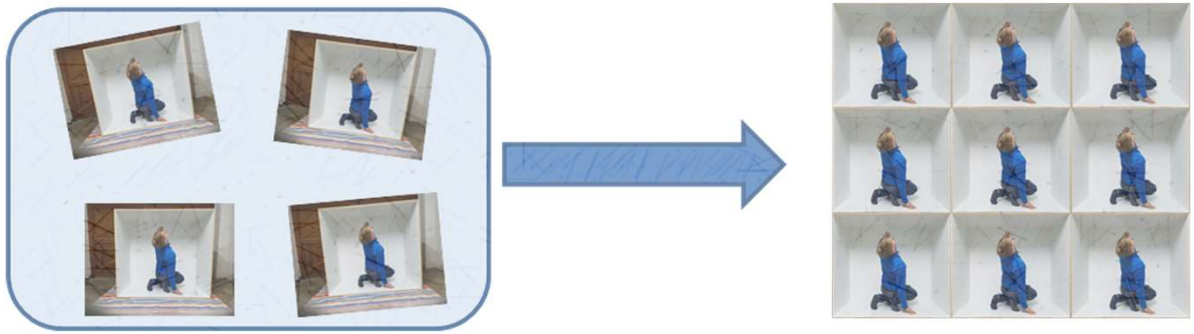
Digitale Bildverarbeitung 1

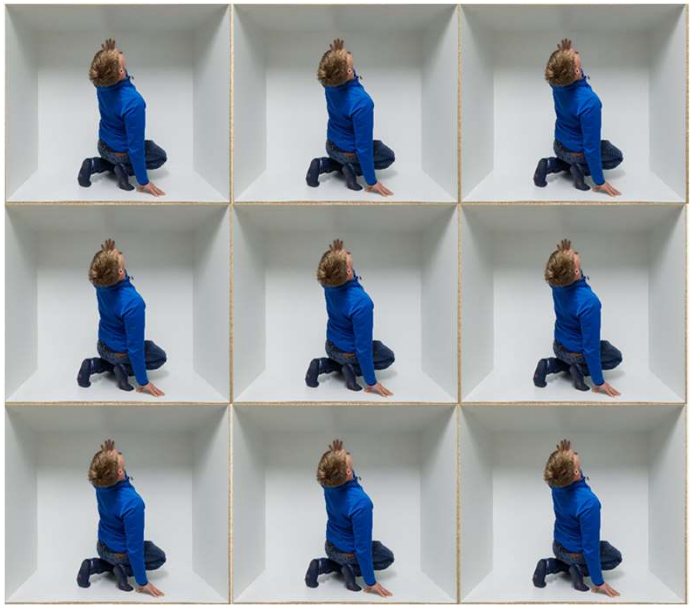
Einführung in die digitale Bilderverarbeitung
für Informatikstudierende im Bachelor

Vorlesung: Michael Möller – michael.moeller@uni-siegen.de
Übungen: Hannah Dröge – hannah.droege@uni-siegen.de



Digitale Bilder

Ziel der Vorlesung: Wie kann ich automatisch einen Setzkasten zusammenbauen?



- Was ist ein Bild?
- Wie wird die 3d Geometrie der Welt im Bild dargestellt?
- Wie wende ich eine Transfomation auf mein Bild an? (Interpolation)
- Bild schärfen, entrauschen oder Kannten erkennen? (Filter, Fourier Transformation, DCT)
- Wie werden Farben aufgenommen und reproduziert?
- Wie kann man Bilder automatisch Clustern und Segmentieren? (thresholding, k-means, morphologische Op.)

>> Setzkasten (mit Python) zusammenbauen

Digitale Bilder

Was sind digitale Bilder?

Diskret: Tensor (3 Matrizen)

$$f \in \mathbb{R}^{3 \times n_x \times n_y}$$

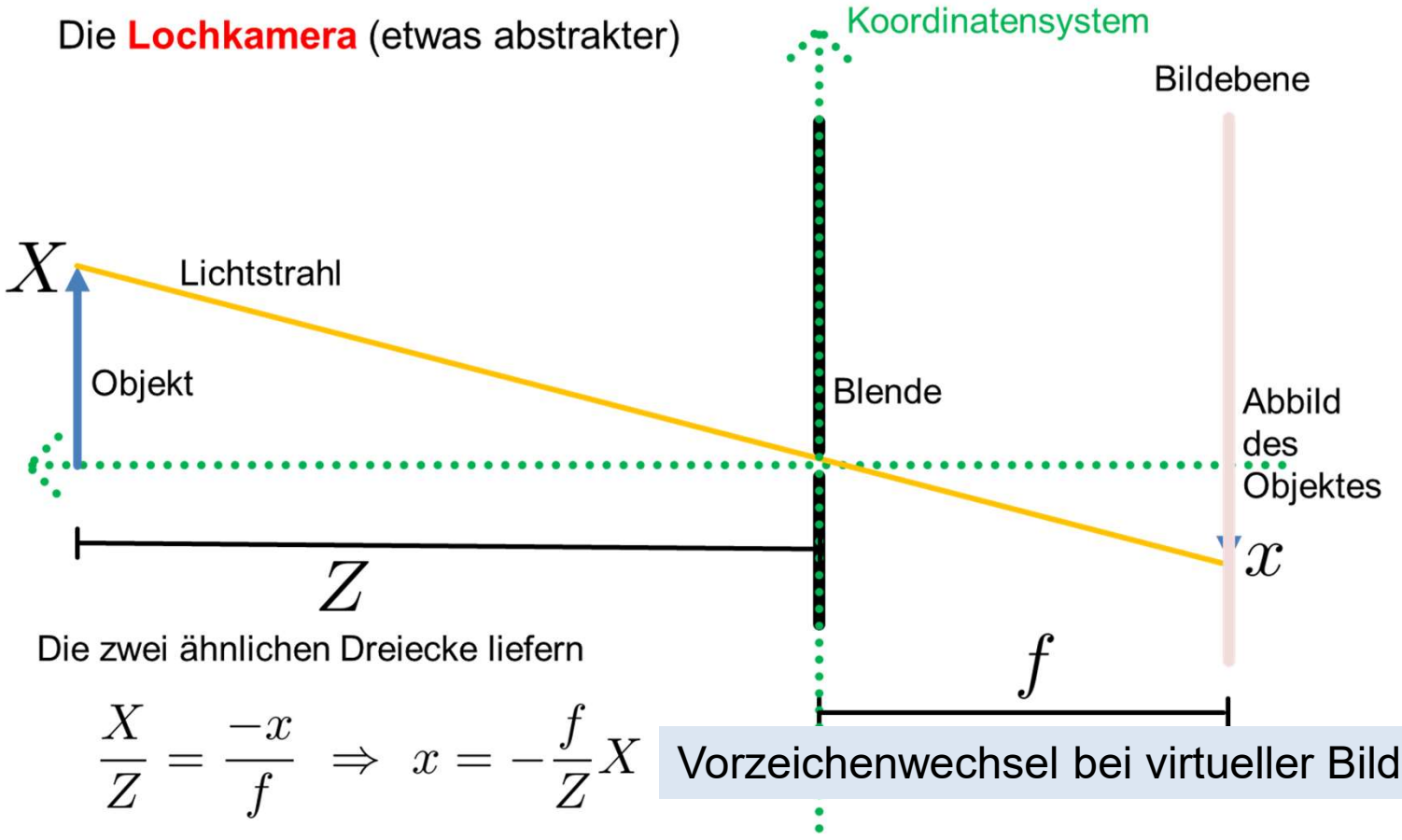
Kontinuierlich

$$f : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

Bildenstehung

Wieso ist das Rechteckt kein Rechteck?

Die **Lochkamera** (etwas abstrakter)

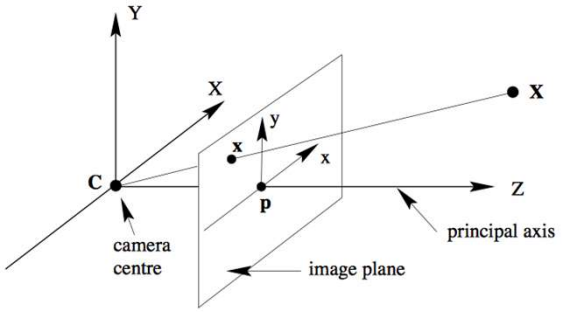


Die zwei ähnlichen Dreiecke liefern

$$\frac{X}{Z} = \frac{-x}{f} \Rightarrow x = -\frac{f}{Z}X$$

Vorzeichenwechsel bei virtueller Bildebene!

Bildgeometrie



Entlang jeder Koordinate ist die Geometrie exakt die gleiche wie im zuvor betrachteten Fall!

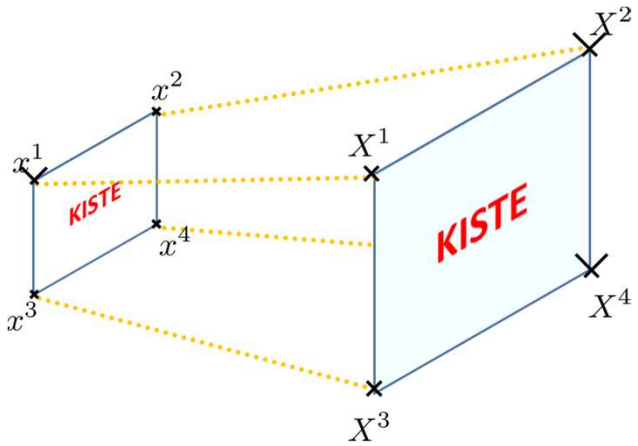
$$x_1 = \frac{f}{Z} X_1 \qquad x_2 = \frac{f}{Z} X_2$$

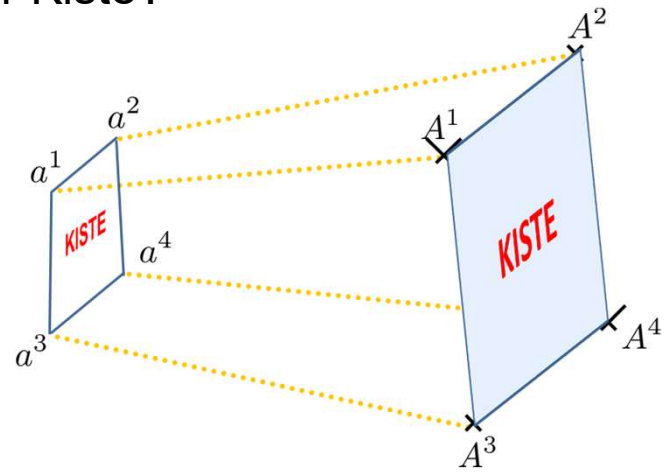
Der Einfachheit halber nehmen wir $f = 1$ an. Gegeben $\boldsymbol{x}$ heißt dies es gibt ein λ mit

$$\lambda \boldsymbol{x} = X$$

Homogene Koordinaten

Aber wie hilft uns dies für das Begradigen der Kiste?





Bildgeometrie

Wir stellen uns vor die Kamera ist fix. Dann entstehen die Koordinaten A^i durch Rotation und Translation der gewünschten Koordinaten X^i :

$$X^i = RA^i + \vec{t} \quad \text{für } i \in \{1, 2, 3, 4\}$$

Die Punkte A^i und X^i liegen jeweils in einer Ebene im Raum.

Rechnung zeigt: Es gibt eine Matrix H, sodass für jeden Punkt folgende Gleichung gilt:

$$\boldsymbol{x}^i = \frac{\lambda_a^i}{\lambda_x^i} H \boldsymbol{a}^i$$

Wir können wir dies nutzen um eine Gleichung für H zu bekommen?

$$0 = B^i \vec{H} \qquad \text{mit} \qquad B^i = \begin{pmatrix} \boldsymbol{a}_1^i \hat{\boldsymbol{x}}^i & \boldsymbol{a}_2^i \hat{\boldsymbol{x}}^i & \boldsymbol{a}_3^i \hat{\boldsymbol{x}}^i \end{pmatrix}$$

Bildgeometrie

Man kann aus den Koordinaten eine Matrix B konstruieren, sodass $0 = B\vec{H}$

Lösung für $\vec{H}$: Eigenvektor zum Eigenwert 0 von $B^T B$

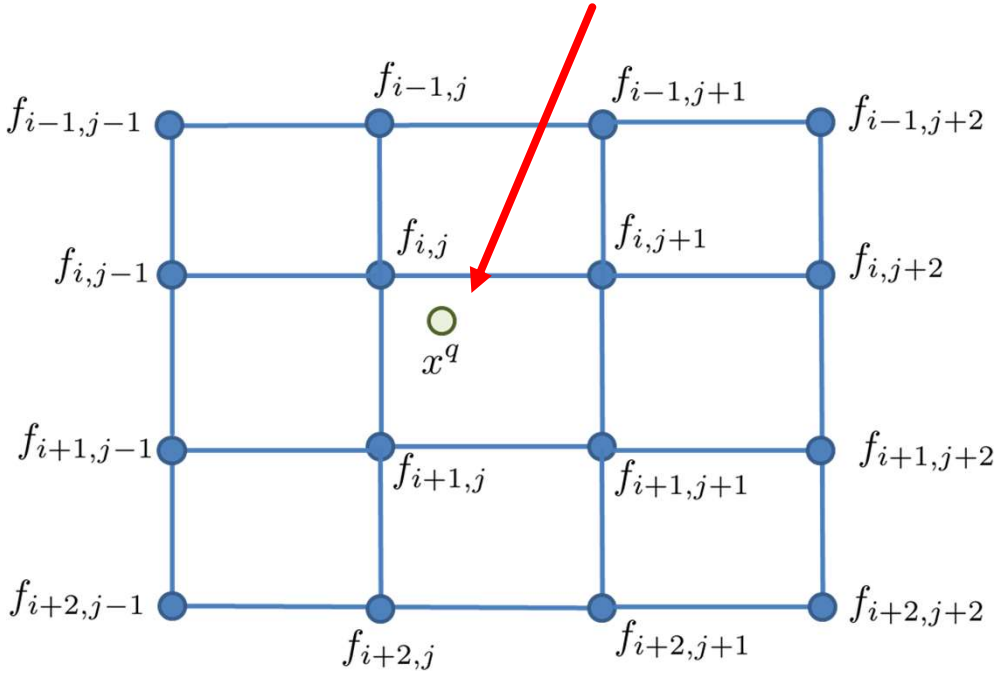
Projektive Transformation

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \frac{1}{H_{31}a_1 + H_{32}a_2 + H_{33}} \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ 1 \end{pmatrix} \quad \text{oder} \quad \boldsymbol{x} = \frac{1}{(H\boldsymbol{a})_3} H\boldsymbol{a}$$

Die Matrix H kann aus 4 Punktkorrespondenzen durch Lösung eines Eigenwertproblems bestimmt werden.

Interpolation

Was ist eine gute Schätzung für den Wert eines Bildpunktes an dieser Stelle?



Kennengelernt:

- Bilineare Interpolation
- Bikubische Interpolation

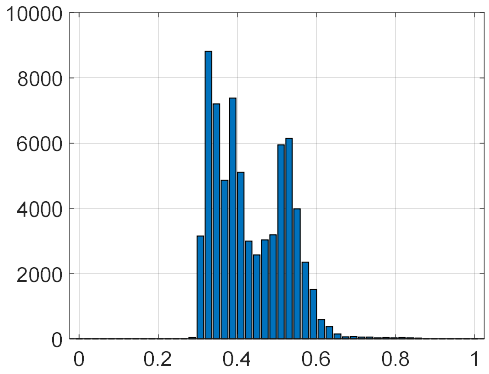
Prinzip: Zwei eindimensionale Probleme.

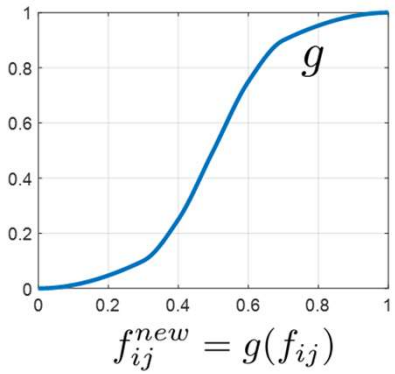
Erst in x-Richtung, dann in y-Richtung

(oder umgekehrt)

Filter

Punktweise





Lokal, linear: Korrelations- oder Faltungsfilter

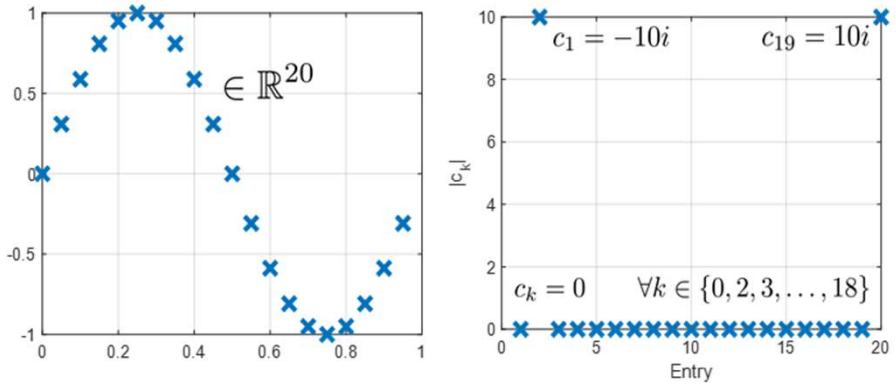
$$g(f_{i,j}) = \sum \left(\begin{bmatrix} f_{i-1,j-1} & f_{i-1,j} & f_{i-1,j+1} \\ f_{i,j-1} & f_{i,j} & f_{i,j+1} \\ f_{i+1,j-1} & f_{i+1,j} & f_{i+1,j+1} \end{bmatrix} \underset{\text{Punktw. Multi.}}{\bullet} \begin{bmatrix} k_{i-1,j-1} & k_{i-1,j} & k_{i-1,j+1} \\ k_{i,j-1} & k_{i,j} & k_{i,j+1} \\ k_{i+1,j-1} & k_{i+1,j} & k_{i+1,j+1} \end{bmatrix} \right)$$

Fourier Transformation

Fouriertrafo: Koeffizientenvektor $\vec{c} \in \mathbb{C}^n$ zur Darstellung eines Signals $\vec{f} \in \mathbb{C}^n$ als Linearkombination von $\frac{1}{n}\vec{z}^{\,k} \in \mathbb{C}^n$ mit $z_l^k = \exp\left(2\pi i \frac{lk}{n}\right)$

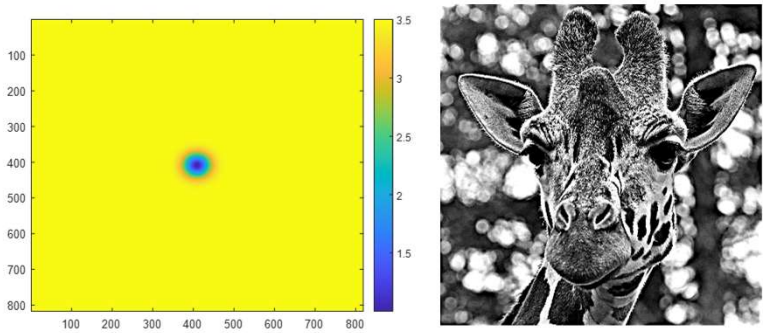
$$c_k = \sum_{l=0}^{n-1} f_l \exp\left(-2\pi i \frac{lk}{n}\right)$$ **Fourier-transformation**

$$\vec{f} = \frac{1}{n} \sum_{k=0}^{n-1} c_k \vec{z}^{\,k}$$ **Inverse Fouriertransformation**



Faltungssatz

$$f * g = \mathcal{F}^{-1}\left(\mathcal{F}(f) \odot \mathcal{F}(g)\right)$$



Für Bilder: Entlang jeder Dimension einzeln!

Cosinus Transformation

Cosinustrafo: Koeffizientenvektor $\vec{c} \in \mathbb{R}^n$ zur Darstellung eines Signals $\vec{f} \in \mathbb{R}^n$ als Linearkombination von $\vec{z}^{\,k} \in \mathbb{R}^n$ mit $z_l^0 = \frac{1}{\sqrt{n}}$ und $z_l^k = \sqrt{\frac{2}{n}} \cos(\pi k x_l)$

für $x_l = \frac{(l+\frac{1}{2})}{n}, \quad l \in \{0, 1, \ldots, n-1\}$

$$c_k = \langle f, z^k \rangle = \begin{cases} \frac{1}{\sqrt{n}} \sum_{l=0}^{n-1} f_l & \text{falls } k = 0 \\ \sqrt{\frac{2}{n}} \sum_{l=0}^{n-1} f_l \cos\left(\pi k \frac{l+\frac{1}{2}}{n}\right) & \text{sonst.} \end{cases}$$

DCT

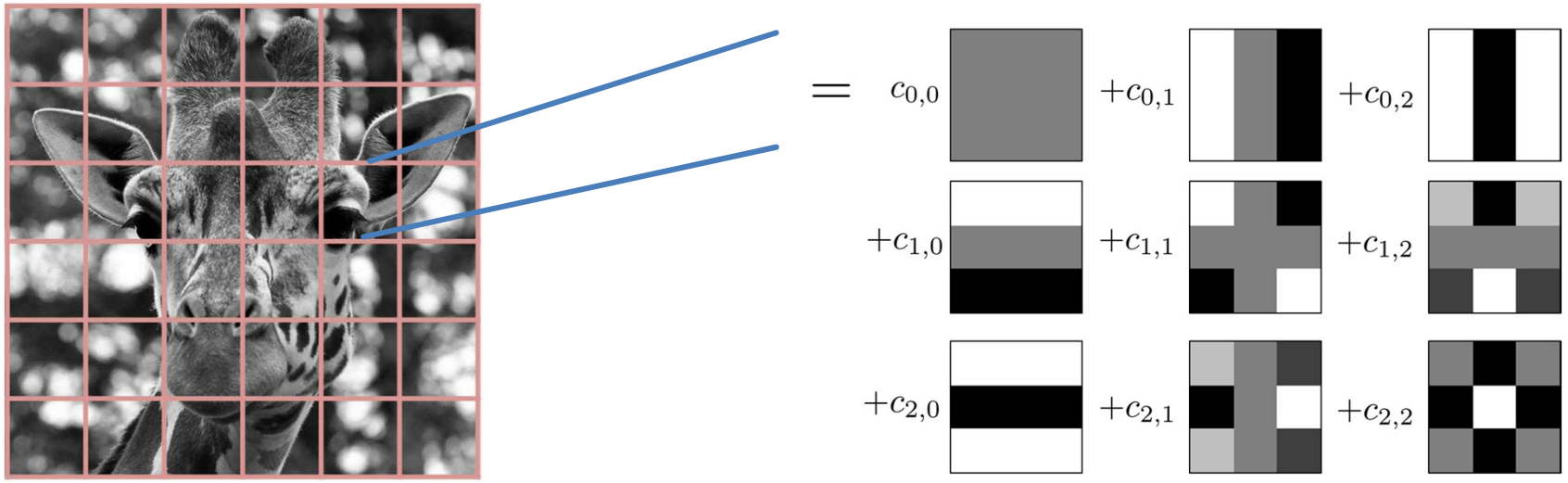
$$f_l = \frac{1}{\sqrt{n}} c_0 + \sqrt{\frac{2}{n}} \sum_{k=1}^{n-1} c_k \cos\left(\pi k \frac{l+\frac{1}{2}}{n}\right)$$

Inverse DCT

Für Bilder: Entlang jeder Dimension einzeln!

Cosinus Transformation

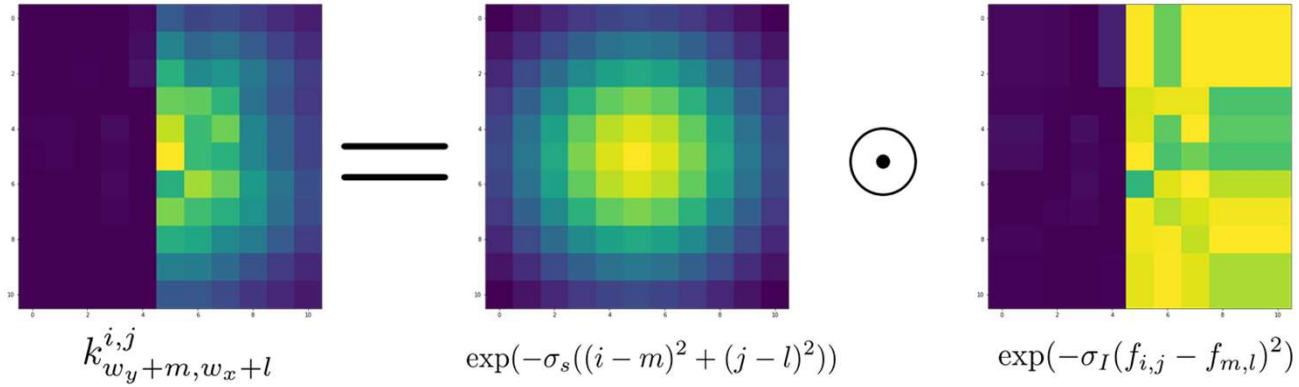
Anwendung für Bilder, z.B. Kompression (JPG) oder auch Entrauschen, durch Darstellung kleiner Patches mittels DCT und Manipulation der Koeffizienten

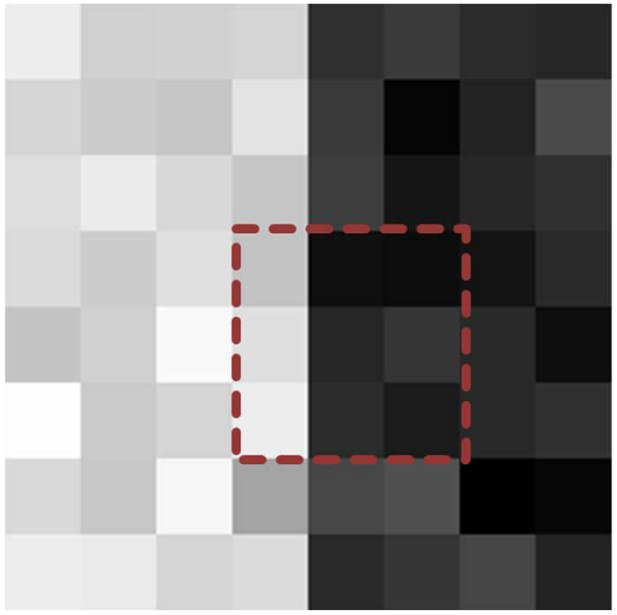


Thresholden führt zu Entrauschen

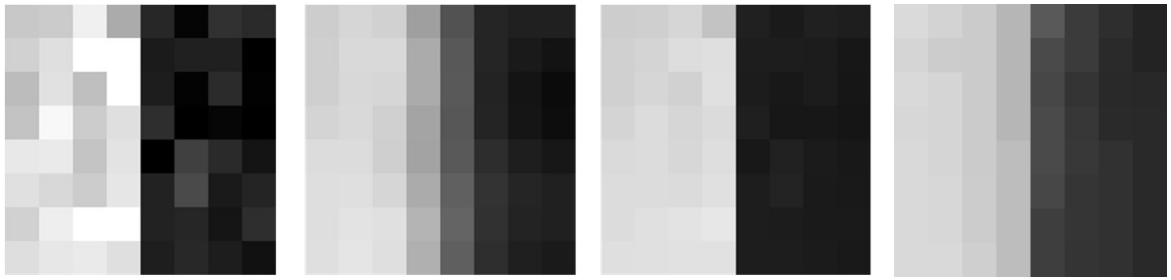
Nichtlineare Filter

Bilateralfiler: Gaußfilter, der lokal basierend auf der Ähnlichkeit jedes Pixels zu seinen Nachbarn gewichtet wird.

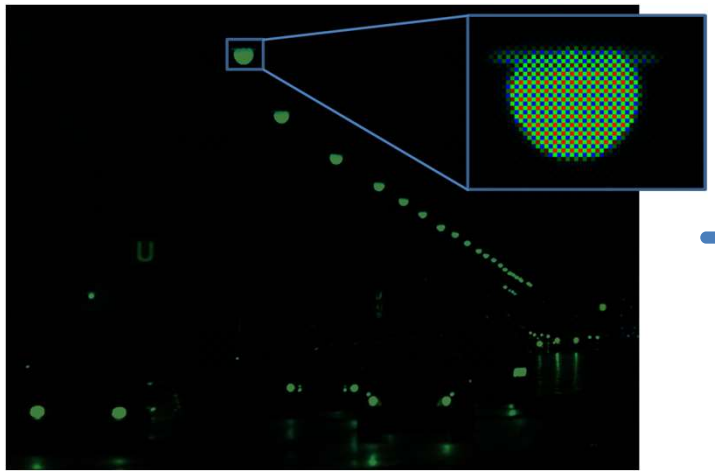




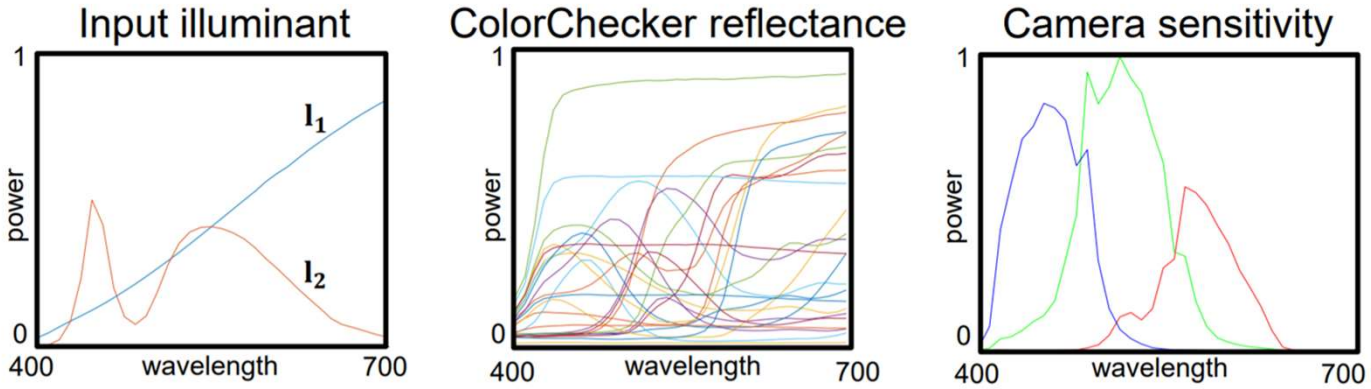
Medianfilter: Ersetze den aktuellen Pixelwert durch den Median aller Werte in einer Umgebung vordefinierter Größe um den aktuellen Pixel



Farben



Prozesskette



From: Karaimer and Brown, "Improving Color Reproduction Accuracy on Cameras", 2018

$$X \propto \int I_{illu}(\lambda)\ I_{reflect}(\lambda)\ I^{R}_{cam}(\lambda)\ d\lambda \qquad Y \propto \int I_{illu}(\lambda)\ I_{reflect}(\lambda)\ I^{G}_{cam}(\lambda)\ d\lambda$$

$$Z \propto \int I_{illu}(\lambda)\ I_{reflect}(\lambda)\ I^{B}_{cam}(\lambda)\ d\lambda$$

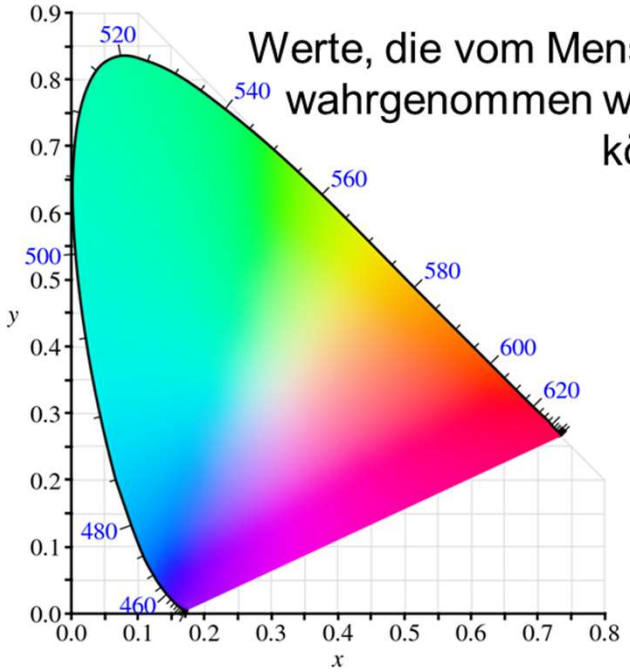
Farben

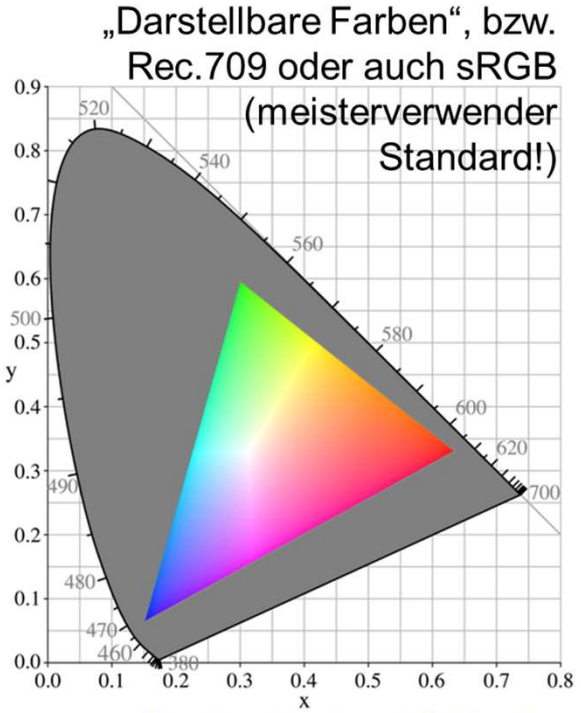
Intensität „herausdividieren“:

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

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

$$z = \frac{Z}{X+Y+Z}$$



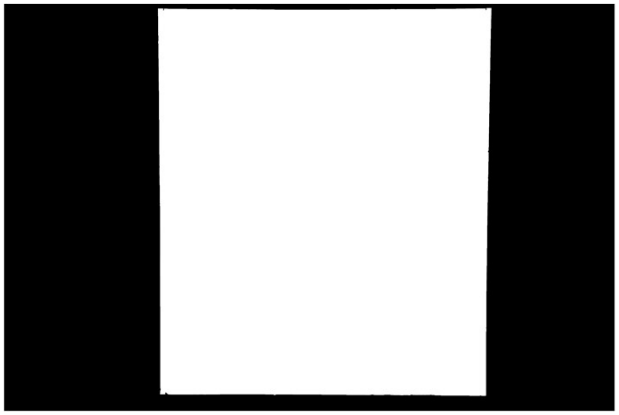


From https://en.wikipedia.org/wiki/Gamut

Für Algorithmen kann man oft besser in anderen Farbräumen als RGB arbeiten!

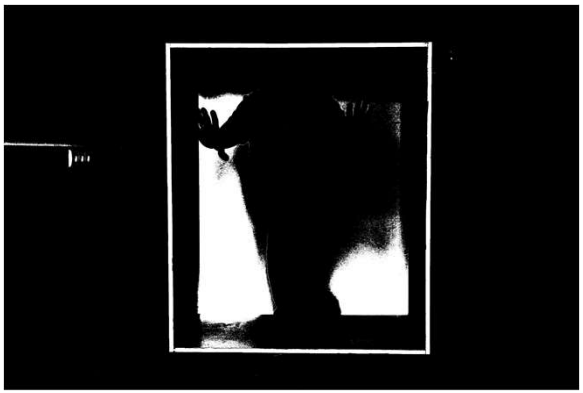
$$\begin{pmatrix} v \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3} \\ 1/\sqrt{2} & 0 & -1/\sqrt{2} \\ 1/\sqrt{6} & -2/\sqrt{6} & 1/\sqrt{6} \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

Segmentierung


Thresholding:

$$m_{i,j} = \begin{cases} 1 & \text{falls } f_{i,j} > t \\ 0 & \text{sonst.} \end{cases}$$



Wobei t der Threshold ist

und f ein Bild welches aus dem Original durch eine geeignete Transformation gewonnen wurde, z.B. Grauwertbild, Farbwinkel, etc.

Segmentierung

Dilatation

Formale Definition:

Für $G \subset \mathbb{Z}^2$ und $S \subset \mathbb{Z}^2$ ist die Dilatation von G mit S gegeben durch

$$G \oplus S := \{x \mid G \cap S_x \neq \emptyset\}$$

wobei

$$S_x := \{x + y \mid y \in S\}$$

die Verschiebung von S nach x beschreibt.

Weiß (=1) genau dann wenn Strukturelement zentriert am aktuellen Pixel noch irgendwo auf einen anderen weißen Pixel trifft

Erosion

Formale Definition:

Für $G \subset \mathbb{Z}^2$ und $S \subset \mathbb{Z}^2$ ist die Erosion von G mit S gegeben durch

$$G \ominus S := \{x \mid S_x \subset G\}$$

wobei

$$S_x := \{x + y \mid y \in S\}$$

die Verschiebung von S nach x beschreibt.

Weiß (=1) genau dann wenn Strukturelement zentriert am aktuellen Pixel überall auf weiße Pixel trifft

Erosion der Dilatation: **morphological Closing**

Dilatation der Erosion: **morphological Opening**

- Füllen von Löchern in der Segmentierung
- Auswählen der größten Wegzusammenhangskomponente

Clustering

Automatische Aufteilung in Regionen und Clustercentern

Schritt 1: Gegeben $c^l \in \mathbb{R}^3,\ l \in \{1, \ldots, k\}$ bestimme die Segmentierung $m \in \mathbb{R}^{n_y \times n_x \times k}$ des Bildes $f \in \mathbb{R}^{n_y \times n_x \times 3}$ mittels

$$m_{i,j,:} = e_l \quad \text{falls} \quad \|f_{i,j,:} - c^l\| \leq \|f_{i,j,:} - c^t\| \quad \forall t \in \{1, \ldots, k\}$$

e_l: l-te Einheitsvektor

Schritt 2: Gegeben eine Segmentierung $m \in \mathbb{R}^{n_y \times n_x \times k}$ des Bildes $f \in \mathbb{R}^{n_y \times n_x \times 3}$

Bestimme geeignete Farben für jedes Segment, indem die Durchschnittsfarben vom jeweiligen Segment gebildet wird.

$$c^l = \frac{1}{\sum_{i=0}^{n_y-1} \sum_{j=0}^{n_x-1} m_{i,j,l}} \sum_{i=0}^{n_y-1} \sum_{j=0}^{n_x-1} f_{i,j,:}\ m_{i,j,l}$$



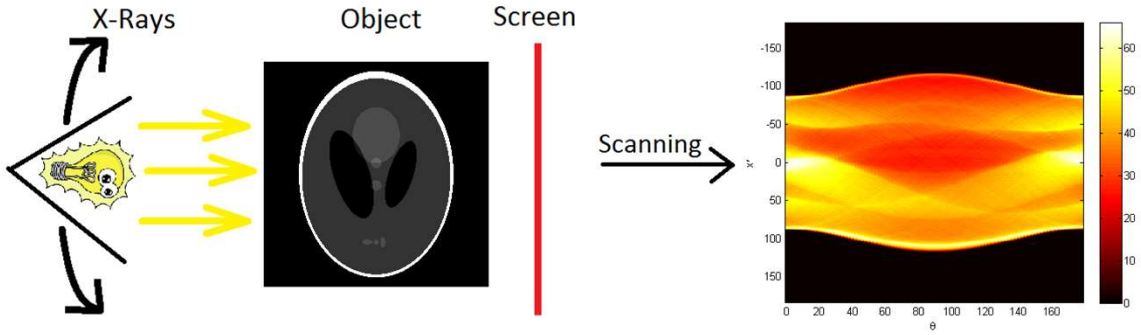

Advanced Topics

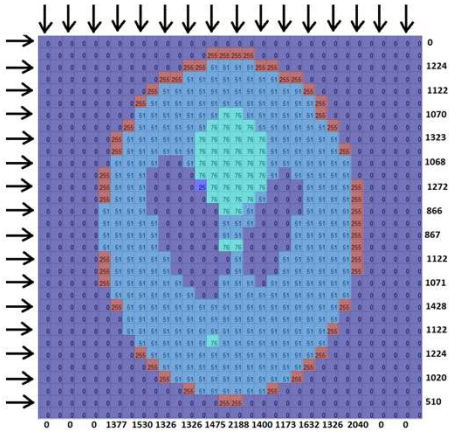
$$E(u) = \frac{1}{2}\|A(u) - f\|^2 + R(u)$$

Wie gut erklärt unsere aktuelle Schätzung u die Daten f?

Ist-unsere aktuelle Schätzung „regulär“ / rauschfrei genug?

Berechne ein $\hat{u} \in \arg\min_u E(u)$ als Lösung!





Maschinelles Lernen

Eingabe- bzw. Trainingsdaten x_i

„Netzwerk“ $N(x;\theta)$

Gewünschte Ausgaben y_i

Training: $\min_\theta \sum_{\text{Trainingsbeipiele } (x_i,y_i)} \text{Loss}(\mathcal{N}(x_i;\theta), y_i))$