03-MB-709.03

Echtzeitbildverarbeitung

Prof. Dr. Udo Frese

Überblick über den Prüfungsstoff

Konzepte

Algorithmen

Anwendungen

Echtzeitimplementierung

Überblick über den Prüfungsstoff

Überblick über den Prüfungsstoff

Was sollte man aus der Vorlesung (für die Prüfung) mitnehmen?

- **Algorithmen:**
  - Was wird berechnet?
  - Woraus?
  - Wie funktioniert die Berechnung („Idee in 2min“)?
  - Siehe Diagramm und Kurzzusammenfassung im folgenden.
- **Anwendungen:**
  - Was würde passieren, wenn man ein bestimmtes Verfahren anwendet?
  - Welche Probleme treten auf?
  - Wie kann man sie angehen?
  - Welche Voraussetzungen macht man bei einem Verfahren?
  - Wie kann man die Umgebung / Beleuchtung / Kamera passend gestalten?
  - Prüfung startet mit kleinem Anwendungsproblem.

Überblick über den Prüfungsstoff

Was sollte man aus der Vorlesung (für die Prüfung) mitnehmen?

- **Konzepte:**
  - Wie ist es formell definiert (Gleichung)?
  - Was bedeutet das real?
  - Wo (in welchem Algorithmus) verwendet man es?
  - Siehe Liste am Ende.
- **Echtzeitimplementierung:**
  - Wie optimiert man im Detail am Beispiel **Linien – Hough Transformation**?
  - LUT, Laufzeiger, Festkommaarithmetik, Ausnutzung des Gradienten, SIMD-Parallelisierung, Multicore-Parallelisierung

Konzepte

- **Fragen an das Auditorium erklären können.**
- **Momente, Hauptträgheitsachsen**
- **Farbdarstellung mit RGB (Rot, Grün, Blau)**
- **Filter und ihr Effekt**
- **Homogene Koordinaten / Koordinatensysteme**
- **Kameragleichung**
- **Probabilistik: P(..|..) (was bedeutet es anschaulich)**
- **im folgenden noch einmal alle Konzepte im Kurzüberblick.**
  - reicht, den Inhalt der folgenden Folien verstanden zu haben.
  - reicht *nicht*, den Inhalt der folgenden Folien auswendig zu lernen.

Konzepte

Hauptträgheitsachsen

- **Momente sind Integrale von $x^i y^j$ über die betrachtete Region**
  - Moment 0. Ordnung: I ist Integral über 1, die Fläche
  - Momente 1. Ordnung: Ix, Iy sind Integral über x bzw. y
  - Momente 2. Ordnung: Ixx, Ixy, Iyy sind Integral über x^2, xy, y^2
- **geschlossene Formeln für Intervall, Summe Intervalle einer Region**
- **Schwerpunkt Ix/I, Iy/I bestimmt Position der Region**
- **aus Eigensystem der Momente 2. Ordnung Matrix (Formel)**
  - erst auf den Schwerpunkt als Nullpunkt verschieben.
  - Hauptachsenrichtung θ bestimmt Orientierung der Region. Nur definiert, wenn $\sigma 1 > \sigma 2$.
  - $\sigma 1$, $\sigma 2$ bestimmen großen / kleinen Halbmesser äquivalenter Ellipse (Größe bzw. Länglichkeit)

Konzepte

Farbdarstellung mit RGB

- **Farbe**
  - physikalisch das Lichtspektrum (Energie vs. Wellenlänge)
  - Objektfarbe abhängig von Beleuchtung und Betrachtungswinkel
  - 3 menschliche Farbrezeptoren ⇒ Farbwahrnehmung aus Rot, Grün, Blau
  - Im Rechner RGB32: 1 Byte Rot, 1 Byte Grün, 1 Byte Blau, 1 Byte frei
  - Farbkamera: Mosaik - Farbfilter (Bayer Filter) vor dem CCD Chip
- **Farbsegmentierung**
  - Klassifikation (Rot, Grün, Blau)→ Klasse
  - z.B.: Tabelle manuell erstellen

Konzepte

Faltungsoperationen

- **Lineare, translationsinvariante Abbildung**
- **Ergebnispixel ist gewichtete Summe der Pixel in der Umgebung des Eingangsspixels**
- **Bilder glätten, Kontrast vergrößern, Kanten detektieren**
- **Skalieren der Faktoren, Addieren von Offsets um Wertebereich anzupassen**
- **Summe der Filterkoeffizienten gibt Antwort auf konstantes Bild (DC Anteil, von Directed Current = Gleichstrom)**
  - DC Anteil 1 $\Rightarrow$ Helligkeit „im Großen“ bleibt gleich (DC treu).
  - DC Anteil $\neq 0$ $\Rightarrow$ Filter skalierbar, so dass DC Anteil =1.
  - DC Anteil 0 $\Rightarrow$ Helligkeit „im Großen“ spielt keine Rolle (DC frei).
- **Wichtigstes Beispiel: SobelX, SobelY zur Kantendetektion.**

Konzepte

Homogene Koordinaten, Transformationen

- 4. Komponente entweder 0 (freier Vektor) oder 1 (Ortsvektor).
- Koordinatentransformationen als 4×4 Matrix.
- Matrix immer als AInB benennen, dann ist Konsistenz ablesbar.
- Spalte 1-3: X, Y, Z Achse von A in B-Koordinaten.
- Spalte 4: Ursprung von A in B Koordinaten.
- Q ist Orientierung (3-DOF), t ist Position (3-DOF).
- Q orthonormal, $Q^TQ=I$, Achsen Länge 1 und senkrecht
- Inverse mit Spezialformel (nicht die Formel selbst)
- Kameragleichung (ausführlich die Bedeutung erklären können)

$$\left(\begin{array}{ccc|c} & Q & & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

Konzepte

Kameragleichung

- **Definiert Abbildung**
  - Eingabe: p_W Punkt in Weltkoordinaten $\in R^3$
  - Eingabe: CInW Kamerapose in Weltkoordinaten $\in R^{4\times4}$
  - Ausgabe: (u,v) Punkt in Bildkoordinaten $\in R^2$
  - p: $R^3 \times R^{4\times4} \rightarrow R^2$:($p_W$, CInW) $\rightarrow$ (u,v),
- **Vier Schritte (in Formel zeigen können)**
  - Welt- zu Kamerakoordinaten
  - Perspektive
  - Verzerrung
  - Skalierung
- **welche Variablen sind in bestimmten Anwendungen (un-) bekannt?**

$$p\begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix} = \begin{pmatrix} x_C / z_C \\ y_C / z_C \end{pmatrix}$$

$$d_{\kappa_1,\kappa_2}\begin{pmatrix} x \\ y \end{pmatrix} =$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \cdot \left(1+\kappa_1\left(x^2+y^2\right)+\kappa_2\left(x^2+y^2\right)^2\right)$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \alpha \cdot d_{\kappa_1,\kappa_2}\left(p\left(CInW^{-1} \cdot p_W\right)\right)$$

Konzepte

Probabilistik:

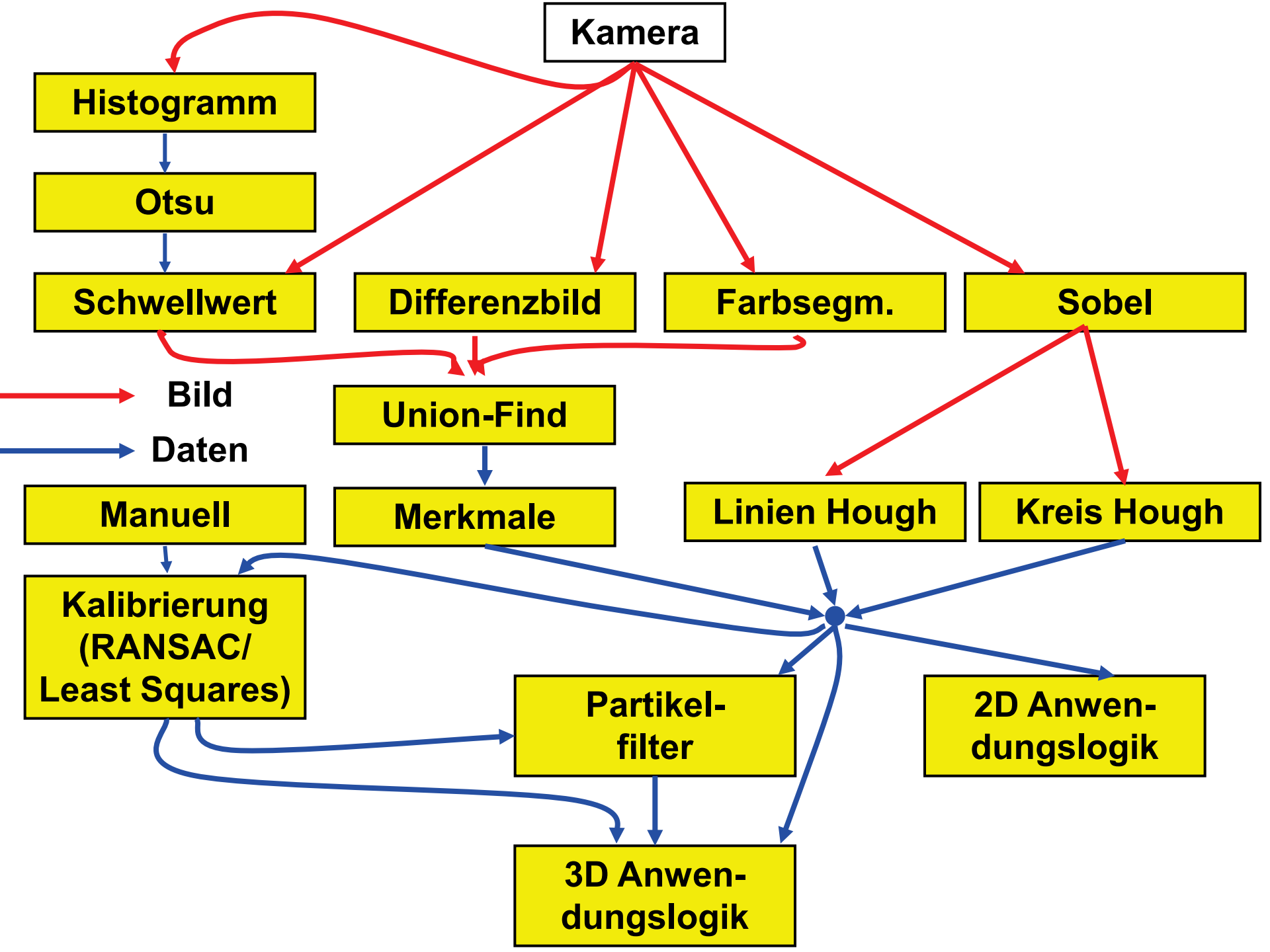
- **Bedeutung der Wahrscheinlichkeitsverteilungen (sonst keine Herleitungen)**
- **Least Square: X: Parameter, Z=z Messungen**
  - P(X=x|Z=z): W., dass Parameter x sind wenn z gemessen wurde (wird maximieren).
  - P(Z=z|X=x): W., z zu messen, wenn Parameter x wären (Modell, Gauss).
  - P(X=x): A-priori Wahrscheinlichkeit für die Parameter x. (Modell, oft uniform).
  - P(Z=z): Wahrscheinlichkeit z zu messen (Nicht wichtig, warum?).
- **Partikel Filter: X_t: Zustand, $Z_t=z_t$ Messung, $U_t=u_t$ ZÜM.**
  - $P(X_t=x_t \mid z_t, u_{t-1}, z_{t-1}, \ldots, z_1, u_0, z_0)$: Zustand gegeben alle Messungen (dargestellt durch Partikel) (kurze Schreibweise ohne =).
  - $P(Z_t=z_t|X_t=x_t)$: W. z_t zu messen, wenn der Zustand x_t wäre (Modell).
  - $P(X_t=x_t|X_{t-1}=x_{t-1},U_t=u_t)$: W. in Zustand x_t zu gelangen, wenn Vorgängerzustand x_{t-1} und ZÜM u_t wäre(Modell).

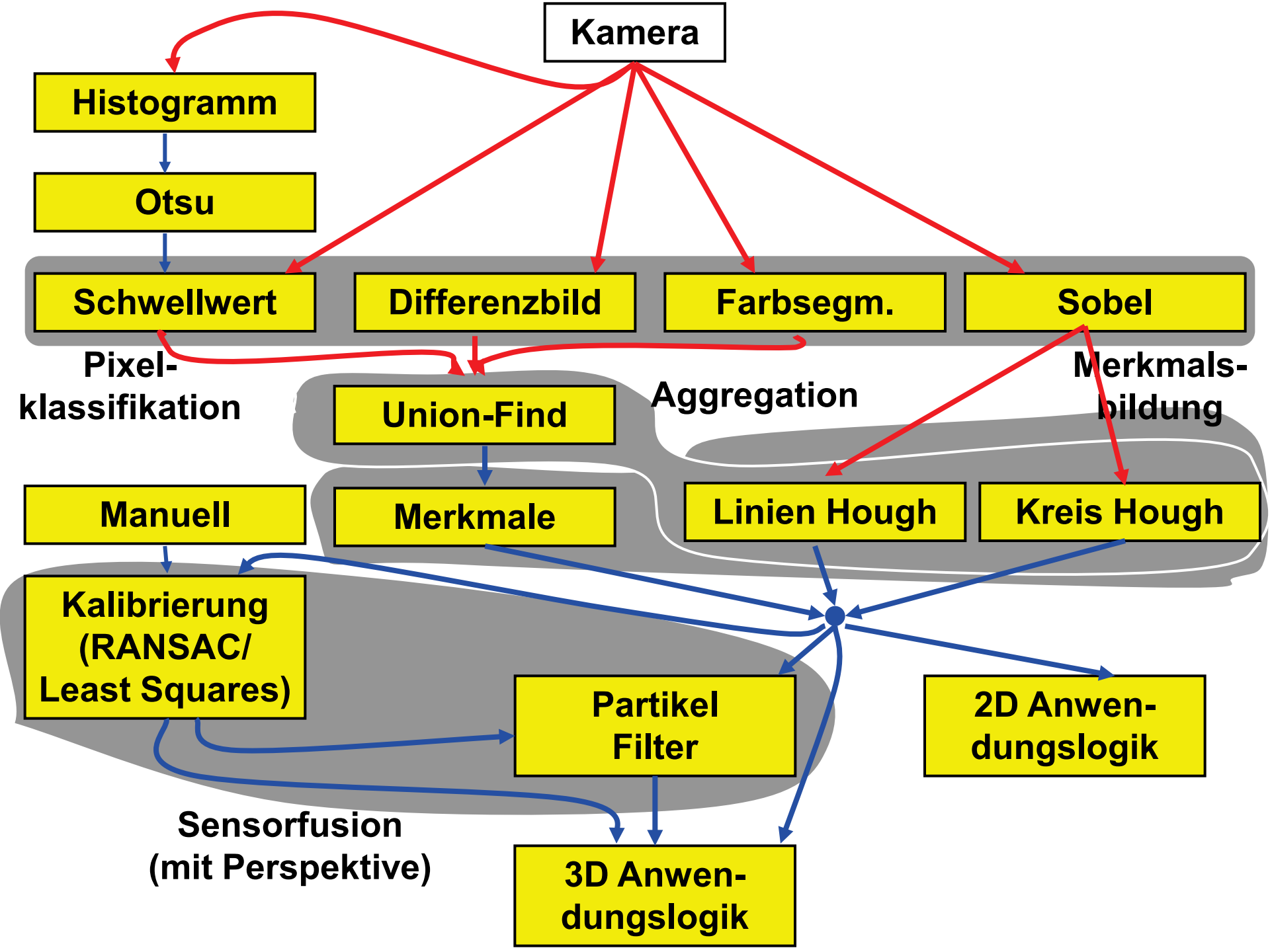
Algorithmen

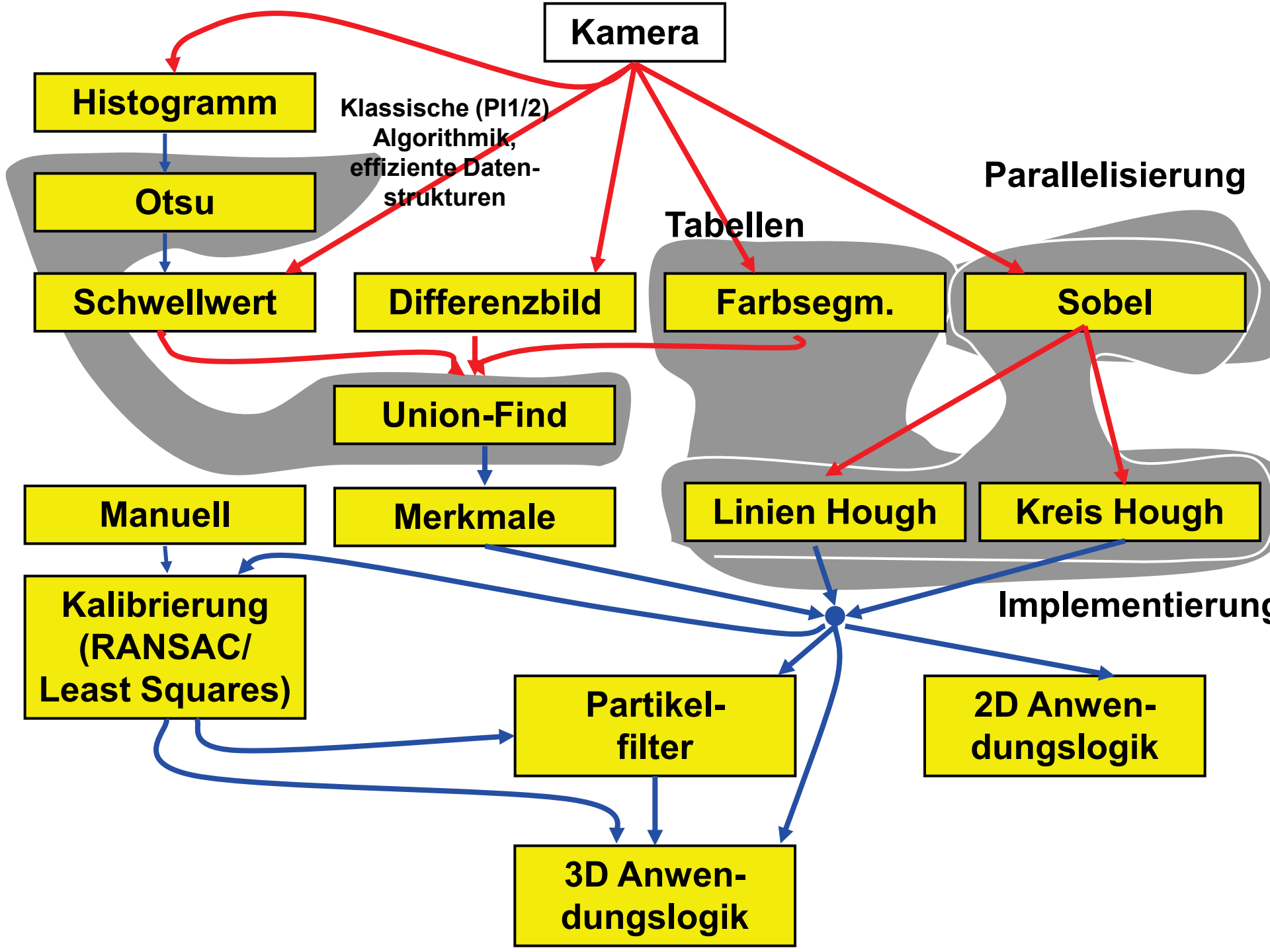
Algorithmen

Algorithmen:

- **Was wird berechnet?**
- **Woraus?**
- **Wie funktioniert die Berechnung („Idee in 2min“)?**
- **außer: Linien Hough.**
  - da auch die detaillierte Techniken zur effizienten Implementierung.
- **alle Algorithmen im Kurzüberblick.**
  - reicht, den Inhalt der folgenden Folien verstanden zu haben.
  - reicht *nicht*, den Inhalt der folgenden Folien auswendig zu lernen.

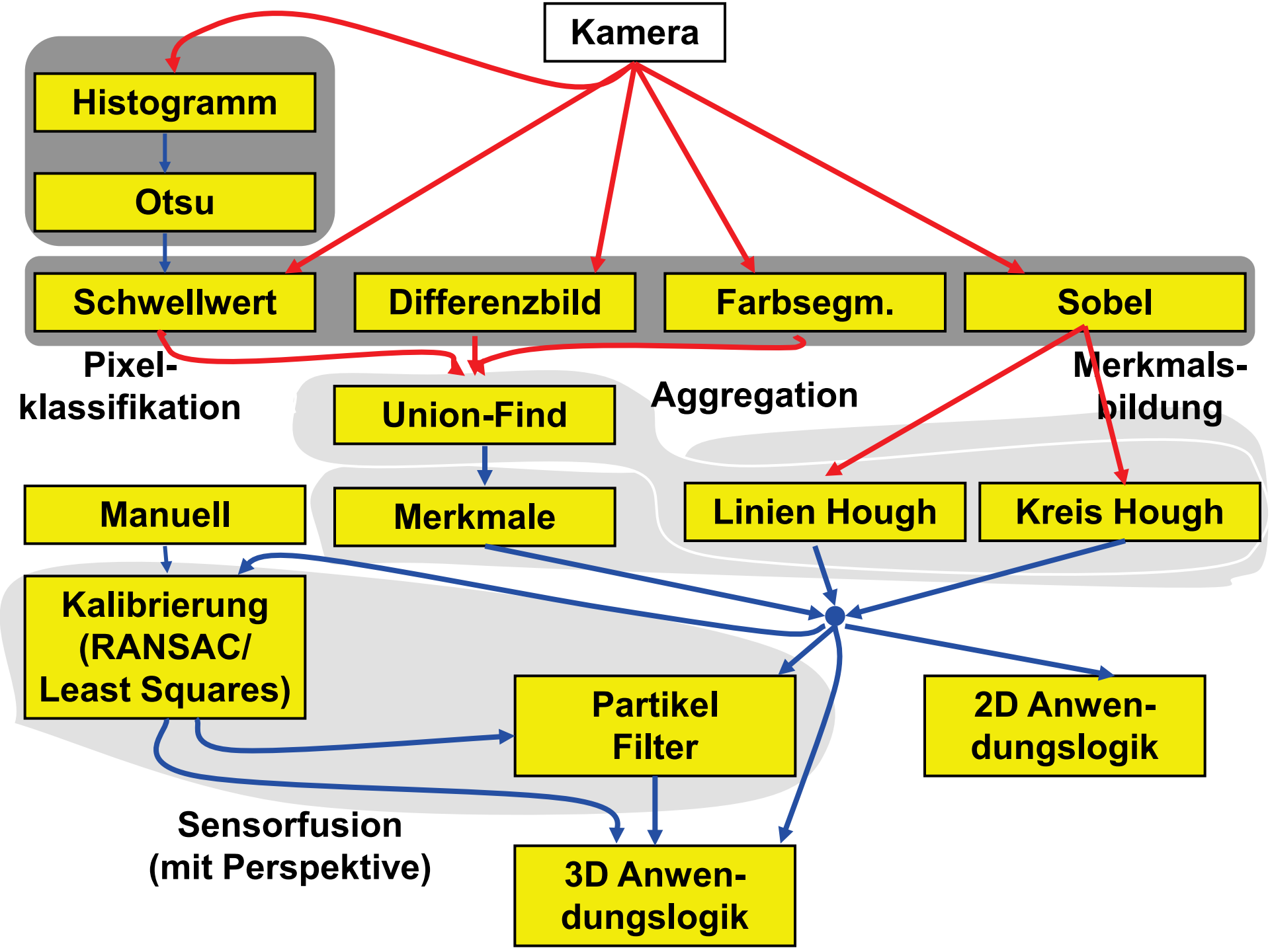

Kamera
Histogramm
Otsu
Schwellwert
Differenzbild
Farbsegm.
Sobel
Bild
Daten
Union-Find
Manuell
Merkmale
Linien Hough
Kreis Hough
Kalibrierung (RANSAC/ Least Squares)
Partikel- filter
2D Anwen- dungslogik
3D Anwen- dungslogik


Kamera
Histogramm
Otsu
Schwellwert
Differenzbild
Farbsegm.
Sobel
Pixel-
klassifikation
Union-Find
Aggregation
Merkmals-
bildung
Manuell
Merkmale
Linien Hough
Kreis Hough
Kalibrierung
(RANSAC/
Least Squares)
Partikel
Filter
2D Anwen-
dungslogik
Sensorfusion
(mit Perspektive)
3D Anwen-
dungslogik


Kamera
Histogramm
Klassische (PI1/2) Algorithmik, effiziente Daten-strukturen
Parallelisierung
Otsu
Tabellen
Schwellwert
Differenzbild
Farbsegm.
Sobel
Union-Find
Manuell
Merkmale
Linien Hough
Kreis Hough
Implementierung
Kalibrierung (RANSAC/ Least Squares)
Partikel-filter
2D Anwen-dungslogik
3D Anwen-dungslogik

Algorithmen

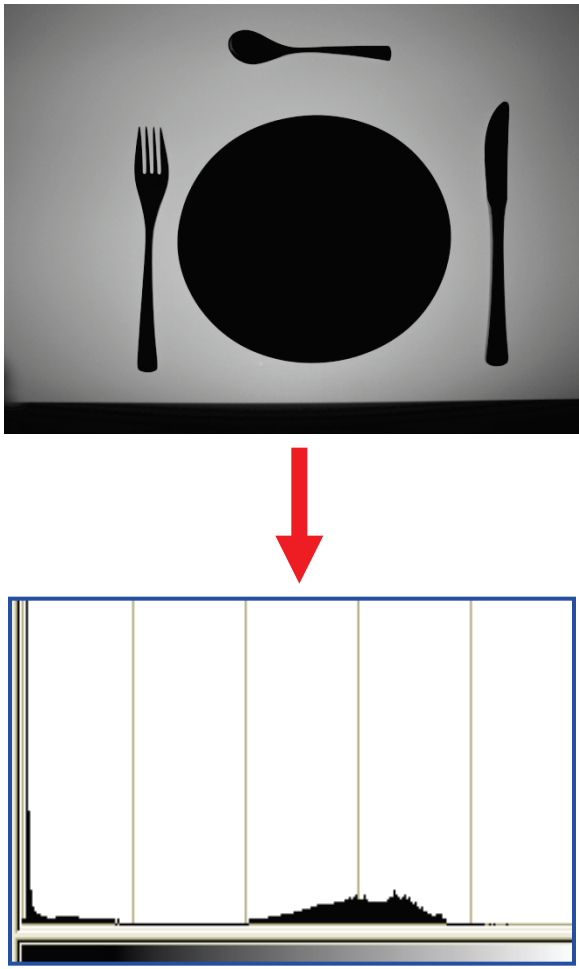
Pixelklassifikation & Schwellwertberechnung


Kamera
Histogramm
Otsu
Schwellwert
Differenzbild
Farbsegm.
Sobel
Pixel-
klassifikation
Union-Find
Aggregation
Merkmals-
bildung
Manuell
Merkmale
Linien Hough
Kreis Hough
Kalibrierung
(RANSAC/
Least Squares)
Partikel
Filter
2D Anwen-
dungslogik
Sensorfusion
(mit Perspektive)
3D Anwen-
dungslogik

Algorithmen

Histogramm

- **Was? Häufigkeit der verschiedenen Helligkeiten im Bild.**
- **Woraus? Bild.**
- **durchs Bild laufen, für jeden Pixel den Histogramm - Eintrag der zur Helligkeit des Pixels gehört erhöhen.**



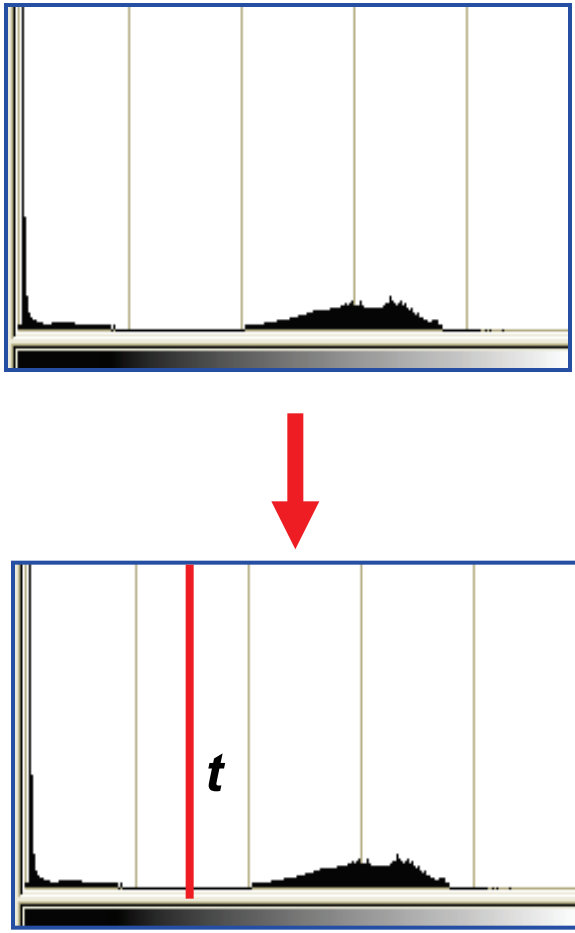
Algorithmen

Otsu automatischer Schwellwert

- **Was? Optimalen Schwellwert *t* zum binarisieren von hell/dunkel Bildern.**
- **Woraus? Histogramm *h*.**
- **bilde Pixel auf zwei Helligkeiten (i_{high}, i_{low}) für <*t* und >=*t* ab.**
- **minimiere den quadratischen Fehler:**

$$e(i_{low}, i_{high}) = \sum_{i=0}^{t} h(i)(i - i_{low})^2 + \sum_{i=t+1}^{255} h(i)(i - i_{high})^2$$

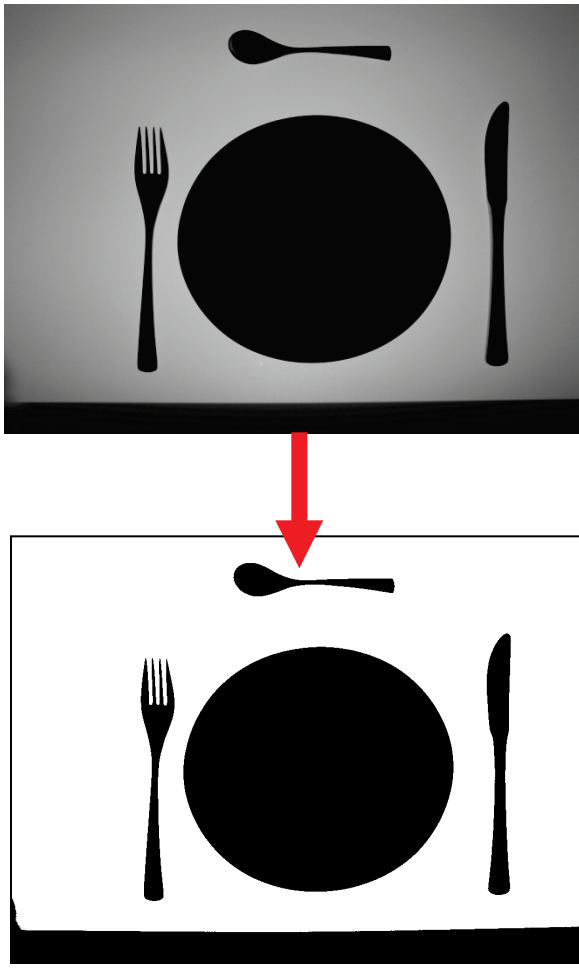
- **optimale i_{high}, i_{low} Werte berechenbar aus Summe *i* bzw. i^2 für Werte <t und >=t.**
- **t von 0..255 laufen, Summen aktualisieren.**



Algorithmen

Schwellwert

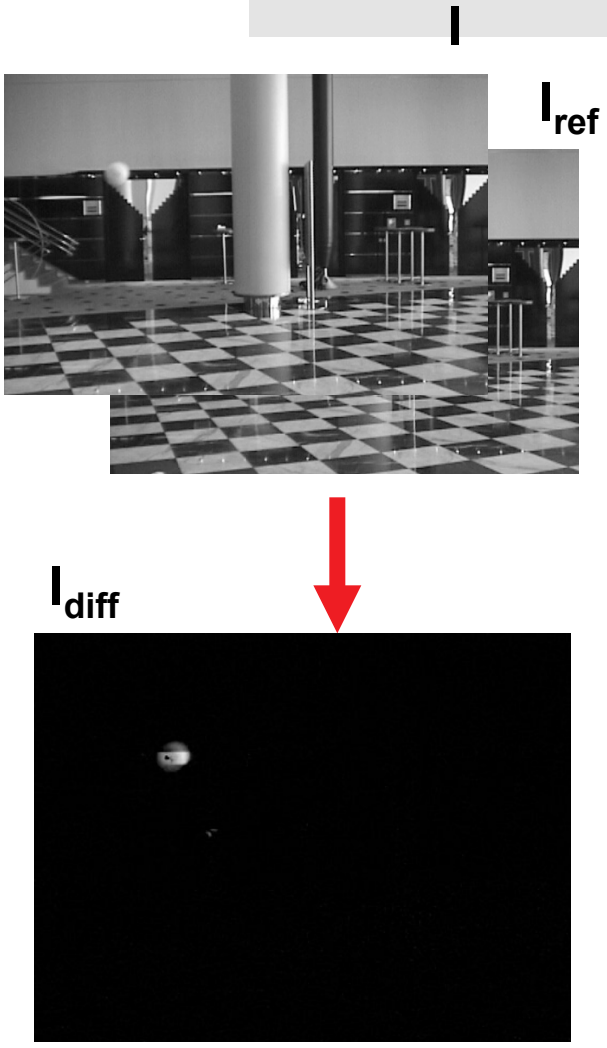
- Was? Binärbild, hell (bzw. Hintergrund) auf 1, dunkel (bzw. Objekt) auf 0.
- Woraus? Bild, Schwellwert *t.*
- alle Pixel >=t auf 1, alle <t auf 0.
- oder run length code:
- zusammenhänges Intervall von schwarzen Pixeln (x_{lo}, x_{hi}, y) codiert für $\{(x,y) \mid x \in [x_{lo}..x_{hi}]\}$.
- durchlaufen des Bildes, vergleichen mit *t* und zählen.



Algorithmen

Differenzbild

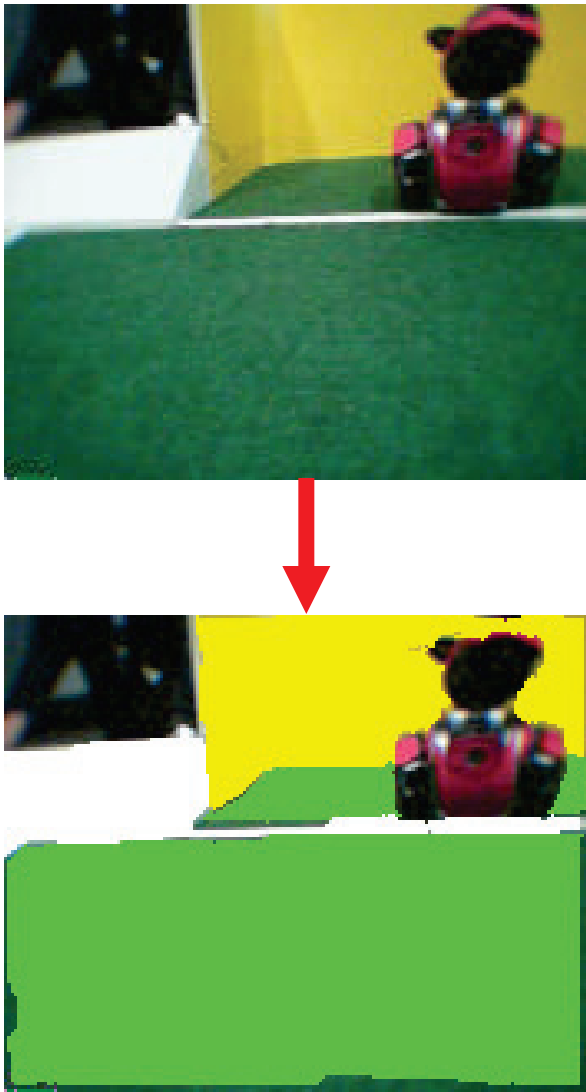
- Was? Differenzbild I_{diff}
- Woraus? aktuelles Bild (I), Referenzbild (I_{ref})
- Betrag der Differenz zwischen aktuellem Bild und Referenzbild ausrechnen $I_{diff} = |I - I_{ref}|$
- zeigt, wo im Bild Bewegung ist
- referenzbild einmal fest aufnehmen
- besser: gleitend dem aktuellen Bild nachführen: $I_{ref} = \alpha I + (1-\alpha) I_{ref}$, mit α z.B. 0.1



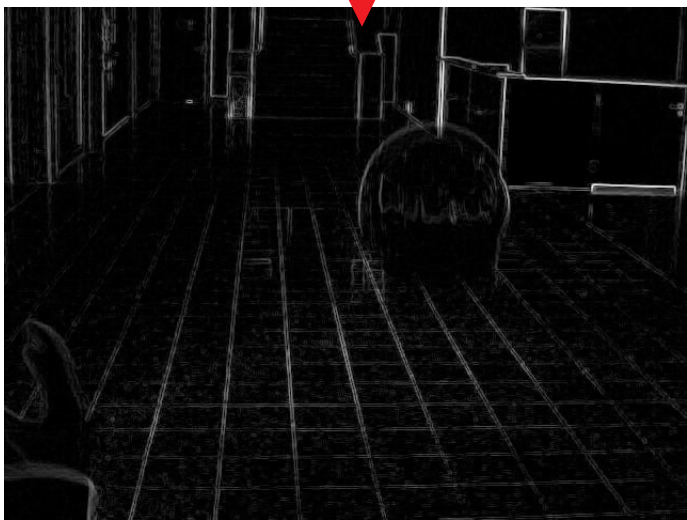
Algorithmen

Farbsegmentierung

- **Was? Für jeden Pixel im Bild eine Farbklasse (z.B. Rasengrün, Ballorgange, Torgelb)**
- **Woraus? Bild und Tabelle Farbvektor auf Klasse**
- **Tabelle muss von Hand erstellt werden**



Algorithmen



Sobel Operator

- **Was? Kantenbild, in dem die Helligkeit angibt, wie stark die unmittelbare Umgebung eines Pixel nach einer Kante aussieht (Betrag des Sobelvektors).**
- **Was? Zusätzlich Richtung der Kante (Richtung des Sobelvektors).**
- **Woraus? Bild.**
- **laufe durch das Bild und berechne zwei jeweils speziell gewichtete Summen der 8 Nachbarpixeln eines jeden Pixel.**
- **Koeffizienten SobelX, SobelY.**

Algorithmen

Sobel Operator

- **SobelX**
  - glättet (Tiefpass) in Y Richtung
  - differenziert in X Richtung.
- **SobelY**
  - glättet (Tiefpass) in X Richtung
  - differenziert in Y Richtung.
- **für lineare Helligkeitsverläufe ist (SobelX, SobelY) 8* der Gradient.**
- **Richtung zeigt senkrecht zur Kante auf die hellere Hälfte.**
- **Länge gibt Stärke des Kontrastes entlang der Kante an.**

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

SobelX

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

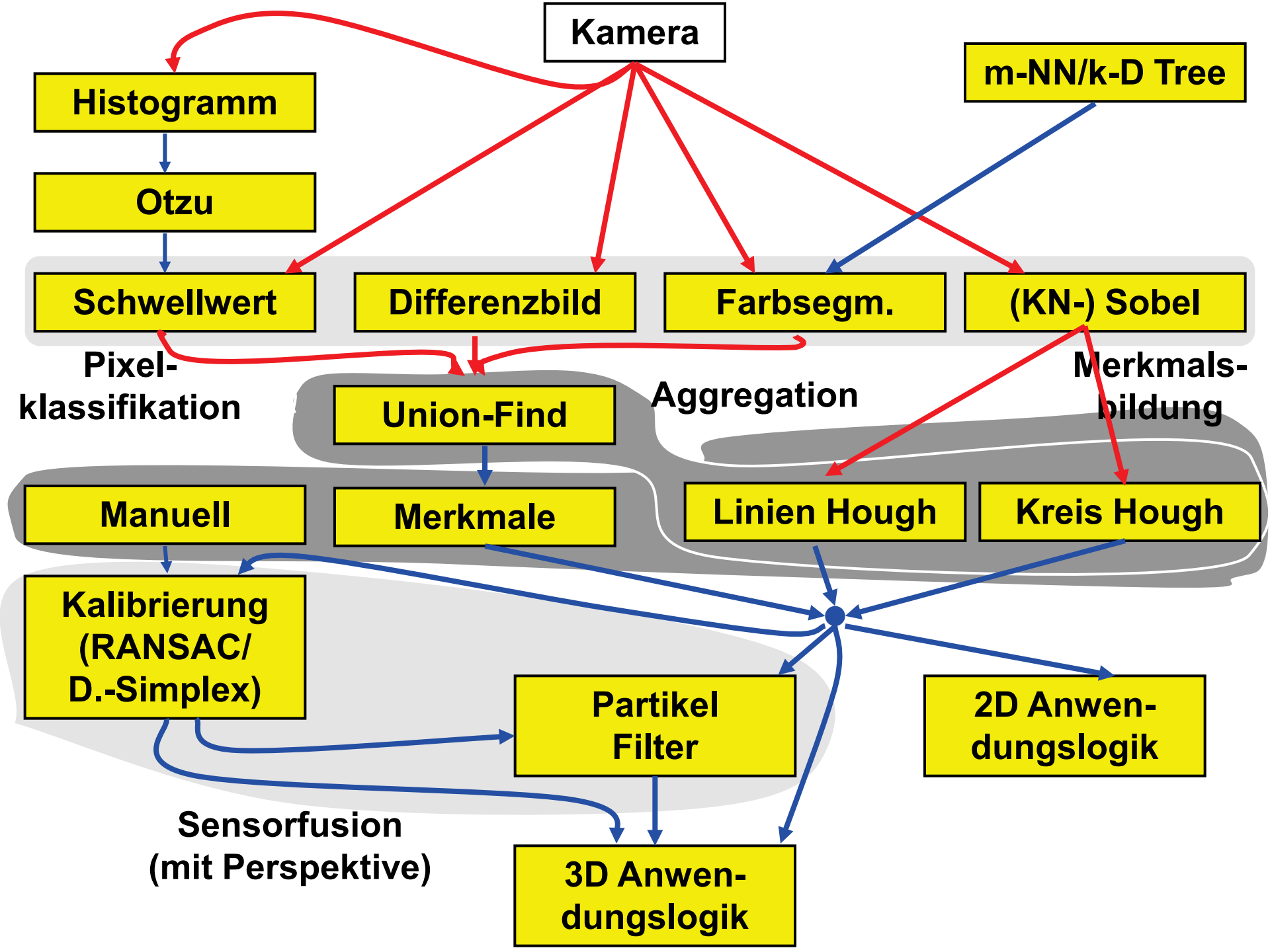
SobelY

$$\arctan 2\left(Sobel_y, Sobel_x\right)$$

$$\sqrt{Sobel_x^{\,2} + Sobel_y^{\,2}}$$

Algorithmen

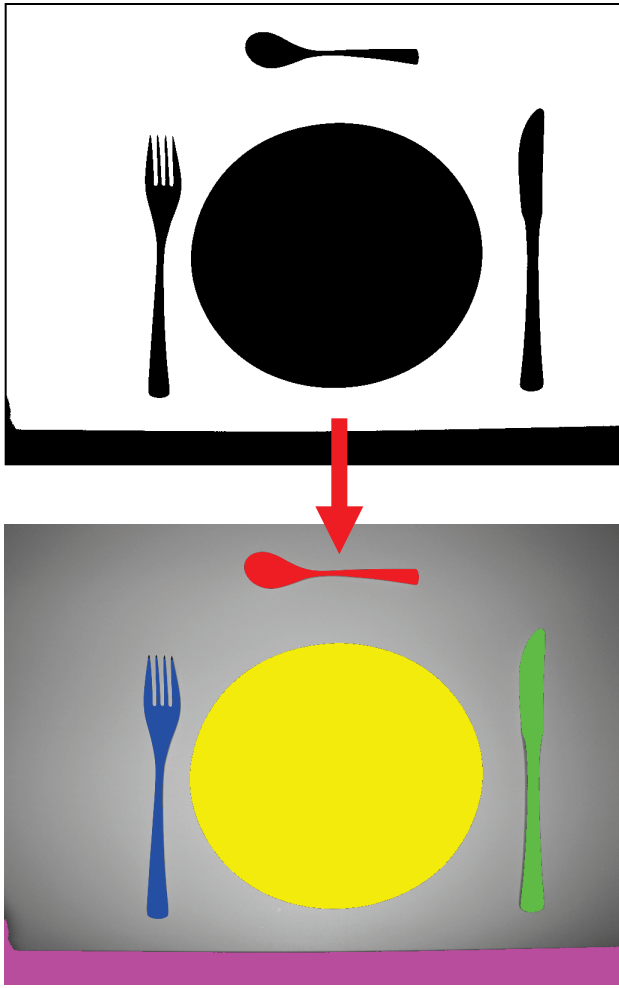
Aggregation & Merkmalsfindung


Kamera
m-NN/k-D Tree
Histogramm
Otzu
Schwellwert
Differenzbild
Farbsegm.
(KN-) Sobel
Pixel-
klassifikation
Union-Find
Aggregation
Merkmals-
bildung
Manuell
Merkmale
Linien Hough
Kreis Hough
Kalibrierung
(RANSAC/
D.-Simplex)
Partikel
Filter
2D Anwen-
dungslogik
Sensorfusion
(mit Perspektive)
3D Anwen-
dungslogik

Algorithmen

Union-Find-Algorithmus

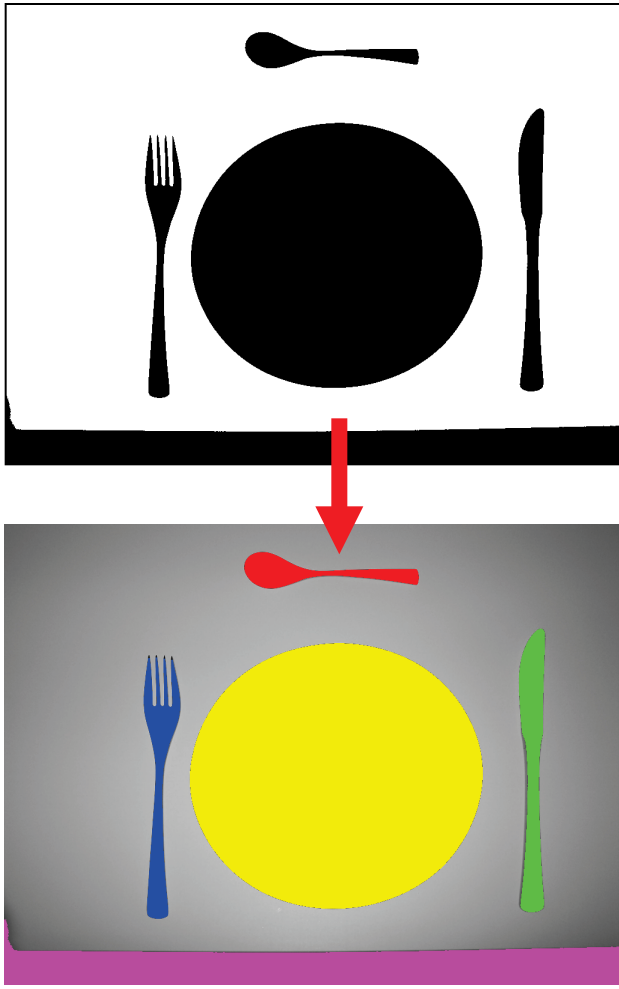
- **Was? Run length encoded Binärbild, in eine zusammenhängende Region mit einheitlicher Nummer (Farbe) markiert ist.**
- **Woraus? Run length encoded Binärbild.**
- **verwalte eine Union-Find Datenstruktur über den Intervallen.**
- **laufe mit zwei Zeigern `run`, `flw` durch die Intervalle von oben nach unten, links nach rechts durch.**
- **`flw` folgt `run` mit einer Zeile Abstand. Sobald `flw` mehr als eine Zeile zurückliegt wird er weiter gezogen.**



Algorithmen

Union-Find-Algorithmus

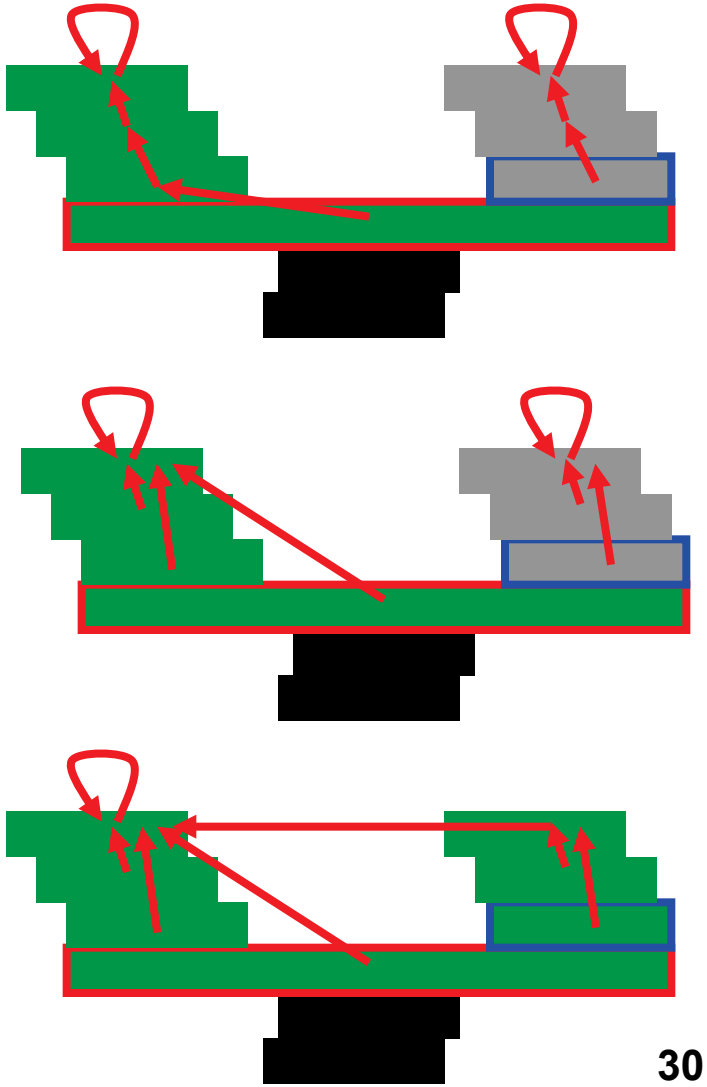
- Berühren sich die Intervalle `run` und `flw`, werden sie in der Union-Find Datenstruktur vereinigt und `run` oder `flw` (der früher endet) weiter gezogen.



Algorithmen

Union-Find-Algorithmus

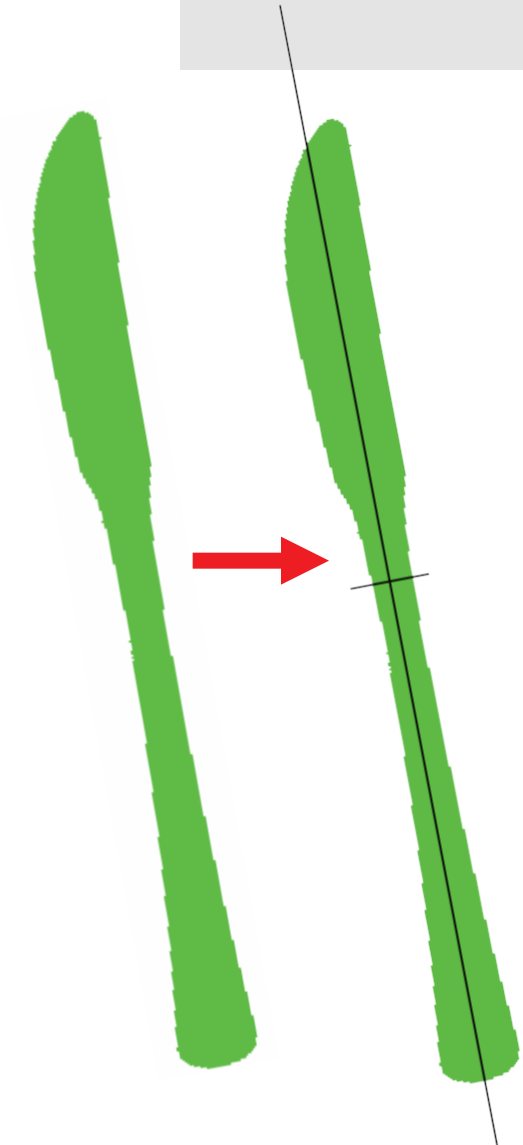
- Union Find Datenstruktur: Jedes Intervall hat einen Zeiger auf ein Eltern Intervall das vor ihm liegt oder sich selbst (Wurzel)
- Intervalle mit gleicher Wurzel gehören zur gleichen Komponente.
- Pfadkomprimierung: Von einem Intervall bis zur Wurzel hoch alle Intervalle direkt auf die Wurzel zeigen lassen. Macht Repräsentation effizienter ändert nichts an den dargestellten Regionen.
- Vereinigen zweier Regionen: Pfadkom-primierung von beiden, dann die spätere Wurzel auf die frühere zeigen lassen.



Algorithmen

Merkmale:

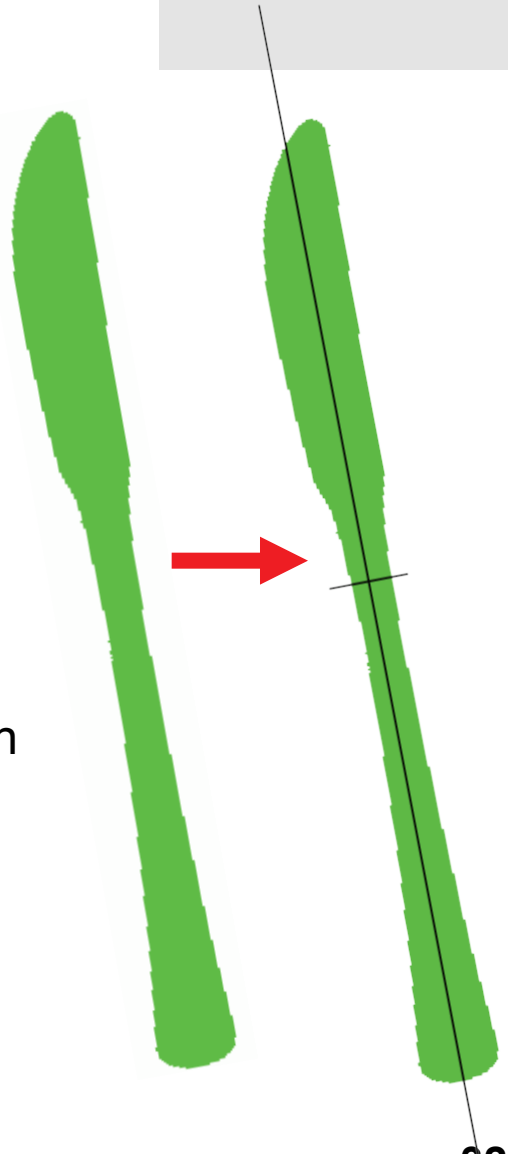
- **Was? Fläche, Schwerpunkt, Hauptträgheits-achsen, also Zahlen, die in einem integralen, d.h. gemittelten Sinne Aufschluss über die Gestalt und Lage einer Region geben.**
- **Woraus? In Regionen gruppiertes run length encoded Binärbild.**
- **Momente 0. bis 2. Ordnung berechnen**
- **Momente sind Integrale über $x^i y^j$ über die betrachtete Region**
  - Moment 0. Ordnung: I ist Integral über 1, die Fläche
  - Momente 1. Ordnung: I_x, I_y sind Integral über x bzw. y
  - Momente 2. Ordnung: I_{xx}, I_{xy}, I_{yy} sind Integral über x^2, xy, y^2



Algorithmen

Merkmale:

- **Momente sind additiv.**
- **geschlossene Formeln für Intervall (Formel selbst nicht nötig), Summe Intervalle einer Region.**
- **Schwerpunkt *lx*/*l*, *ly*/*y* bestimmt Position der Region.**
- **Aus Eigensystem der Momente 2. Ordnung Matrix**
  - erst auf den Schwerpunkt verschieben.
  - Hauptachsenrichtung θ bestimmt Orientierung der Region
  - σ_1, σ_2 bestimmen großen / kleinen Halbmesser äquivalenter Ellipse (Größe bzw. Länglichkeit)



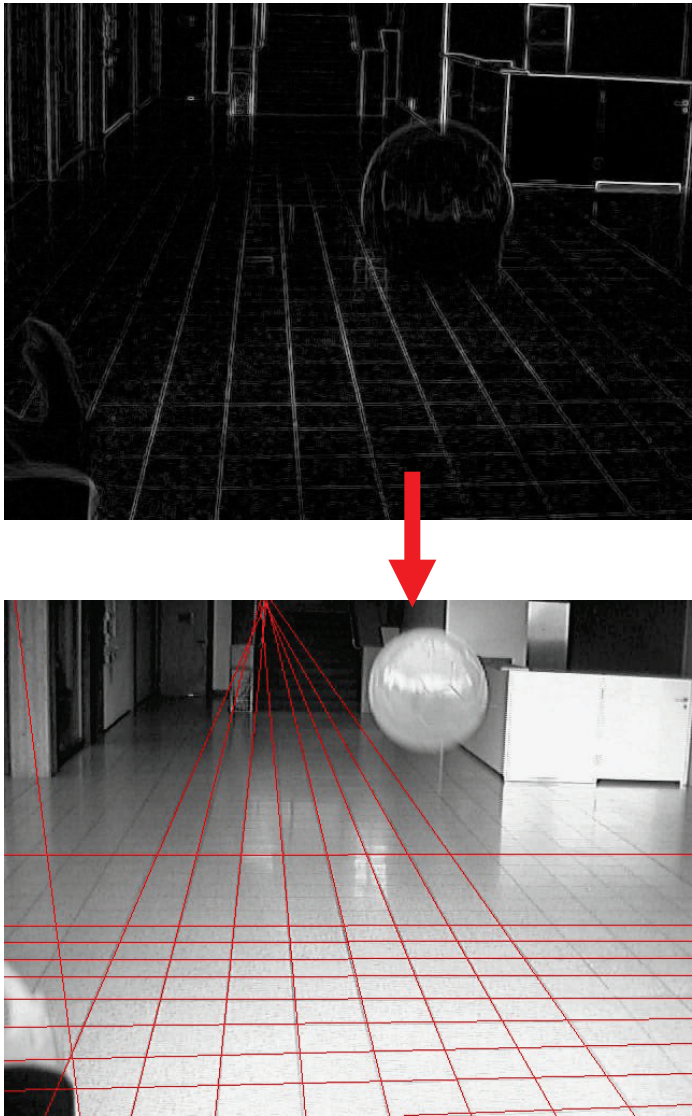
Algorithmen

Linien Hough:

- **Was? Mengen von Geraden (keine Strecken) im Bild.**
- **Woraus? Sobel Bild (Betrag und Richtung)**
- **Pixel im Houghraum definiert eine Gerade in Hessescher Normalform mit Parameter (α, *d*).**

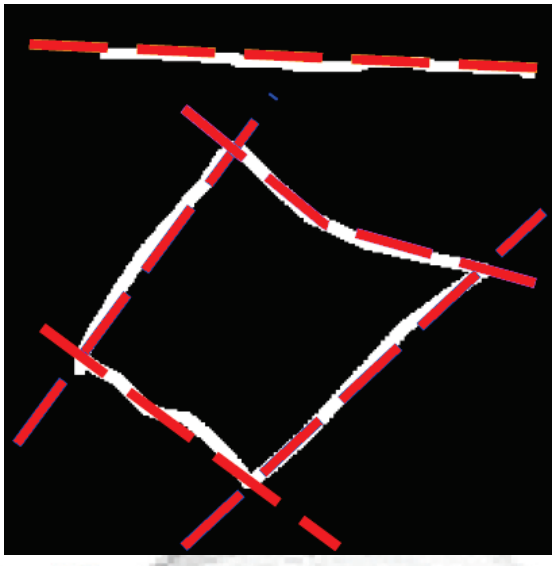
$$x\cos\alpha + y\sin\alpha - d = 0$$

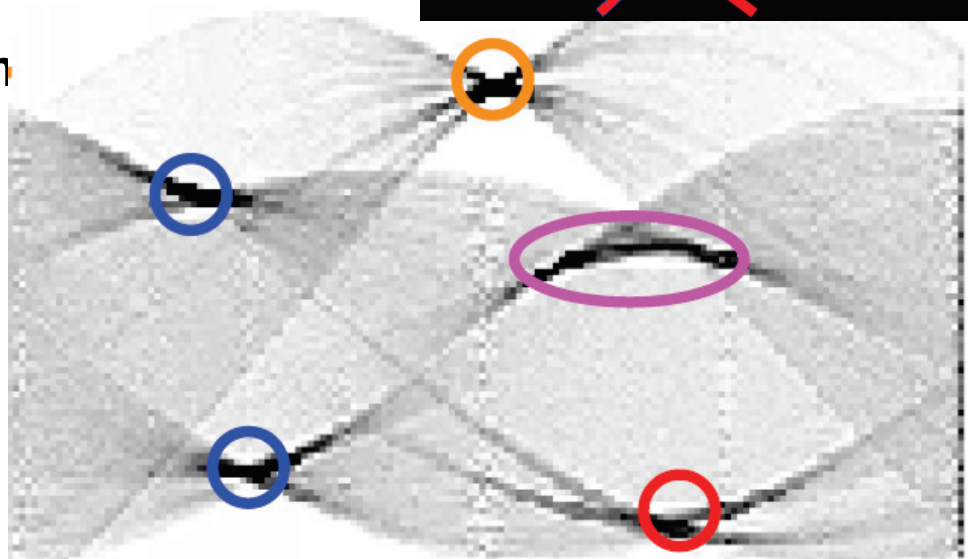
- **zu jedem Pixel (*x*,*y*) mit hinreichend hohem Sobel Betrag (Schwellwert) werden alle (α, *d*) erhöht, die die Gleichung erfüllen.**
- **dazu α durchlaufen, *d* ausrechnen.**
- **Houghakkumulator um 1 erhöhen**



Algorithmen

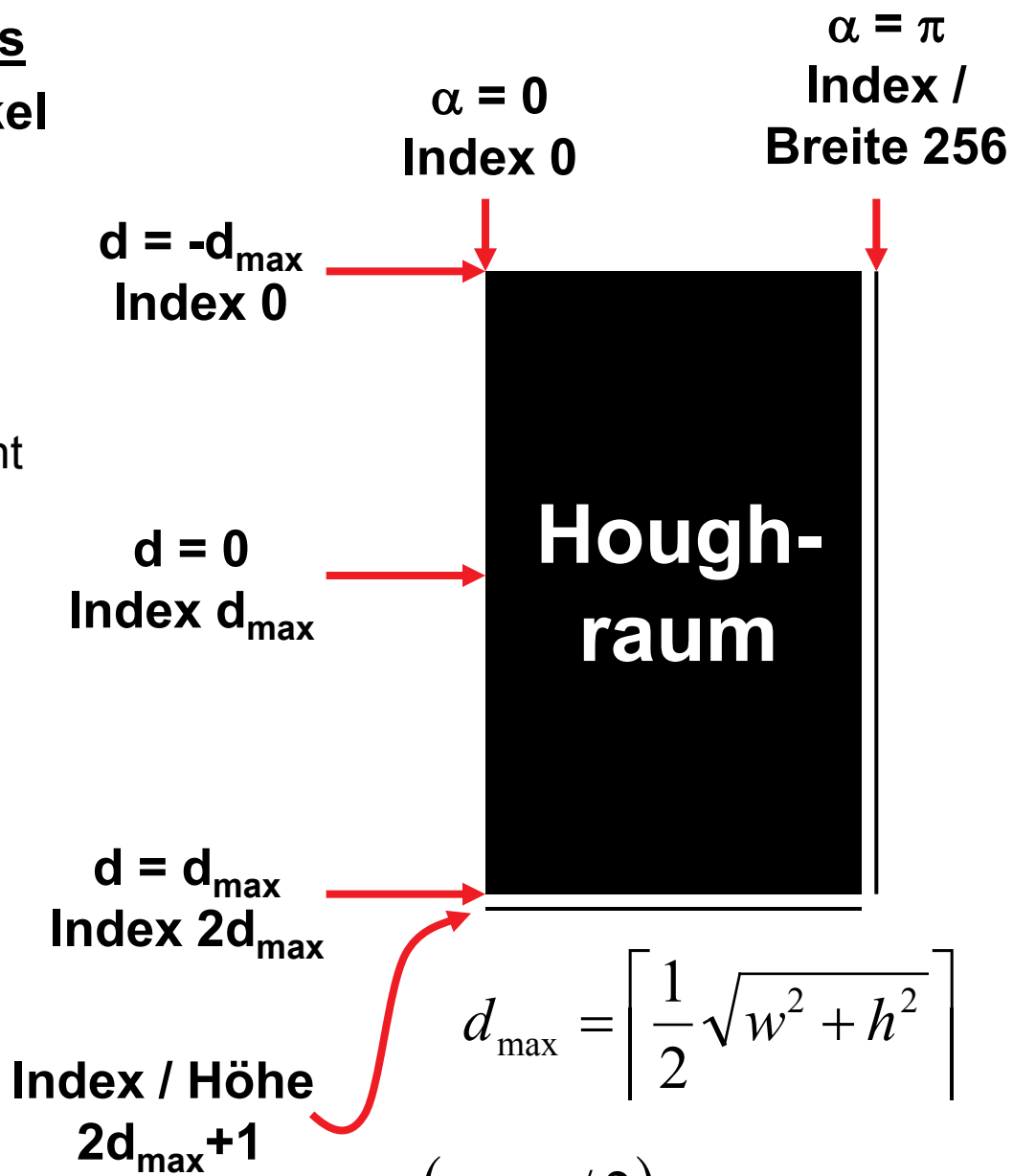
- Menge der Geraden die durch einen festen Punkt (x,y) gehen bilden im Bildraum einen Stern, im Hough-Raum eine Sinuswelle.
- Optimierung: Winkel α läuft nur Richtung des Sobelvektors +/- Unsicherheit.
- hinterher alle Houghwerte als Geraden nehmen, die über einem Schwellwert liegen und in einer Umgebung maximal sind.
- *d* auf Bildmitte beziehen, um Wertebereich für d einzuschränken.





Struktur des Linienhoughraumes

- **für Linienextraktion, 256 Winkel**
- **Winkel α**
  - α von 0 bis π (excl.)
  - Houghraum Breite 256
  - Indizes 0 bis 255 (incl.)
  - 256 entspricht π (excl.) äquivalent zu 0 (wrap around)
- **Distanz d**
  - d Bezug auf Bildmitte (w/2,h/2)
  - Diskretisiert 1 (1 Pixel d-Distanz Bildraum = 1 Pixel Houghraum)
  - d von $-d_{max}$ bis $+d_{max}$ (incl.)
  - Indices 0 bis $2d_{max}$ (incl.)
  - Houghraumhöhe $2d_{max}+1$

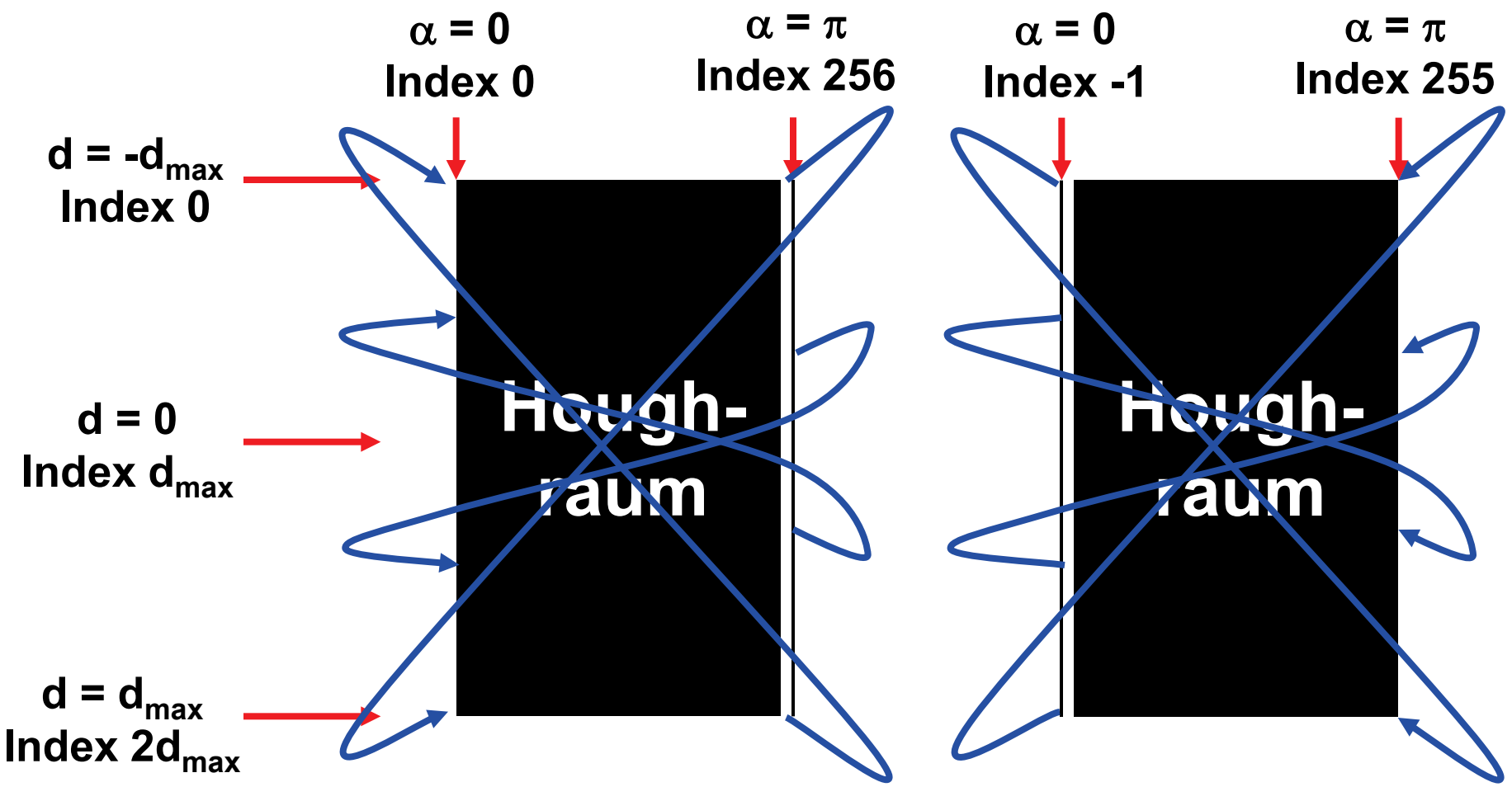


$$d_{\max} = \left\lceil \frac{1}{2}\sqrt{w^2 + h^2} \right\rceil$$

$$(x - w/2)\cos\alpha + (y - h/2)\sin\alpha - d = 0$$

Struktur des Linienhoughraumes

- α-wrap around mit Vorzeichenwechsel bei d
- (π,d) entspricht $(0,-d)$
- Index $(256, d_{idx})$ entspricht $(0, 2d_{max}-d_{idx})$
- Index $(-1,d_{idx})$ entspricht $(255, 2d_{max}-d_{idx})$



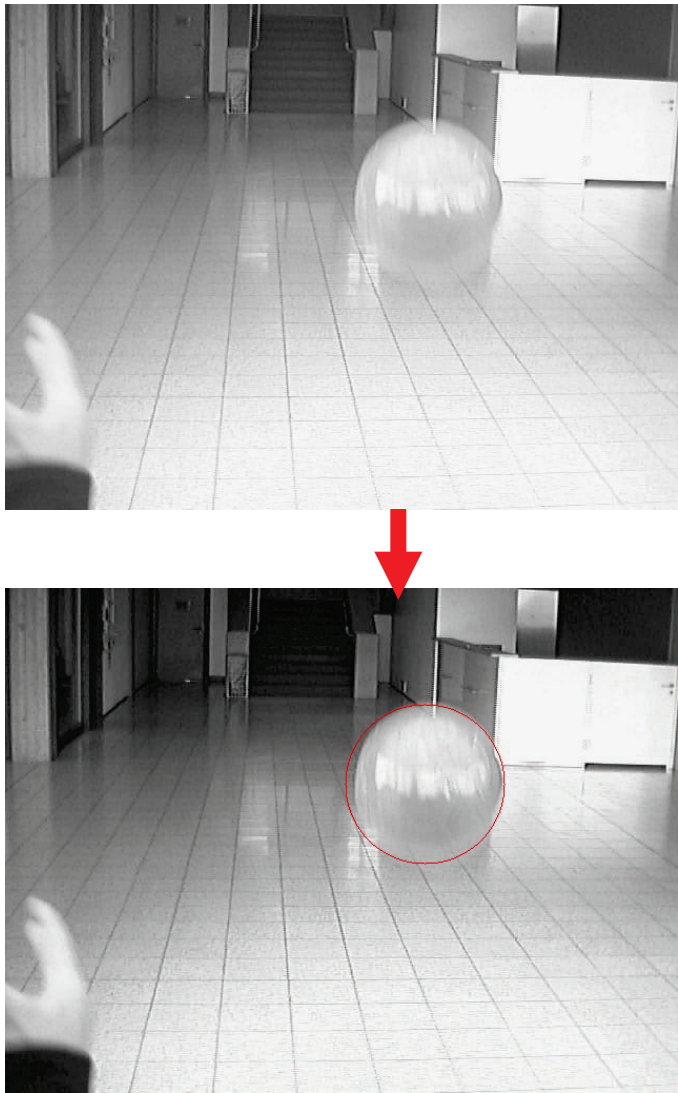
Algorithmen

Kreis Hough Transformation

- Was? Menge von Kreisen im Bild.
- Woraus? Sobel Bild (Betrag und Richtung)
- Pixel im Hough Raum definiert einen Kreis in Mittelpunktsform:

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$

- um Platz zu sparen nur (x_c, y_c) als Houghraum. r weglassen. Rand von r_{max}.
- zu jedem Pixel (*x*,*y*) mit hinreichendem Sobelkontrast für r $\in$[r_{min}..r_{max}] entlang und entgegen der Sobel Richtung laufen und die dortigen Hough Zellen erhöht.



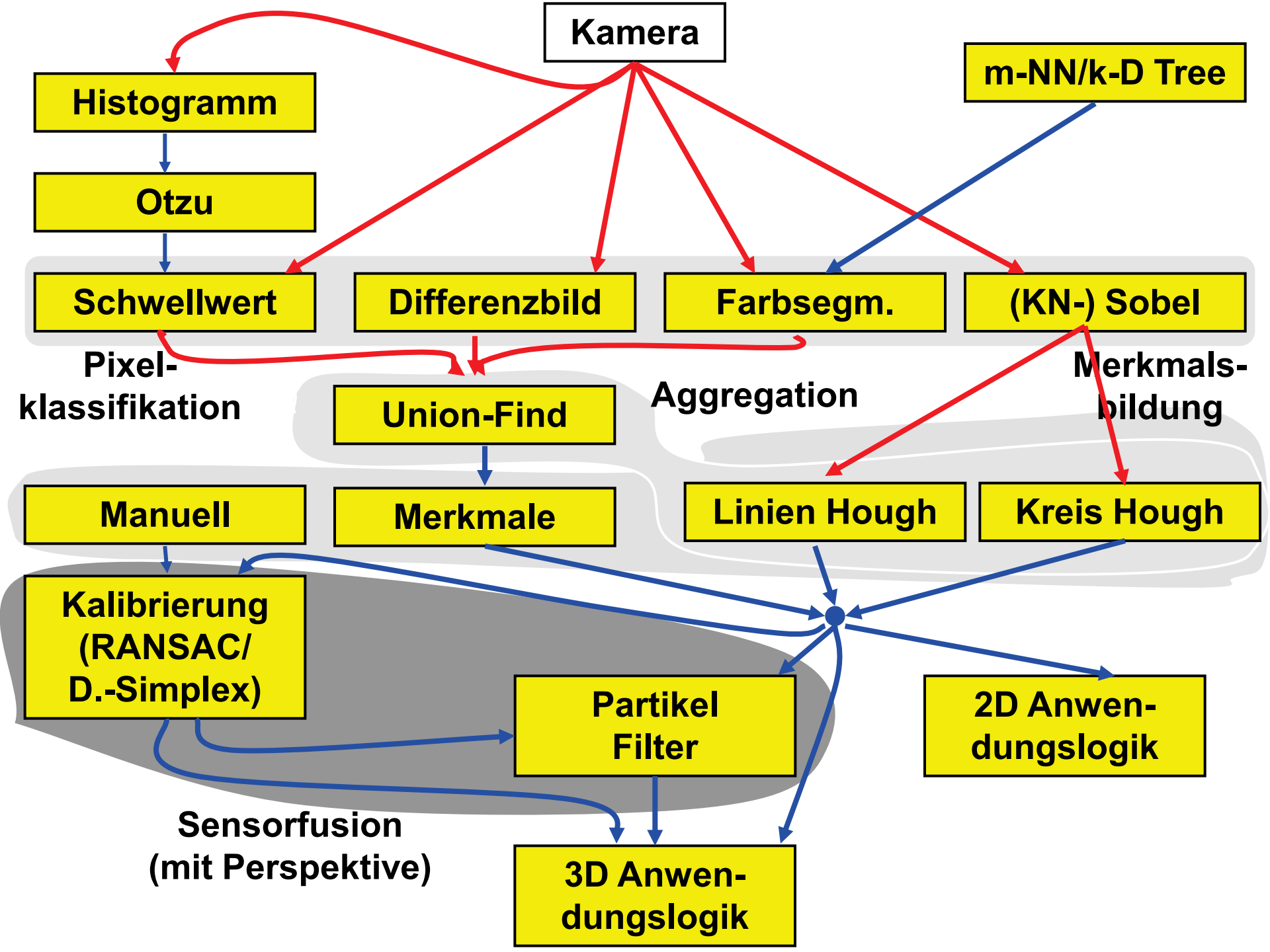
Algorithmen

Manuell

- **Was? Punkte im Bild**
- **Woraus? Bild**
- **Manuell anwählen.**
- **für Kalibrierung eine gute Idee, weil Menschen sehr viel robuster erkennen können als Bildverarbeitung**

Algorithmen

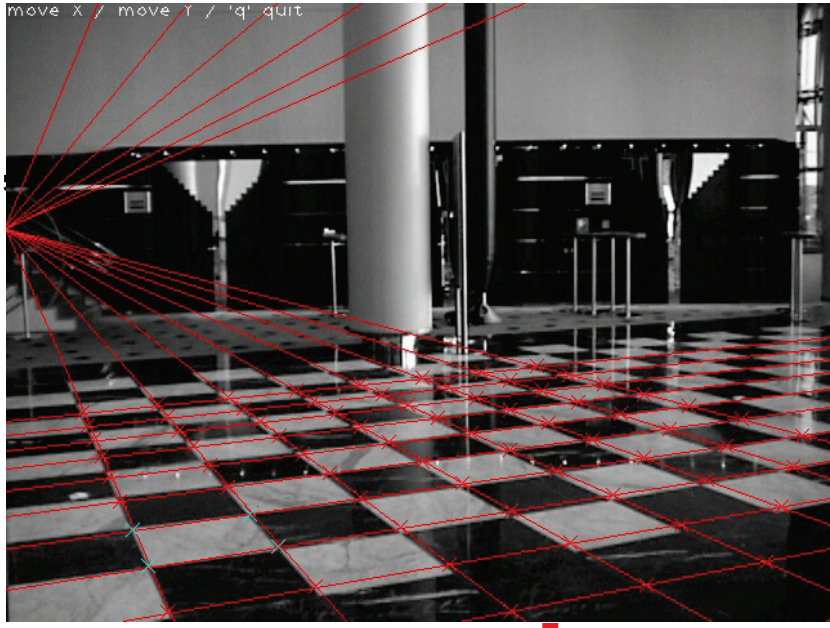
Sensorfusion

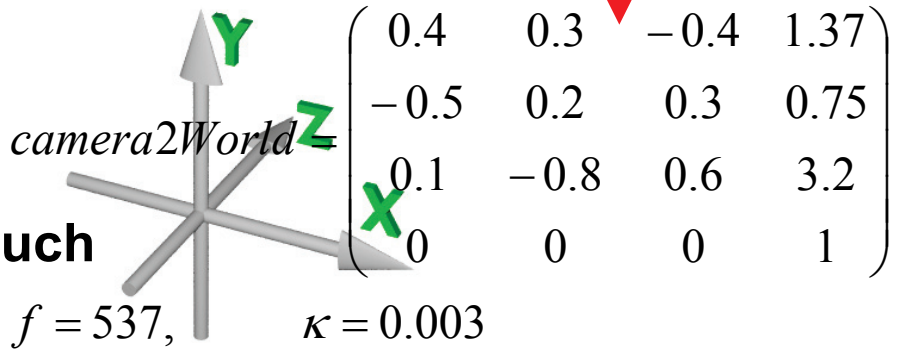

Kamera
m-NN/k-D Tree
Histogramm
Otzu
Schwellwert
Differenzbild
Farbsegm.
(KN-) Sobel
Pixel-
klassifikation
Union-Find
Aggregation
Merkmals-
bildung
Manuell
Merkmale
Linien Hough
Kreis Hough
Kalibrierung
(RANSAC/
D.-Simplex)
Partikel
Filter
2D Anwen-
dungslogik
Sensorfusion
(mit Perspektive)
3D Anwen-
dungslogik

Algorithmen

Kalibrierung (RANSAC)

- Was? Kameraparameter (Kamerapose, Brennweite, Verzerrung), Zuordnung von Bildpunkten zu Weltpunkten
- Woraus? Bildpunkte und Weltpunkte
- berechne hypothetische Kalibrierung aus zufällig gezogenen Zuordnungen (so viele DOF, wie nötig, hier 4)
- projiziere mit hypothetischer Kalibrierung alle Weltpunkte über Kameragleichung
- zähle, wie viele passen
- wiederhole und nehme besten Versuch
- auch mit Linien statt Punkten





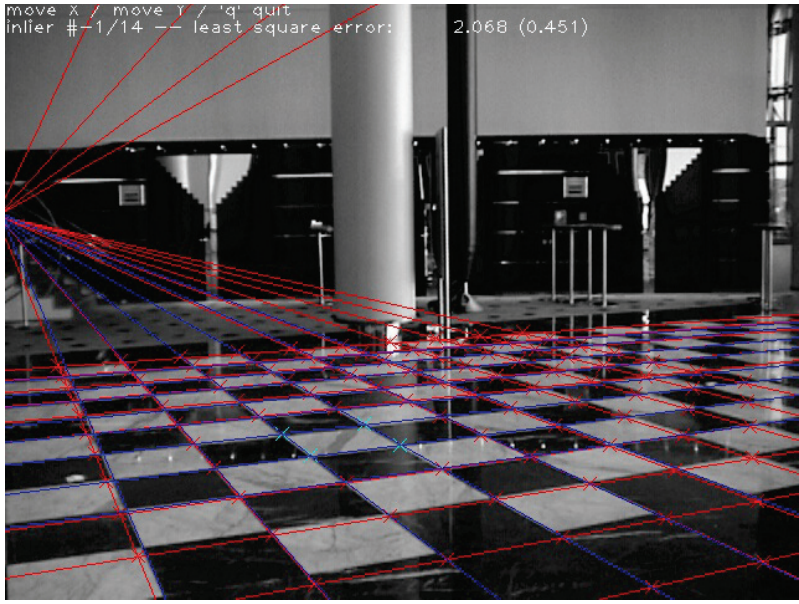
$$camera2World = \begin{pmatrix} 0.4 & 0.3 & -0.4 & 1.37 \\ -0.5 & 0.2 & 0.3 & 0.75 \\ 0.1 & -0.8 & 0.6 & 3.2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

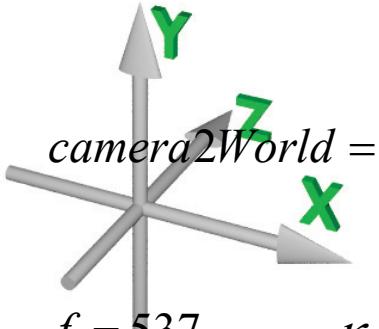
$f = 537, \quad \kappa = 0.003$

Algorithmen

Kalibrierung (Least Squares)

- **Was? Kameraparameter (Kamerapose, Brennweite, Verzerrung)**
- **Woraus? Bildpunkte und *zugeordnete* Weltpunkte.**
- **Für hypothetische Kameraparameter (T_C^W, f, κ) errechnet sich Bildposition aus Kameragleichung**





$$camera2World = \begin{pmatrix} 0.4 & 0.3 & -0.4 & 1.37 \\ -0.5 & 0.2 & 0.3 & 0.75 \\ 0.1 & -0.8 & 0.6 & 3.2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$f = 537, \qquad \kappa = 0.003$$

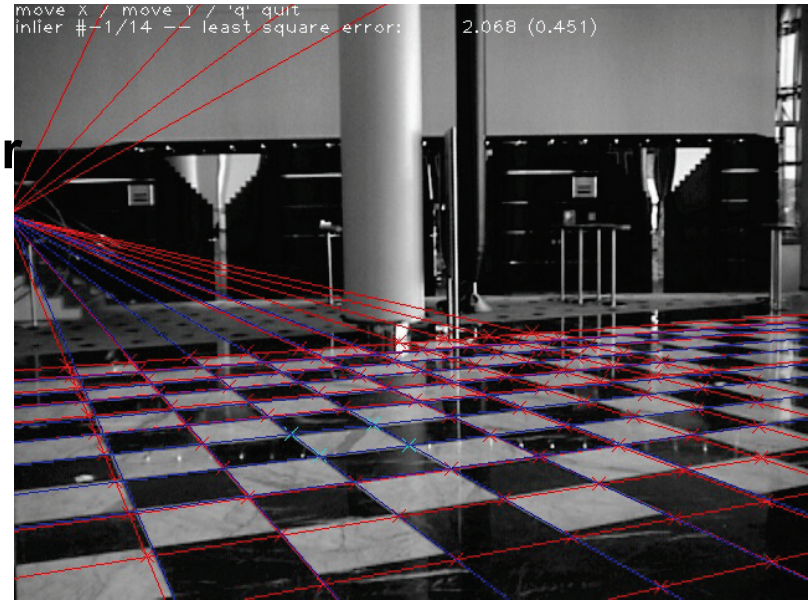
Algorithmen

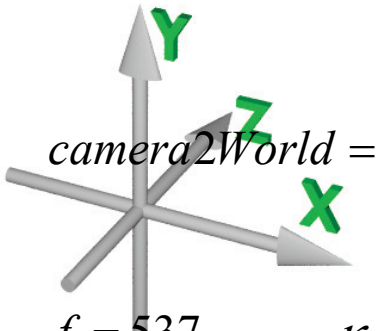
Kalibrierung (Least Squares)

- **Bestimme wahrscheinlichste Parameter gegeben die Messungen arg max$_a$ P(a|y$_\bullet$).**
- **minimiere Summe der quadratischen Abstände zu extrahierten Punkten.**

$$\arg\max_x P\left(X = x \middle| Z = z\right)$$

$$= \arg\min_x \sum_i \frac{\left(z_i - f_i(x)\right)^2}{\sigma_i^2}$$





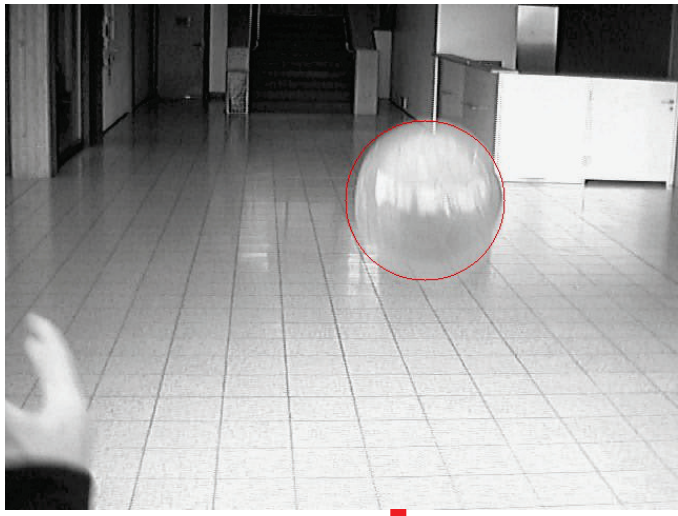
$$camera2World = \begin{pmatrix} 0.4 & 0.3 & -0.4 & 1.37 \\ -0.5 & 0.2 & 0.3 & 0.75 \\ 0.1 & -0.8 & 0.6 & 3.2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$f = 537, \qquad \kappa = 0.003$

Algorithmen

Partikel Filter

- **Was? Zustand (z.B. Position und Geschwindigkeit bei einem Ball) fortlaufend aktualisiert wenn neue Messungen ankommen.**
- **Woraus? Bildmerkmale (oder allgemeine Messungen), Dynamikmessungen (z.B. Bewegungs-kommando) alles fortlaufend.**
- **a-posteriori Verteilung $P(X|z_t,u_{t-1},z_{t-1},\ldots)$ durch gewichtete Partikel repräsentieren.**



$$p = \begin{pmatrix} 1.37 \\ 2.41 \\ 0.99 \\ 1 \end{pmatrix}, v = \begin{pmatrix} 0.4 \\ 3.4 \\ 2 \\ 0 \end{pmatrix}$$

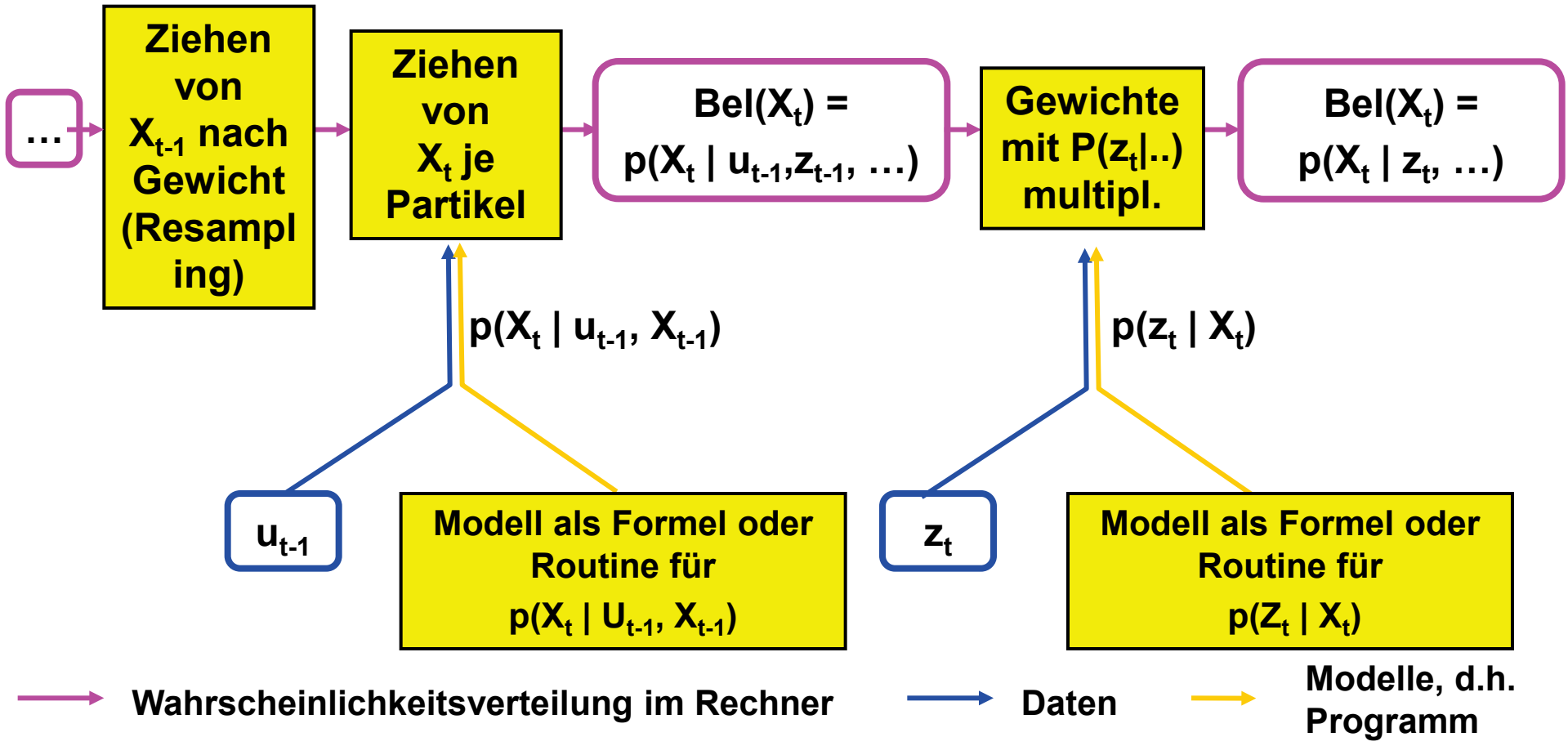
Algorithmen

Hintergrund: Rekursiver Bayes'scher Filter

$$Bel(x_t) = \eta\, p(Z_t = z_t \mid X_t = x_t) \int p(X_t = x_t \mid x_{t-1} = x_{t-1}, U_{t-1} = u_{t-1})\, Bel(x_{t-1})\, dx_{t-1}$$

- **Ein Zyklus:**
  - ziehen aus Bel(X_{t-1}) (resampling) gemäß Gewicht als Wahrscheinlichkeit, neues Gewicht 1.
  - ziehen aus P(X_t=x_t|X_{t-1}=x_{t-1},U_t=u_t), d.h. anwenden des Zustandsübergangsmodells mit zufälligen Rauschen.
  - multiplizieren des Gewichtes mit P(Z_t=z_t|X_t=x_t) d.h. gemäß Messmodell
- **Initialisierung besser auf Grund der ersten paar Messungen statt gleichverteilt.**

Algorithmen



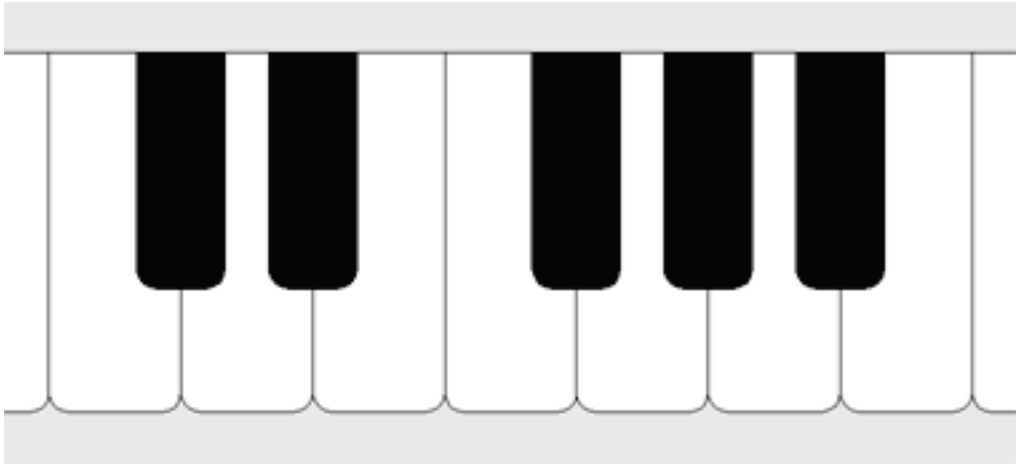
Anwendungen

Anwendungen

- **In dem Schnellrestaurant McFuture sollen die Tablettwagen auf Knopfdruck von ihrem Platz im Restaurant in die Küche und zurück fahren. Dazu sind sie mit Antrieb und einer an die Decke schauenden Kamera ausgestattet. Die Decke kann mit beliebigen Markierungen versehen werden und von dem Erkennen von Hindernissen sehen wir todesmutig ab. Wie könnte man das Problem lösen?**
- **Ein Kasino in Las Vegas bietet das Spiel Kniffel an, bei dem 5 Würfel geworfen werden. Zur statistischen Auswertung sollen die Würfel auf dem Tisch von einer Kamera beobachtet und die Augenzahl erkannt werden. Wie könnte man das Problem lösen?**

Anwendungen

- **Zu Ehren von Johann Sebastian Bach ("Das wohltemperierte Klavier") soll eine Aktionskunst Installation im Haus der Musik erfolgen. Auf den Boden ist eine große Klaviatur gemalt, die von einer Deckenkamera beobachtet wird. Steht eine Person auf einer Taste soll der entsprechende Akkord erklingen. Wie könnte man dies realisieren?**



Anwendungen

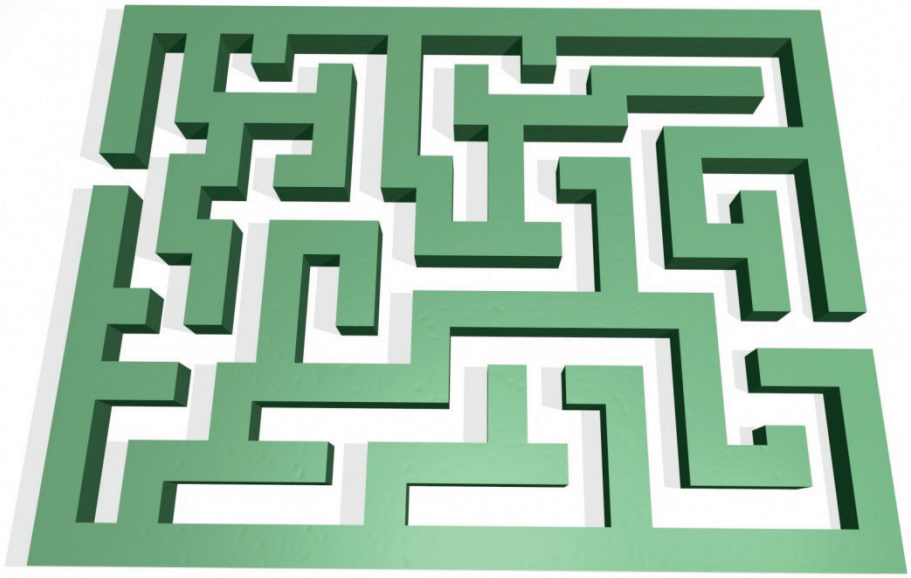
- **Auf einem Förderband werden quaderförmige Pakete unterschiedlicher Länge und Breite aber fester Höhe 20cm angeliefert. Ein Roboterarm soll die Pakete greifen und auf einer Palette stapeln. Wie könnte man das Problem angehen? Nur die grobe Idee.**
- **In einem soziologischen Experiment soll das Verhalten von Gästen einer Party erforscht werden. Dazu trägt jeder Gast einen speziell gestalteten Hut und die ganze Gesellschaft wird von einer Deckenkamera beobachtet. Per Bildverarbeitung soll die Position jedes Gastes bestimmt werden. Wie sollte man die Hüte gestalten und wie könnte die Bildverarbeitung vorgehen?**

Anwendungen

- **Ein zweibeiniger Laufroboter soll eine Treppe ersteigen. Der Roboter ist mit einer Kamera im Kopf ausgestattet, mit der er die Treppe erkennen soll. Bekannt ist die Lage der Kamera in Bezug auf die Füße (Kamerakalibrierung) und die Masse der Treppenstufen (30cm*20cm). Zu ermitteln ist, wo der Roboter auf der Treppe steht um dann den nächsten Schritt zu bestimmen. Wie könnte man dies Problem mit Bildverarbeitung lösen?**

Anwendungen

- **Ein kleiner mobiler Roboter soll autonom den Weg durch ein Labyrinth finden. Das Labyrinth besteht aus würfelförmigen Blöcken (20cm*20cm*20cm). Die Kamera ist vorne am Roboter in bekannter Lage zum Boden montiert. Wie könnte man mit Bildverarbeitung die unmittelbare Umgebung des Roboters erkennen und die Position im Labyrinth verfolgen?**



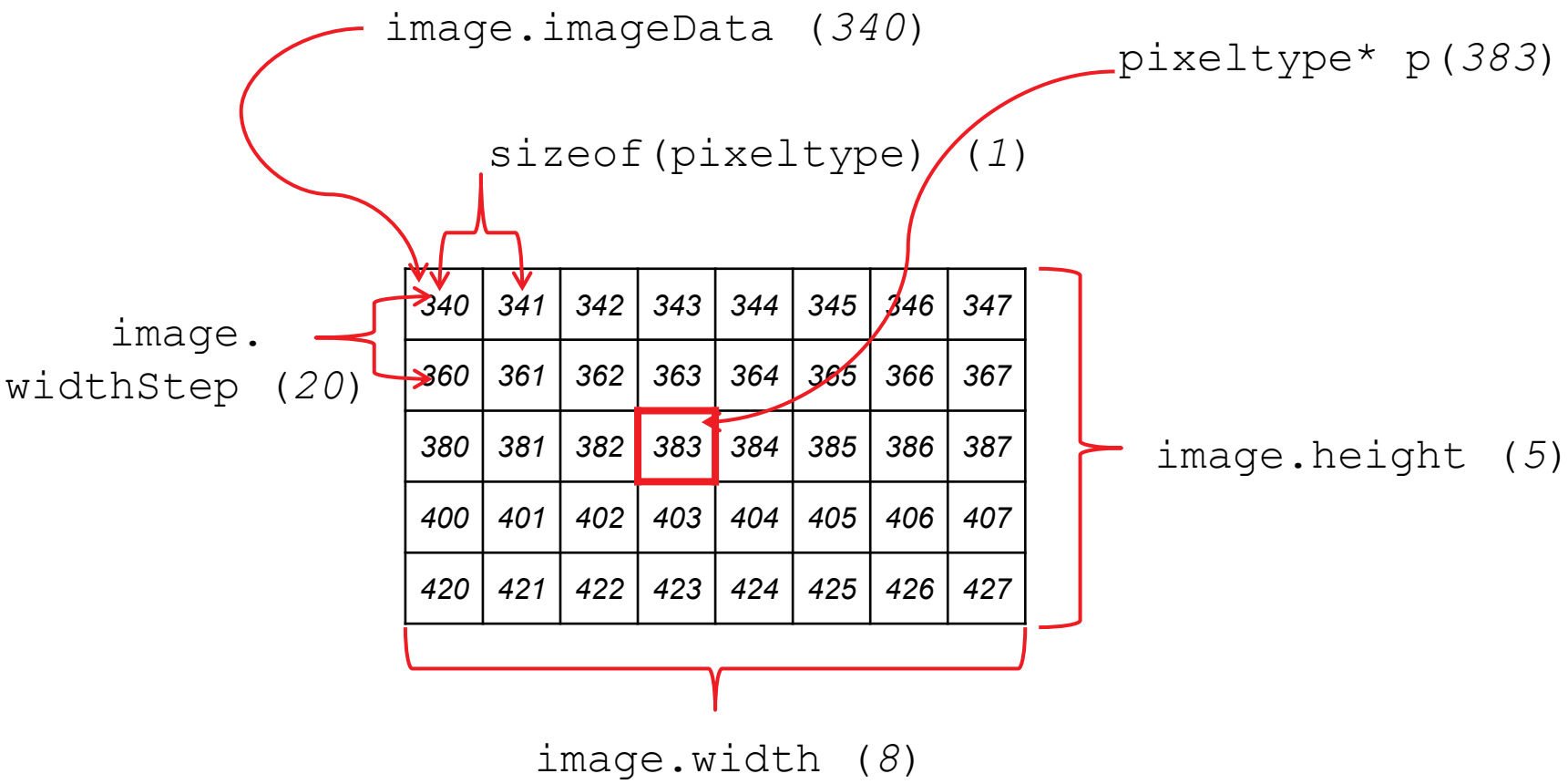
Anwendungen

- In einem Robotermuseum soll ein Roboter einem Museumsangestellten folgen, nicht aber den Besuchern. Wie könnte man das mit Bildverarbeitung realisieren? Wie könnte man den Angestellten kleiden um sich das Problem zu erleichtern?
- Mit Bildverarbeitung soll die Schwingungsperiode eines Uhrpendels bestimmt werden (also wie lange ein Pendelschlag dauert). Wie könnte man das mit Bildverarbeitung realisieren? Wie kann man die Genauigkeit der geschätzten Schwingungsperiode trotz kleinen Fehlern in der Bildverarbeitung verbessern?

Echtzeitimplementierung

Echtzeitimplementierung

Aufbau eines Bildes im Speicher



Echtzeitimplementierung

Echtzeitimplementierung der Houghtransformation für Linien

- nur Winkel, die ungefähr der Richtung des Sobel Vektors entsprechen im Houghraum erhöhen.
- α in 256 Schritte [0..π) diskretisieren, Periodizität beachten.
- *d* in Bezug auf Bildmitte um Houghraum klein zu halten.
- Sobel Betrag und diskretisiertes & normalisiertes Winkel-intervall für jede SobelX, SobelY Kombination vortabellieren (LUT1).
- cos(α), sin(α) für jeden diskretisierten Winkel tabellieren (*256 für Festkommaarithmetik) (LUT2).
- konstante Offsets aus Berechnungen herausziehen.
- Tabellen- / Houghraumdimension als Zweierpotenz, dadurch Multiplikation per <<.
- mehrstufige Pointerketten: Zwischenpointer speichern.

Echtzeitimplementierung

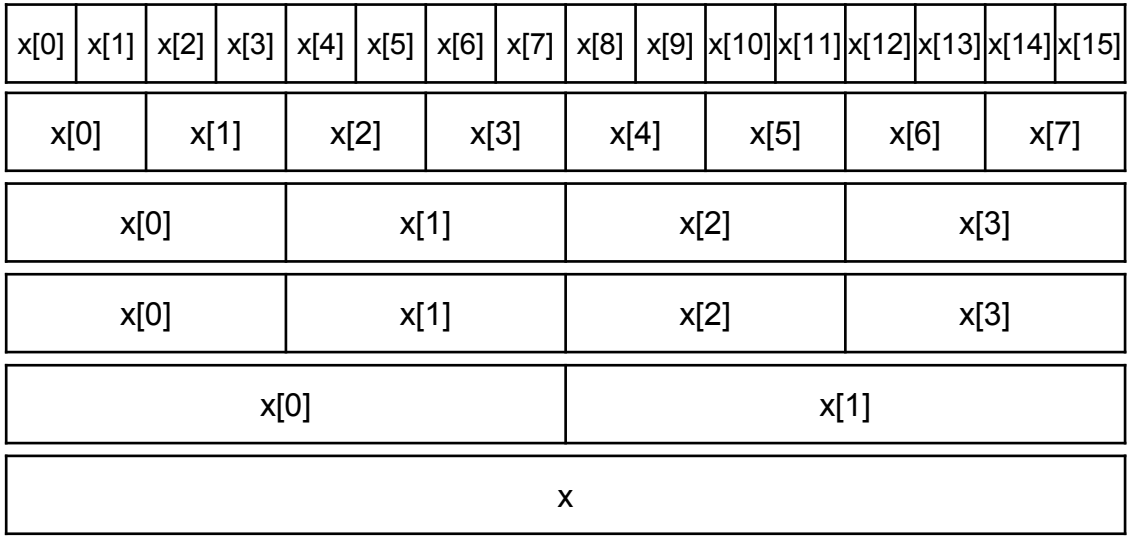
Multicore-Parallelisierung

- für einfach parallelisierbare Programme (z.B. Filter)
- serielles Programm schreiben
- mit Sonderanweisungen `#pragma` dem Compiler sagen, was parallel laufen kann
- `#pragma omp parallel`
- `#pragma omp for`
- `#pragma omp critical`
- Einsatz in der Y-Schleife einer Bildverarbeitungsroutine

Echtzeitimplementierung

SIMD-Parallelisierung

- **Befehl wendet die selbe Operation auf mehrere Werte gleichzeitig an**
- **Interpretiert als**
  - (unsigned) char x[16]
  - (unsigned) short x[8]
  - (unsigned) int x[4]
  - float x[4]
  - double x[2]
  - 128 Bits



- **Verfügbare Befehle:**
  - `+, -, *, min, max, &, |, <<, >>,` permutieren, konvertieren
  - ausserdem für `float`: `/, 1/, sqrt(float), 1/sqrt()`
  - keine Bedingungen möglich (manchmal umgehbar)
  - In C Datentypen `__mm128`,.. mit intrinsic-Funktionen `_mm_operation_typ`

Das Ende

- **Viel Erfolg**

- **Fragen oder Fehler im Skript bitte per Email melden!**