaare (a, b)

- es ist unklar, welche Parameterpaare (a,b) erforderlich sind, um eine vollständige Rekonstruktion zu ermöglichen

Lösungsansatz DWT:

- üblicherweise sind diskrete Signale mit begrenzter Abtastung gegeben
- für diskrete Signale muss eine endliche Zahl von Parameterpaaren (a,b) ausreichen, so dass eine Rücktransformation ohne Verlust möglich ist
→ Discrete Wavelet Transform (DWT)

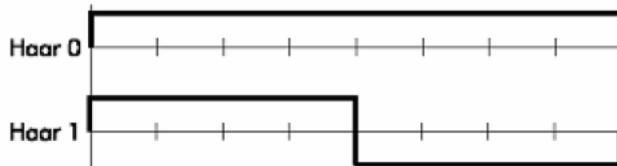
Durchbruch: FWT:

- Multiresolution Analysis führte zum Algorithmus der schnellen DWT → Fast Wavelet Transform (FWT)

11.7 FWT

Input	A	B	C	D	E	F	G	H
Detail 1	$A/2 - B/2$	$B/2 - A/2$	$C/2 - D/2$	$D/2 - C/2$	$E/2 - F/2$	$F/2 - E/2$	$G/2 - H/2$	$H/2 - G/2$
Durchschnitt 1	$AB := A/2 + B/2$		$CD := C/2 + D/2$		$EF := E/2 + F/2$		$GH := G/2 + H/2$	
Detail 2	$AB/2 - CD/2$		$CD/2 - AB/2$		$EF/2 - GH/2$		$GH/2 - EF/2$	
Durchschnitt 2	$ABCD := AB/2 + CD/2$				$EFGH := EF/2 + GH/2$			
Detail 3	$ABCD/2 - EFGH/2$				$EFGH/2 - ABCD/2$			
Durchschnitt 3					$ABCD/2 + EFGH/2$			
Rekonstruktion								
Input	12	4	6	8	4	2	5	7
Detail 1	4	-4	-1	1	1	-1	-1	1
Durchschnitt 1	8		7		3		6	
Detail 2	0.5		-0.5		-1.5		1.5	
Durchschnitt 2			7.5				4.5	
Detail 3			1.5				-1.5	
Durchschnitt 3					6			
Rekonstruktion	12	4	6	8	4	2	5	7

11.8 FWT als Filteroperation



- **Betrachtungsweise**
 - Haar0 und Haar1 skaliert auf ein Intervall der Länge 2
 - Amplitude von Haar0 wird zu $\frac{1}{2}$ (nicht normalisiert)
 - Amplituden von Haar1 werden zu $\frac{1}{2}$ und $-\frac{1}{2}$ (nicht normalisiert)
- **Schreibweise als Filterkoeffizienten**
 - Haar0 = [$\frac{1}{2}$, $\frac{1}{2}$] =: low (Skalierungsfunktion, Tiefpass)
 - Haar1 = [$\frac{1}{2}$, $-\frac{1}{2}$] =: high (Waveletfunktion, Hochpass)
- **Anwendung**
 1. Ausgangssignal jeweils mit low und high falten
 2. Schritt 1 mit dem zuvor skalierten (low) Signal wiederholen

11.9 Rücktransformation

Generelle Rekonstruktion des Ausgangssignals:

- im Wesentlichen wieder ein Skalarprodukt mit einem Wavelet
- Analyse- und Synthese-Wavelet müssen nicht identisch sein

Bei der FWT:

- Verwendung eines Synthese-Filterpaars, passend zum Analyse-Filterpaar und Anwendung der Faltung

11.10 FWT Filterbank

Bi-orthogonale Filterbank:

- (h_L, h_H) : Filterpaar der Vorwärtstransformation (Low, High)
- (g_L, g_R) : Filterpaar der Rückwärtstransformation (Left, Right)
- Rücktransformationsfilter können aus der Vorwärtstransformationsfilter abgeleitet werden:
 $\langle h_L | g_R \rangle = 0$ und $\langle h_H | g_L \rangle = 0$

Beispiel

– Daubechies 5/3, bi-orthogonal [PGF, JPEG2000]

$$h_L = \frac{1}{4\sqrt{2}} [-1, 2, \bar{6}, 2, -1] \quad h_H = \frac{1}{2\sqrt{2}} [-\bar{1}, 2, -1]$$

$$g_L = \frac{1}{2\sqrt{2}} [1, \bar{2}, 1] \quad g_R = \frac{1}{4\sqrt{2}} [-1, \bar{-2}, 6, -2, -1]$$

		Transformationsmatrizen für Signalvektor der Länge 8
Vorwärtstransformation	$\begin{bmatrix} 6 & 4 & -2 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & 6 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & 6 & 2 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 & 2 & 5 \\ 8 & -2 & 4 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 4 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2 & 4 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -4 & 4 \end{bmatrix}$	Rücktransformation
		$\begin{bmatrix} 4 & 0 & 0 & 0 & -4 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 5 & -1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 2 & -2 & 0 & 0 \\ 1 & 0 & 2 & 2 & 0 & -1 & 6 & -1 \\ 4 & 0 & 0 & 4 & 0 & 0 & -2 & -2 \\ 0 & 0 & 2 & 2 & 0 & -1 & 6 & -1 \\ 0 & 0 & 0 & 4 & 0 & 0 & -2 & -2 \\ 0 & 0 & 0 & 4 & 0 & 0 & -2 & 6 \end{bmatrix}$

11.11 FWT in zwei Dimensionen

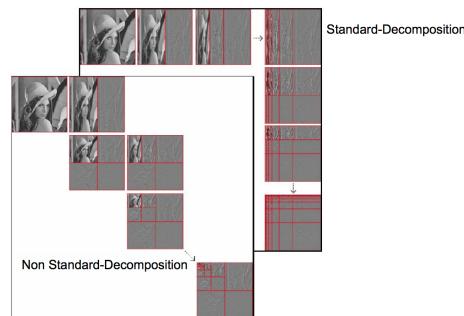
Die Grundidee ist wie bei der DFT wird für mehrere Dimensionen auf die eindimensionale Transformation zurückgegriffen. Die Filterpaare müssen auch normalisiert sein (L_2 -Norm = 1)

Standard Dekomposition:

- mehrstufige FWT für jede Zeile des Bildes
- mehrstufige FWT für jede Spalte des zeilentransformierten Bildes

Nicht-Standard Dekomposition:

- 1 Filterschritt für jede Zeile eines Bildes durchführen
- 1 Filterschritt für jede Spalte des zeilentranformierten Bildes
 → vier Quadranten entstehen: LL, HL, LH, HH
- rekursiv den Quadranten LL wieder filtern



11.12 Bilddaten-Dekorrelation

Innerhalb eines Farbkanals gibt es noch weitere Redundanz (z.B. Flächen mit gleicher Farbe). Um die Redundanz zu reduzieren findet eine Transformation vom Bild- in den Frequenzraum statt.

DCT (discrete cosine transform):

- schnell, einfach
- Blockbildung
- Einsatz in JPEG und MPEG

DWT (discrete wavelet transform):

- schnell, einfach
- keine Blockbildung
- verlustlose Integer-Arithmetik möglich
- Einsatz in PGF, JPEG 2000, MPEG-4

11.13 Quantisierung

Ziel:

- möglichst kleine Anzahl von unterschiedlichen Wavelet-Koeffizienten
- Koeffizienten mit kleinen Absolutwerten (viele Nullen)

Ansatz:

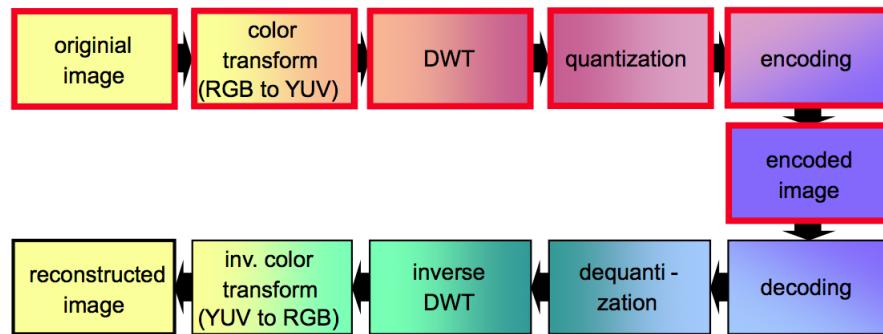
- Zusammenfassen von ähnlich grossen Wavelet- Koeffizienten in Gruppen → Quantisierung
- anstatt des Wertes eines Wavelet-Koeffizienten wird seine Gruppennummer abgespeichert
- für jede Gruppe muss die Spannweite abgespeichert werden → Quantisierungstabelle
- je grösser die Spannweite einer Gruppe, desto grösser der Informationsverlust → ohne Quantisierung kein Informationsverlust

12 PGF

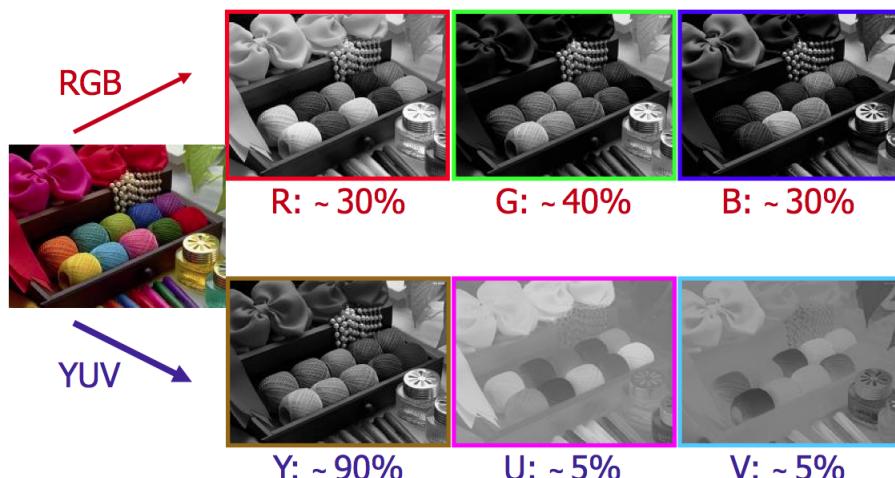
12.1 Features

- sehr hohe verlustlose Kompressionsrate
- sehr gute verlustbehaftete Bildqualität/Kompressionsrate
- kontinuierlicher Datenstrom erlaubt schrittweise Erhöhung der Auflösung

12.2 Aufbau



12.3 Farbtransformation



12.4 5/3 Wavelet Filterbank

- kann ganzzahlig implementiert werden, obwohl Faktor $\sqrt{2}$ vorkommt
- ist relativ kurz (kleine Anzahl von Filterkoeffizienten)
- gehört zu den besten Filtern für Bildverarbeitung

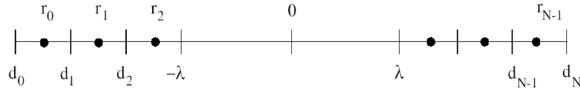
12.5 Skalare Quantisierung

Grundschema

- uniform: die Spannweiten aller Gruppen sind gleich lang (Zweiertpotenzen)
→ keine Quantisierungstabelle notwendig
- sehr einfach, alle Koeffizienten durch die Spannweite dividieren

dead zone

- größeres Intervall um Null herum, in welchem alle Koeffizienten auf Null gesetzt werden
- die meisten Wavelet-Koeffizienten sind um Null herum
→ sehr grosse Gruppe mit wenig Informationsverlust



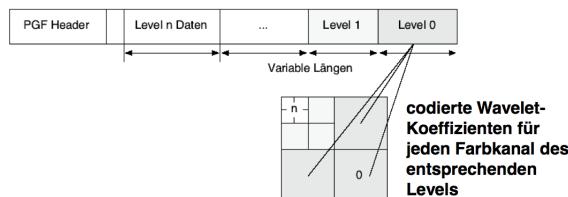
12.6 Progressive Kodierung

Ziele:

- eingebunder Bitstrom fürs sequentielles Lesen
- schnelles, progressives Kodierungsschema
- gute Komprimierungsrate

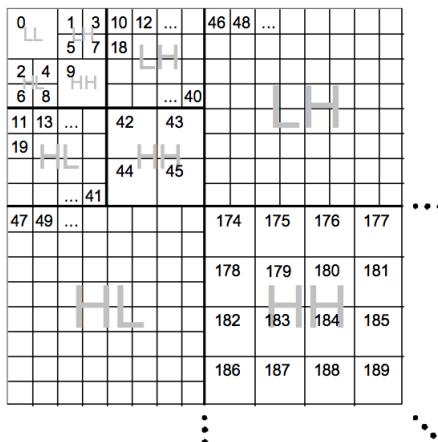
Ansatz:

- Wavelet-Koeffizienten getrennt nach Level abspeichern
- Umordnen der Wavelet-Koeffizienten, so dass Koeffizienten mit ähnlich grossen Absolutwert nahe beieinander liegen



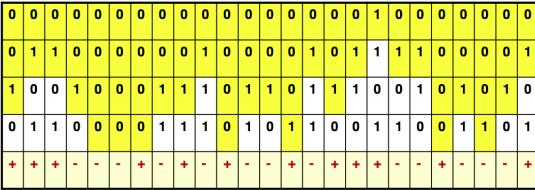
Umordnen der Koeffizienten:

Koeffizienten in den zusammengehörigen LH- und HL-Subbands an der gleichen Bildposition ähnlich gross



12.7 Bitplane-Kodierung

- **Verfahren**
 - umgeordnete, quantisierte Absolutwerte der Koeffizienten werden in Makroblöcke gleicher Länge aufgeteilt
 - pro Makroblock Bitplane-Kodierung anwenden
→ Aufteilung in signifikante Bits und Verfeinerungsbits
 - signifikante Bits enthalten sehr lange Folgen von Nullen und werden deshalb adaptiv lauflängenkodiert

MSB	
LSB	

12.8 Lauflängenkodierung (RLE)

- **Annahme**
lange Sequenzen von Nullen
- **statisches Verfahren**

Codewort	Inputsequenz
0	2^k Nullen
1n0	0 < n < 2^k Nullen gefolgt von einer 1, positives Vorzeichen an der Stelle i
1n1	0 < n < 2^k Nullen gefolgt von einer 1, negatives Vorzeichen an der Stelle i

n wird mit k Bits abgespeichert

12.9 Adaptives RLE

Problem:

- der bestmögliche Wert für k hängt vom Input ab
- wird k zu gross gewählt, so werden zu viele Bits verschwendet, um n abzuspeichern
- wird k zu klein gewählt, so werden zu viele Codeworte erzeugt

Ansatz (adaptiv):

- k nach jedem Codewort anpassen
- k mit 0 initialisieren
- nach einem Codewort 0: $k + +$
- nach einem anderem Codewort: $k - -(k > 0)$