64-424
Intelligente Roboter

http://tams.informatik.uni-hamburg.de/lectures/2011ws/vorlesung/ir

Jianwei Zhang

Universität Hamburg
Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik
Technische Aspekte Multimodaler Systeme

Wintersemester 2011/2012

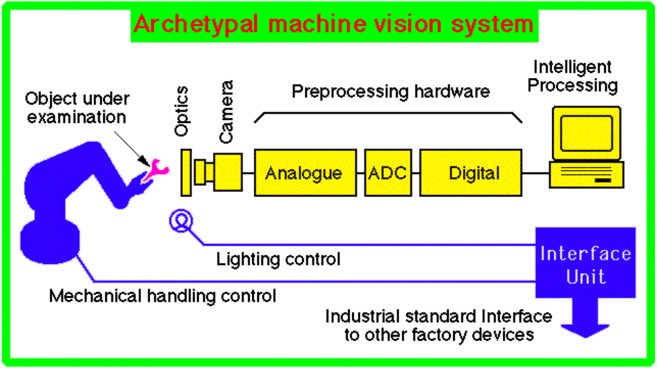
Gliederung

Gliederung (cont.)

Sichtsysteme in der Robotik

- Linearkamera (z.B. Barcode-Scanner)
- analoge CCD-Kamera (schwarz/weiß)
- analoge CCD-Farbkamera (1 Chip oder 3 Chips)
- High-Dynamic-Range CMOS-Kamera
- Digitalkameras (USB oder Firewire)
- Kamera + strukturiertes Licht
  (Laser, Farbstreifen oder -muster)
- Stereosysteme
- *catadioptrische* Systeme (z.B. omnidirektionale Sichstysteme)
  - *dioptrics* $\rightarrow$ Linsen
  - *catoptrics* $\rightarrow$ Spiegel

Sichtsysteme und Manipulation



Sichtsysteme in industriellen Roboteranwendungen

- Greifen von Objekten
  - separat oder geordnet liegende Werkstücke
  - wahllos durcheinander liegende Werkstücke („Bin-Picking", „Griff in die Kiste")
- Handhabung von Objekten
  - Veredeln, Abdichten, Entgraten, Schneiden, Binden, Verpacken, ...
  - Inspektion während der Fertigung
- Montage
  - Punkt- und Bogenschweißen
  - Schrauben, Stecken
  - Befestigen, Kleben
  - ...

Sichtsysteme in der kognitiven Robotik

Erkennen von

- Objekten
  - statisch: Typisieren, Suchen, Indizieren, ...
  - dynamisch: Bewegungen, Detektion von Fehloperationen, ...
- Menschen
  - Gesicht
  - Blickrichtung
  - Gestik
  - Operationsvorgang (einzelne Operationen, Reihenfolge, ...)

Sichtsysteme in der kognitiven Robotik (cont.)

Weltmodellierung:

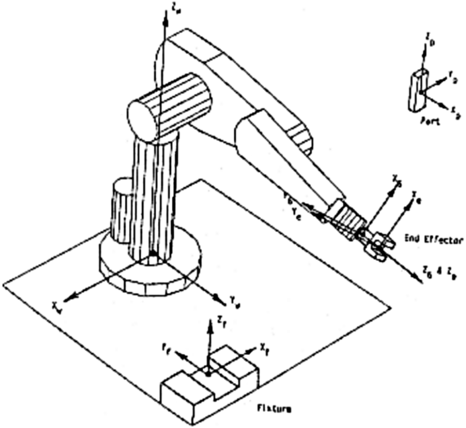
- Ortung von Objekten
- 3D-Rekonstruktion
- Belegung der Umwelt
- Lage/Orientierung des Roboters
  - relativ
  - absolut
  - ...bezogen auf verschiedene Koordinatensysteme

Sichtsysteme in der kognitiven Robotik (cont.)

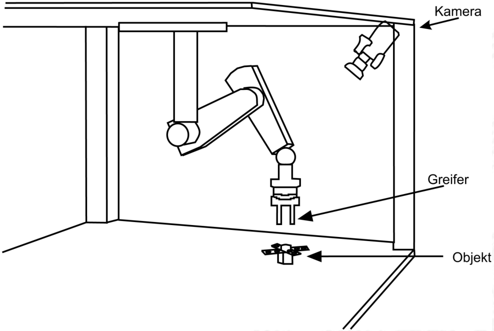
Vom Sichtsystem geführte Bewegungen:

- Visual-Servoing
  - Grob- und Feinpositionierung
  - Verfolgung von beweglichen Objekten
  - Schwingen, Jonglieren, Balancieren, ...
- Kollisionsvermeidung
  - basierend auf dem Prinzip des optischen Flusses
  - 3D-Abstandsmessungen
- Koordination mit anderen Robotern und/oder Menschen
  - Intentionserkennung
  - Bewegungsschätzung

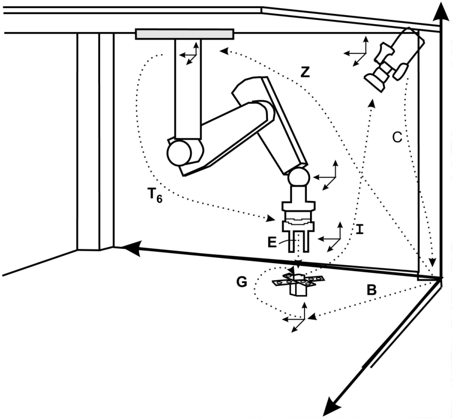
Koordinatensysteme eines Manipulationssystems



Roboter - Tisch - Kamera



Transformationen



Transformationen (cont.)

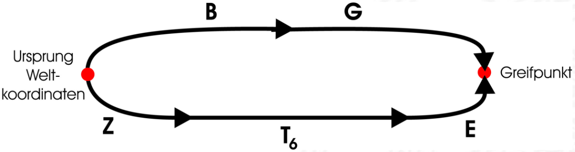
Punkte eines Koordinatensystems lassen sich über Transformationen in ein anderes Koordinatensystem überführen:

- Z: Transformation von Welt- in Manipulatorbasiskoordinaten
- T_6: komplette kinematische Transformation von der Manipulatorbasis bis zum Manipulatorende für einen Roboterarm mit sechs Freiheitsgraden
- E: Transformation vom Manipulatorende zum Greifer
- B: Transformation von Welt- zu Objektkoordinaten
- G: Beschreibung der Greifposition/-orientierung in Objektkoordinaten

Transformationen (cont.)

Greift der Roboter ein Objekt, lässt sich die Weltkoordinate des Greifpunktes auf zwei Arten bestimmen und es ergibt sich:

$$ZT_6E = BG$$



Transformationen (cont.)

Soll der Manipulator zur Greifposition bewegt werden, muss folgende Gleichung gelöst werden:

$$T_6 = Z^{-1} B G E^{-1}$$

Um die Position des Objektes nach vollendeter Greifoperation zu bestimmen, berechnet man:

$$B = Z T_6 E G^{-1}$$

Transformationen (cont.)

Eine Kamera im System liefert zwei weitere Transformationen:

C: Transformation von Kamera- in Weltkoordinaten
(Bestimmung der Transformation *off-line* durch Kalibrierung der Kamera)

I: Transformation vom Koordinatensystem im Greifpunkt zum Kamerakoordinatensystem
(Greifpunkt wird mit Methoden der Bildverarbeitung bestimmt)

Transformationen (cont.)

Die Transformation P vom Greifpunkt zum Weltkoordinatensystem ist dann:

$$P = I\ C$$

Die Kamera-Welt-Kalibrierung kann aus folgender Gleichung abgeleitet werden:

$$C = I^{-1}\ P$$

Homogene Koordinaten

- *Homogene Koordinaten* sind bekannt aus der Computergrafik um Probleme in der Matrixberechnung zu umgehen
- Ein n-dimensionaler Raum wird mit $n+1$ Dimensionen dargestellt
- Aus (x, y, z) wird (hx, hy, hz, h), wobei h eine beliebige Zahl ist
- In der Robotik ist $h = 1$, was einer direkten Projektion zwischen n- und $(n+1)$-dimensionalem Raum entspricht
- h ist eine Art Skalierung

Homogene Transformationen

- Eine 3×3-Matrix kann eine Rotation, Skalierung oder Scherung beschreiben aber keine Verschiebung (Translation)
- Dies kann umgangen werden, indem eine weitere Spalte zur Matrix hinzugefügt wird
- **Aber:** Die Matrix ist dann nicht mehr invertierbar
- Man definiert daher eine 4×4-Matrix um sowohl Rotation, Translation, Scherung, Projektion, lokale und komplette Skalierung zu beschreiben

Homogene Transformationen (cont.)

$$T = \left[\begin{array}{c|c} \mathit{Rotation} & \mathit{Translation} \\ \mathit{Scherung} & \\ \mathit{lokale\ Skalierung} & \\ \hline \mathit{Projektion} & \mathit{Skalierung} \end{array}\right]$$

$$= \left[\begin{array}{c|c} 3 \times 3 & 3 \times 1 \\ \hline 1 \times 3 & 1 \times 1 \end{array}\right]$$

Homogene Transformationen (cont.)

In der Robotik ist man nur an Rotation und Translation interessiert
Daher ergibt sich:

$$T = \left[\begin{array}{c|c} \mathit{Rotation} & \mathit{Translation} \\ ------ & ------- \\ 0 \;\; 0 \;\; 0 & 1 \end{array}\right]$$

Translation

$$Trans(p_x, p_y, p_z) = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um x-Achse

$$Rot_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um y-Achse

$$Rot_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um z-Achse

$$Rot_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Kamera-Kalibrierung

Kamera-Kalibrierung im Kontext der drei-dimensionalen maschinellen Bildverarbeitung ist die Bestimmung der intrinsischen und/oder extrinsischen Kamera-Parameter

Intrinsische Parameter: Interner geometrischer Aufbau und optische Eigenschaften der Kamera

Extrinsische Parameter: Drei-dimensionale Position und Orientierung des Koordinatensystems der Kamera relativ zu einem Weltkoordinatensystem

Kamera-Kalibrierung (cont.)

Welche Information erhält man?

Um 3D-Objekte aus zwei oder mehr Bildern zu generieren, ist es notwendig die Beziehung zwischen dem Koordinatensystem des 2D-Bildes und dem des 3D-Objektes zu kennen

Die Beziehung zwischen 2D und 3D kann durch zwei Transformationen beschrieben werden

Kamera-Kalibrierung (cont.)

1. **Perspektivische Projektion eines 3D-Punktes auf einen 2D-Bildpunkt**
   - Mit der Schätzung eines 3D-Objektpunktes und der dazugehörigen Fehler-Kovarianzmatrix kann die Projektion auf ein Bild vorhergesagt werden
2. **Rückprojektion eines 2D-Punktes auf einen 3D-Strahl**
   - Sei ein 2D-Bildpunkt gegeben, dann gibt es einen Strahl im dreidimensionalen Raum auf dem der korrespondierende 3D-Objektpunkt liegt
   - Gibt es zwei oder mehr Ansichten des 3D-Punktes, kann dessen Koordinate mit Hilfe der *Triangulation* bestimmt werden

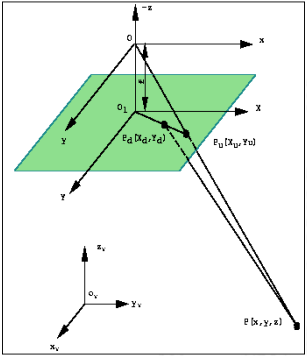
Kamera-Kalibrierung (cont.)

- Die erste Transformation ist nützlich um den Suchraum beim Merkmalsvergleich oder bei der Hypothesen-Verifikation in der Szenenanalyse zu verringern
- Die zweite Transformation ist hilfreich um 3D-Information aus Merkmalen in 2D-Bildern abzuleiten
- Verschiedene Anwendungen für die Transformationen:
  - Automatische Montage
  - 3D-Messtechnik
  - Roboter-Kalibrierung
  - Tracking
  - Trajektorien-Analyse
  - automatische Fahrzeugführung

Kalibrierungsmethoden

- Die Kamera-Kalibrierung kann *on-line* oder *off-line* erfolgen
- Kalibrierungsobjekt:
  - Identifikation der Kamera-Parameter
  - Direkte Erstellung der Koordinaten-Transformation zwischen Kamera- und Weltkoordinaten
- Selbst-Kalibrierung
- Methoden des maschinellen Lernens

Modell einer Kamera ohne Verzeichnung



Lochkameramodell mit und ohne radiale Linseverzeichnung

Modell einer Kamera ohne Verzeichnung (cont.)

- (x_w, y_w, z_w): 3D-Weltkoordinatensystem mit Ursprung O_w
- (x, y, z): 3D-Koordinatensystem der Kamera mit Ursprung O (optisches Zentrum)
- (X, Y): 2D-Bildkoordinatensystem mit Ursprung O_1
- f: Brennweite der Kamera

Transformation von Welt- in Kamerakoordinaten

- Sei $P(x_w, y_w, z_w)$ ein Punkt im Weltkoordinatensystem
- Seine Projektion auf die Bildebene kann wie folgt ermittelt werden:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + t$$

$$\text{mit} \quad R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \quad \text{und} \quad t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

- Die Parameter R und t sind die *extrinsischen* Parameter

Projektion von Kamera- auf Bildkoordinaten

- Punkt P wird auf die analoge (ideale) Bildkoordinate (u, v) projiziert
- *Perspektivische Projektion* mit Brennweite f:

$$u = f\frac{x}{z} \qquad v = f\frac{y}{z}$$

- Die Bildkoordinate (X, Y) errechnet sich aus (u, v) mit:

$$X = s_u u \qquad Y = s_v v$$

- Die Skalierungsfaktoren s_u und s_v rechnen die analogen Koordinaten von Meter in Pixel um
- s_u, s_v und f sind die *intrinsischen* Kamera-Parameter

Projektion von Welt- auf Bildkoordinaten

- Da nur zwei unabhängige intrinsische Parameter existieren, definiert man:

$$f_x \equiv f s_u \quad \text{und} \quad f_y \equiv f s_v$$

- Diese Gleichungen liefern das verzeichnungsfreie Kameramodell:

$$X = f_x \frac{r_1 x_w + r_2 y_w + r_3 z_w + t_x}{r_7 x_w + r_8 y_w + r_9 z_w + t_z}$$

$$Y = f_y \frac{r_4 x_w + r_5 y_w + r_6 z_w + t_y}{r_7 x_w + r_8 y_w + r_9 z_w + t_z}$$

Pixelkoordinaten im Rechner

- Von den im Rechner gespeicherten Bildkoordinaten (X_f, Y_f) wird die Koordinate (C_x, C_y) des Bildmittelpunktes abgezogen
- Damit ergibt sich:

$$X = X_f - C_x$$
$$Y = Y_f - C_y$$

- Die Unsicherheit über den Bildmittelpunkt kann 10-20 Pixel erreichen

Kalibrierung einer Kamera: Grundkonzept

Das Lochkamera-Modell liefert für die Kalibrierung

- die drei unabhängigen extrinsischen Parameter von R
- die drei unabhängigen extrinsischen Parameter von t
- die intrinsischen Parameter f_x, f_y, C_x und C_y

Kalibrierungspunkte

Die Kalibrierung erfolgt mit einer Menge von m Objektpunkten, die

1. bekannte Weltkoordinaten $\{x_{w,i}, y_{w,i}, z_{w,i}\}$, $i = 1, \dots, m$ in hinreichend genauer Präzision haben
2. innerhalb des Sichtfeldes der Kamera liegen

Diese *Kalibrierungspunkte* werden im Kamerabild mit ihren respektiven Kamerakoordinaten $\{X_i, Y_i\}$ detektiert

Kalibrierung

- Das Problem bei der Kalibrierung einer Kamera ist die Identifikation der unbekannten Koeffizienten des Kameramodells
- Die Bestimmung für das verzeichnungsfreie Kameramodell liefert explizit die Position der Kamera in Weltkoordinaten
- Die grundlegendste Strategie für eine Kamerakalibration ermittelt die Koeffizienten mit Hilfe der *linear-least-squares*-Identifikation der im folgenden vorgestellten *perspektivischen Transformationsmatrix* (engl. *Perspective Transformation Matrix*)

Verzeichnungsfreies Kameramodell

Das verzeichnungsfreie Kameramodell

$$X = f_x \frac{r_1 x_w + r_2 y_w + r_3 z_w + t_x}{r_7 x_w + r_8 y_w + r_9 z_w + t_z}, \qquad Y = f_y \frac{r_4 x_w + r_5 y_w + r_6 z_w + t_y}{r_7 x_w + r_8 y_w + r_9 z_w + t_z}$$

lässt sich umschreiben zu

$$X = \frac{a_{11} x_w + a_{12} y_w + a_{13} z_w + a_{14}}{a_{31} x_w + a_{32} y_w + a_{33} z_w + a_{34}}$$

$$Y = \frac{a_{21} x_w + a_{22} y_w + a_{23} z_w + a_{24}}{a_{31} x_w + a_{32} y_w + a_{33} z_w + a_{34}}$$

Perspektivische Transformationsmatrix

- Es kann $a_{34} = 1$ gesetzt werden, da eine Skalierung der Koeffizienten $a_{11}, \ldots, a_{34}$ die Werte von X und Y nicht ändert
- Die Koeffizienten $a_{11}, \ldots, a_{34}$ korrespondieren mit der so genannten *perspektivischen Transformationsmatrix*
- Die vorangegangenen beiden Gleichungen können im folgenden Identifikationsmodell zusammengefasst werden:

$$\begin{bmatrix} x_w & y_w & z_w & 1 & 0 & 0 & 0 & 0 & -Xx_w & -Xy_w & -Xz_w \\ 0 & 0 & 0 & 0 & x_w & y_w & z_w & 1 & -Yx_w & -Yy_w & -Yz_w \end{bmatrix} \begin{bmatrix} a_{11} \\ \vdots \\ a_{33} \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix}$$

Least Squares

- Die elf unbekannten Koeffizienten $a_{11}, \ldots, a_{33}$ werden mit Hilfe der Methode der kleinsten Quadrate bestimmt
- Minimal sind **sechs** Kalibrierungspunkte notwendig
- Jedes Paar Datenpunkte $\{(x_{w,i}, y_{w,i}, z_{w,i}), (X_i, Y_i)\}$ liefert zwei algebraische Gleichungen mit den gesuchten Koeffizienten
- Es kann gezeigt werden, dass die Kalibrierungspunkte **nicht koplanar** sein dürfen
- Ist dies nicht der Fall, ist die erste Matrix im Identifikationsmodell singulär, da die Spalten 3 und 4 sowie 7 und 8 linear abhängig sind

Probleme

- Die vorgestellte Lösung ist noch nicht global optimal, da bisher keine Linsenverzeichnung berücksichtigt wurde
- Es ist nicht möglich explizit die Rotationsmatrix R und den Translationsvektor t zu bestimmen
- Das bedeutet die vorgestellte Kalibrierung ermöglicht **nicht** die Nutzung einer Kamera, die an einem sich bewegenden Roboterarm montiert ist
- Die Herstellung eines präzisen 3D-Kalibrierungsaufbaus ist aufwendiger als die eines 2D-Kalibrierungsobjekts

Stereo-Vision

- Die bisher vorgestellte Kalibrierungsmethode ermöglicht allerdings eine schnelle, wenn auch unpräzise Messung von Punkten mit einem Stereo-Kamera-Aufbau
- Dazu werden zwei Kameras A und B kalibriert und liefern die Kalibrationsvektoren a^A und a^B
- Dann kann die Koordinate $\{x_w, y_w, z_w\}$ eines jeden Punktes der von beiden Kameras gesehen wird berechnet werden
- Jeder unbekannte Punkt hat die korrespondierenden Bildkoordinaten $\{X^A, Y^A\}$ und $\{X^B, Y^B\}$

Stereo-Vision (cont.)

Mit der Gleichung

$$\begin{bmatrix} a_{11} - a_{31}X & a_{12} - a_{32}X & a_{13} - a_{33}X \\ a_{21} - a_{31}Y & a_{22} - a_{32}Y & a_{23} - a_{33}Y \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} X - a_{14} \\ Y - a_{24} \end{bmatrix}$$

für jede Kamera ensteht ein überbestimmtes Gleichungssystem, welches die Bestimmung der 3D-Koordinate eines Punktes aus den Bildkoordinaten erlaubt

Kameramodell mit Linsenverzeichnung

- Reale Kameras und Linsen verursachen eine Vielzahl von Abbildungsfehlern und genügen nicht dem Modell
- Die Hauptfehlerquellen sind:
  1. Räumliche Auflösung relativ gering, da die Auflösung der Kameras ebenfalls noch gering ist (Aktuelle DV-Kameras: 320x200, 640x480 @30 fps; 800x600, 1024x768 @15 fps; 1280x960 @7.5 fps)
  2. Die meisten (billigen) Linsen sind unsymmetrisch und erzeugen Verzerrungen
  3. Der Zusammenbau der Kamera ist nicht präzise durchführbar (Mittelpunkt des CCD-Chips liegt nicht auf der optischen Achse; Chip liegt nicht parallel zur Linse etc.)
  4. Timing-Fehler zwischen Kamera-Hardware und Grabber-Hardware

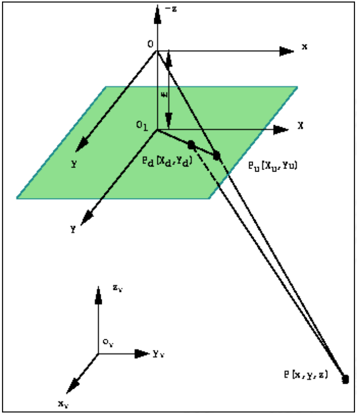
Verzeichnung

- Verzeichnung durch das Linsensystem resultiert in einer geänderten Position der Bildpixel auf der Bildebene
- Das Lochkameramodell wird dem nicht mehr gerecht
- Es wird ersetzt durch folgendes Modell:

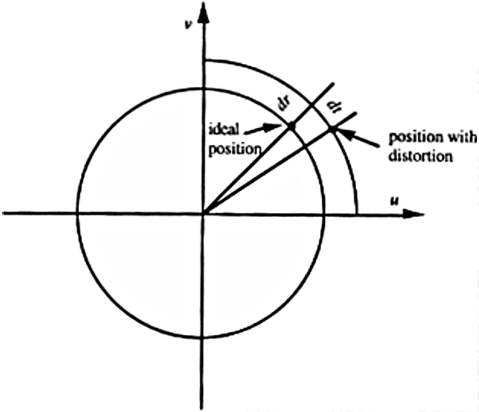
$$u' = u + D_u(u, v)$$
$$v' = v + D_v(u, v)$$

wobei u und v die nicht beobachtbaren, verzeichnungsfreien Bildkoordinaten sind und u' und v' die korrespondierenden verzerrten Koordinaten

Verzeichnung (cont.)



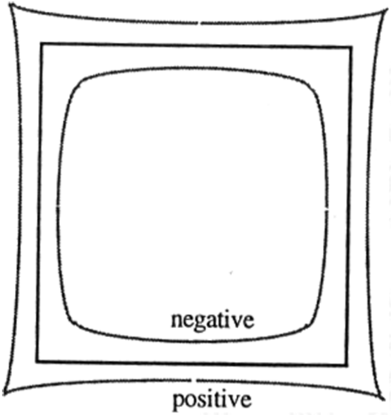
Verzeichnung (cont.)



Arten von Verzeichnungen

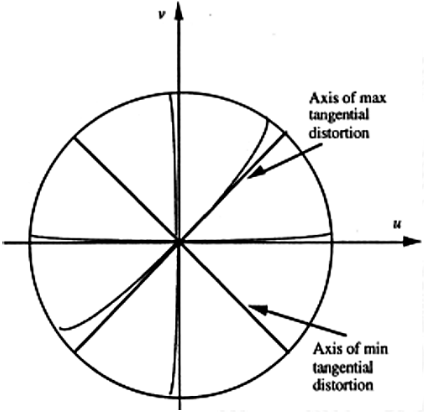
- Es gibt **zwei** Arten von Verzeichnungen:
  - *radial*
  - *tangential*
- Radiale Verzeichnung verursacht einen Versatz der idealen Position nach innen (Tonne) oder außen (Kissen)
- Ursache: fehlerhafte radiale Krümmung der Linse

Radiale Verzeichnung



gerade Linien → keine Verzeichnung

Tangentiale Verzeichnung



gerade Linien → keine Verzeichnung

Modellierung der Linsenverzeichnung

- Nach Weng et. al. (1992) unterscheidet man drei Verzeichnungen, die modelliert werden:
  1. Radiale Linsenverzeichnung (engl. *radial distortion*)
  2. Dezentrierende Verzeichnung (engl. *decentering distortion*)
  3. Verzeichnung des dünnen Prismas (engl: *thin prism distortion*)
- Die dezentrierende Verzeichnung und die Verzeichnung des dünnen Prismas sind sowohl radial als auch tangential
- Bei der dezentrierenden Verzeichnung sind die optischen Zentren der Linsen nicht kolinear

Modell: Radiale Verzeichnung

Radiale Verzeichnung

$$D_{ur} = ku(u^2 + v^2) + O[(u, v)^5]$$

$$D_{vr} = kv(u^2 + v^2) + O[(u, v)^5]$$

Modell: Dezentrierende Verzeichnung

Dezentrierende Verzeichnung

$$D_{ud} = p_1(3u^2 + v^2) + 2p_2uv + O[(u,v)^4]$$

$$D_{vd} = 2p_1uv + p_2(u^2 + 3v^2) + O[(u,v)^4]$$

Modell: Verzeichnung des dünnen Prismas

Verzeichnung des dünnen Prismas

$$D_{up} = s_1(u^2 + v^2) + O[(u, v)^4]$$

$$D_{vp} = s_2(u^2 + v^2) + O[(u, v)^4]$$

Gesamtmodell der Linsenverzeichnung

Wir ignorieren die Terme mit Ordnung höher als 4 und fassen die vorangegangenen Modelle zusammen:

$$D_u = ku(u^2 + v^2) + (p_1(3u^2 + v^2) + 2p_2 uv) + s_1(u^2 + v^2)$$

$$D_v = kv(u^2 + v^2) + (2p_1 uv + p_2(u^2 + 3v^2)) + s_2(u^2 + v^2)$$

Vereinfachtes Modell

Da die radiale Linsenverzeichnung der dominierende Effekt ist, kann folgendes Gleichungssystem als vereinfachtes Kameramodell verwendet werden:

Vereinfachtes Kameramodell mit Verzeichnung

$$u' = u(1 + k' r'^2)$$
$$v' = v(1 + k' r'^2)$$

mit $r'^2 = u^2 + v^2$

Radialer Verzeichnungskoeffizient

Da u und v unbekannt sind, werden diese durch die messbaren Bildkoordinaten X und Y ersetzt und es gilt

$$r'^2 = (X/s_u)^2 + (Y/s_v)^2$$

Definiert man $k \equiv k' s_v^2$, den *radialen Verzeichnungskoeffizienten* (engl. *radial distortion coefficient*), dann folgt

$$\mu \equiv \frac{f_y}{f_x} = \frac{s_v}{s_u}$$

und

$$r^2 \equiv \mu^2 X^2 + Y^2$$

Modell für kleine radiale Verzeichnungen

Mit den oben genannten Modifikationen erhält man folgendes Kameramodell für kleine radiale Verzeichnungen

$$X(1 + kr^2) \cong f_x \frac{r_1 x_w + r_2 y_w + r_3 z_w + t_x}{r_7 x_w + r_8 y_w + r_9 z_w + t_z},$$

$$Y(1 + kr^2) \cong f_y \frac{r_4 x_w + r_5 y_w + r_6 z_w + t_y}{r_7 x_w + r_8 y_w + r_9 z_w + t_z}$$

Variation

Ein für das *least squares* Verfahren nützlicher Trick ist die Verwendung der folgenden Variation des vorigen Modells

$$\frac{X}{1+kr^2} \cong f_x \frac{r_1 x_w + r_2 y_w + r_3 z_w + t_x}{r_7 x_w + r_8 y_w + r_9 z_w + t_z},$$

$$\frac{Y}{1+kr^2} \cong f_y \frac{r_4 x_w + r_5 y_w + r_6 z_w + t_y}{r_7 x_w + r_8 y_w + r_9 z_w + t_z}$$

und gilt unter der Annahme, dass $kr^2 << 1$ ist

radial alignment constraint

Wenn neben der radialen keine weiteren Verzeichnungen auftreten, erhält man das *radial alignment constraint* (**RAC**)

$$\frac{X}{Y} = \mu^{-1} \frac{r_1 x_w + r_2 y_w + t_x}{r_4 x_w + r_5 y_w + t_y}$$

bzw:

$$X_d : Y_d = x : y$$

mit $X_d = f_x X$ und $Y_d = f_y Y$

Tsai's RAC-basierte Kamerakalibrierung

- Annahme C_x, C_y und μ sind bekannt
- Ziel ist die Ermittlung der extrinsischen Parameter R und t sowie der intrinsischen Parameter f_x, f_y und k
- Für die Kalibrierung wird eine Menge **koplanarer** Kalibrationspunkte verwendet werden
- Die Kalibrierung beinhaltet zwei Schritte
  1. Ermitteln der Rotationsmatrix R und der Komponenten t_x und t_y des Translationsvektors
  2. Schätzung der übrigen Parameter aufgrund der Ergebnisse des ersten Schrittes

Kamerakalibrierung nach Tsai: Schritt 1

1. Berechnung der Bildkoordinaten (X_i, Y_i)

Sei N die Anzahl der Bildpunkte, dann gilt für $i = 1, 2, \ldots, N$

$$X_i = X_{f,i} - C_x$$
$$Y_i = Y_{f,i} - C_y$$

wobei $X_{f,i}$ und $Y_{f,i}$ die Pixelwerte im Rechner sind

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

2. Bestimmung der Zwischenparameter $\{v_1, v_2, v_3, v_4, v_5\}$

- Da RAC unabhängig vom k und f ist, können R, t_x und t_y berechnet werden
- Wir definieren

$$\{v_1, v_2, v_3, v_4, v_5\} \equiv \{r_1 t_y^{-1}, r_2 t_y^{-1}, t_x t_y^{-1}, r_4 t_y^{-1}, r_5 t_y^{-1}\}$$

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

- Wenn man für den i-ten Kalibrationspunkt beide Seiten der RAC-Gleichung durch t_y teilt und den entstehenden Ausdruck umarrangiert, erhält man

$$\begin{bmatrix} x_{w,i}Y_i & y_{w,i}Y_i & Y_i & -x_{w,i}\mu X_i & -y_{w,i}\mu X_i \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{bmatrix} = \mu X_i$$

wobei $x_{x,i}$ und $y_{w,i}$ die x und y Koordinaten des i-ten Kalibrationspunktes sind

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

- Die minimale Anzahl an notwendigen **nicht kolinearen** Kalibrationspunkten ist $N = 5$
- In der Praxis sollte $N > 5$ gewählt werden
- **Bemerkung:** Falls $t_y = 0$ kann obige Gleichung auch in Abhängigkeit von t_x formuliert werden
- Erhält man $t_x = t_y = 0$, so ist der gewählte Kameraaufbau in geeigneter Form abzuändern

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

3. Berechnung von R, t_x und t_y

- Definiere $C \equiv \begin{bmatrix} v_1 & v_2 \\ v_4 & v_5 \end{bmatrix}$
- Wenn **keine** Zeile oder Spalte zu null wird, gilt:

$$t_y^2 = \frac{S_r - \sqrt{S_r^2 - 4(v_1 v_5 - v_4 v_2)^2}}{2(v_1 v_5 - v_4 v_2)} \qquad \text{mit} \qquad S_r \equiv v_1^2 + v_2^2 + v_4^2 + v_5^2$$

- Andernfalls gilt

$$t_y^2 = (v_i^2 + v_j^2)^{-1}$$

wobei v_i und v_j die Elemente aus C sind, die nicht null sind

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

- Physikalisch sollten die Vorzeichen von x und X sowie y und Y gleich sein
- Dies wird genutzt, um das Vorzeichen von t_y zu bestimmen
- Annahme: $t_y > 0$
- Berechnung von

$$
\begin{aligned}
r_1 &= v_1 t_y \\
r_2 &= v_2 t_y \\
r_4 &= v_4 t_y \\
r_5 &= v_5 t_y \\
t_x &= v_3 t_y
\end{aligned}
$$

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

- Mit einem beliebigen Kalibrationspunkt lassen sich die folgenden Koordinaten bestimmen:

$$x = r_1 x_w + r_2 y_w + t_x$$
$$y = r_4 x_w + r_5 y_w + t_y$$

- Gilt $sign(x) = sign(X)$ und $sign(y) = sign(Y)$, dann gilt die Annahme $sign(t_y) = 1$ und wir behalten r_1, r_2, r_4, r_5 und t_x
- Andernfalls setzen wir $sign(t_y) = -1$ und drehen die Vorzeichen von r_1, r_2, r_4, r_5 und t_y entsprechend um

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

- Es gibt zwei mögliche Lösungen für die Rotationsmatrix R, wenn eine 2×2-Teilmatrix bekannt ist
- Diese Lösungen sind f_x mit positivem und mit negativem Vorzeichen
- R kann wie folgt berechnet werden

$$
\begin{aligned}
r_3 &= \pm(1 - r_1^2 - r_2^2)^{1/2} \\
r_6 &= \mp sign(r_1 r_4 + r_2 r_5)(1 - r_4^2 - r_5^2)^{1/2} \\
[r_7\ r_8\ r_9]^T &= [r_1\ r_2\ r_3]^T \times [r_4\ r_5\ r_6]^T
\end{aligned}
$$

- Eine der beiden Lösungen führt zu einem positiven f_x in *Schritt 2*

Kamerakalibrierung nach Tsai: Schritt 1 (cont.)

Anmerkungen:

- Die sich ergebende Matrix R ist möglicherweise nicht orthonormal
- Es müssen daher noch Orthonormalisierungsschritte durchgeführt werden, die hier nicht weiter erläutert werden

Kamerakalibrierung nach Tsai: Schritt 2

Bestimmung der Parameter t_z, k, f_x und f_y

- Wenn R, t_x und t_y bekannt sind, können die übrigen Parameter mit folgender Gleichung für den i-ten Kalibrationspunkt bestimmt werden:

$$\begin{bmatrix} -X_i & x_i & -x_i r_i^2 \end{bmatrix} \begin{bmatrix} t_z \\ f_x \\ kf_x \end{bmatrix} = X_i w_i$$

mit

$$x_i \equiv r_1 x_{w,i} + r_2 y_{w,i} + t_x$$
$$w_i \equiv r_7 x_{w,i} + r_8 y_{w,i}$$

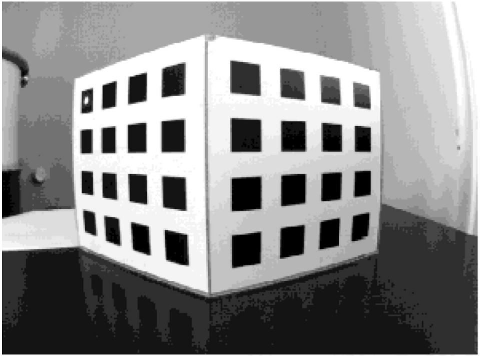
Kamerakalibrierung nach Tsai: Schritt 2 (cont.)

- Immer wenn mehr als drei Kalibrationspunkte benutzt werden, entsteht ein überbestimmtes Gleichungssystem
- Die Lösung mit Hilfe des *least-squares*-Verfahrens liefert k, t_z und f_x
- Mit f_x lassen sich die übrigen Parameter errechnen:

$$f_y = f_x \mu$$

$$k = (kf_x) f_x^{-1}$$

3D-Kalibrierungsaufbau



Typischer 3D-Kalibrierungsaufbau

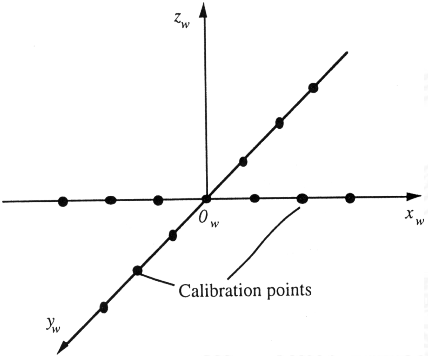
Schnelle RAC-basierte Kalibrierung

- Wenn für den erste Schritt des Tsai-Algorithmus nur Kalibrationspunkte auf der x- und y-Achse des Weltkoordinatensystems verwendet werden, vereinfacht sich die RAC-Gleichung
- Üblicherweise: Die mittlere Reihe und mittlere Spalte einer typischen Kalibrierungsplatte definieren dann die x_w- und y_w-Achse
- Beim Tsai-Algorithmus wird das *linear-least-squares*-Verfahren in Schritt eins auf fünf, in Schritt zwei auf drei Variablen angewandt
- Mit obiger Vereinfachung muss das *least-squares*-Verfahren dreimal für zwei Variablen angewandt werden

Schnelle RAC-basierte Kalibrierung (cont.)

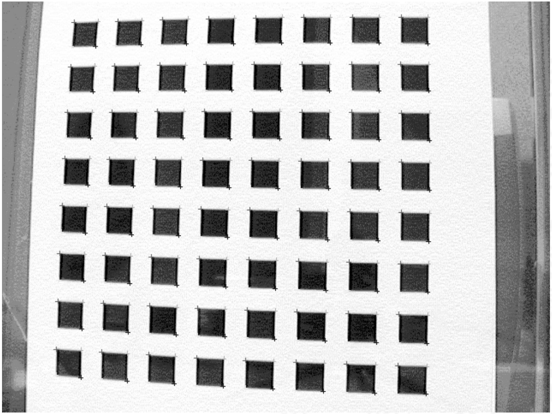
- Da es hierfür eine geschlossene Lösung gibt, reduziert sich die für die Kalibrierung notwendige Rechenzeit signifikant
- Voraussetzung für die schnelle Variante des Tsai-Algorithmus ist, das μ sowie $C_x$und C_y *a priori* bekannt sind
- Wie bei Tsai's Kalibrierung sind zwei Schritte notwendig:
  1. Verwendung von Kalibrationspunkten auf der x_w- und y_w-Achse und einer vereinfachten RAC-Gleichung, um R, t_x und t_y zu bestimmen
  2. Bestimmung der übrigen Parameter mit allen sichtbaren Kalibrationspunkten

Schnelle RAC-basierte Kalibrierung (cont.)



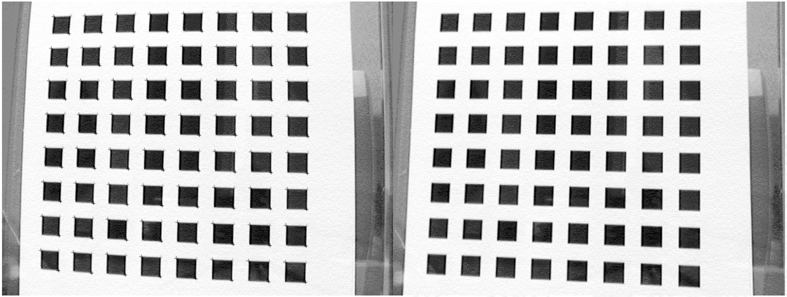
Kalibrationspunkte für die erste Phase der schnellen RAC-basierten Kalibrierung

Schnelle RAC-basierte Kalibrierung (cont.)



Typische Kalibrierungsplatte

Schnelle RAC-basierte Kalibrierung (cont.)



Mit einer kalibrierten Kamera lässt sich das Bild entzerren

Implizite Kamerakalibrierung

- Die *implizite Kamerakalibrierung* berücksichtigt alle Linsenverzeichnungen (Gesamtmodell)
- Wie bei der Kalibrierung ohne Linsenverzeichnung werden die Parameter nicht explizit bestimmt
- **Notation:**

(u^i, v^i)	Pixelkoordinaten im Bild
(x_i, y_i)	Weltkoordinaten von Punkt P auf Ebene π_i

Implizite Kamerakalibrierung (cont.)

- Projektion vom Pixel- auf das Weltkoordinatensystem auf eine Ebene π_1 ergibt sich aus dem Gesamtmodell wie folgt:

$$X_1 = \frac{\sum_{0 \leq i+j \leq 3} a_{ij}^{(1)} u_1^i v_1^j}{\sum_{0 \leq i+j \leq 3} a_{ij}^{(3)} u_1^i v_1^j}$$

$$Y_1 = \frac{\sum_{0 \leq i+j \leq 3} a_{ij}^{(2)} u_1^i v_1^j}{\sum_{0 \leq i+j \leq 3} a_{ij}^{(3)} u_1^i v_1^j}$$

wobei $a_{ij}^{(k)}$ die Transformationskoeffizienten sind

Implizite Kamerakalibrierung (cont.)

- Mit Hilfe zweier Kalibrationsebenen π_1 und π_2 können die Transformationskoeffizienten zu den beiden Ebenen bestimmt werden
- Somit ist jeder Bildpunkt auf zwei Ebenen projezierbar
- Die wirkliche Koordinate liegt auf der Verbindungsgeraden durch die beiden Punkte

Bestimmung einer Zeigerichtung: Motivation

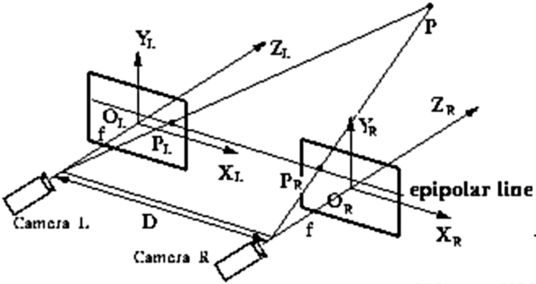
Motivation:

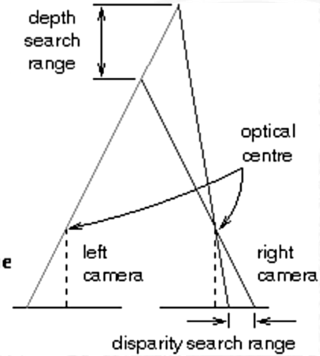
- Die Erkennung von Handgesten kann im Rahmen der Mensch-Maschine-Kommunikation genutzt werden
- Anwendungen im Bereich der virtuellen Realität, Multimedia oder Roboter-Instruktion und -Teleoperation

Lösungen:

- Sensoren an der Hand (z.B. Daten-Handschuh)
- Stereo-Vision (kalibriert)
- Stereo-Vision (unkalibriert)

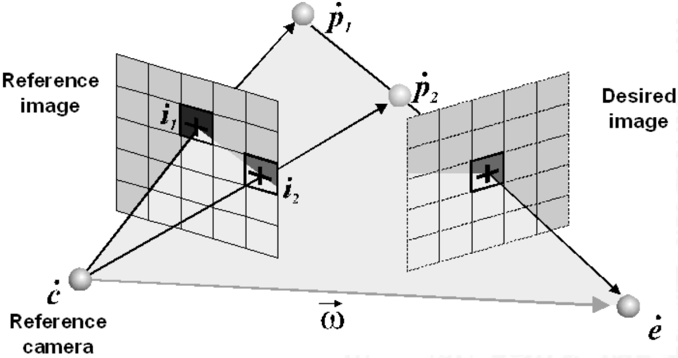
Bestimmung einer Zeigerichtung: Stereo-Vision





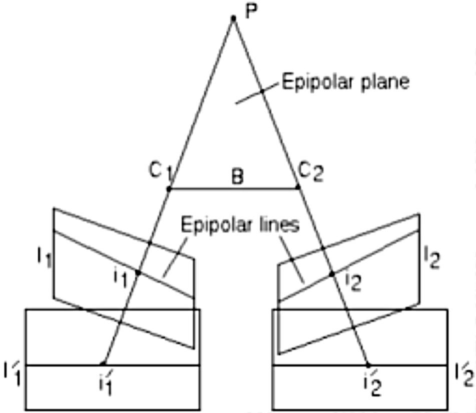
Prinzipieller Stereoaufbau mit parallelen optischen Achsen

Bestimmung einer Zeigerichtung: Epipolarlinien



Der mit einem Punkt aus Bild 1 korrespondierende Punkt kann auf der zugehörigen Epipolarlinie in Bild 2 gefunden werden

Bestimmung einer Zeigerichtung: Epipolarlinien



Bei parallelen optischen Achsen sind die Epipolarlinien horizontale Linien

Unkalibrierte Stereo-Vision

- Cipolla et. al. (1994) präsentieren eine unkalibriertes Stereo-System zur Erkennung von Zeigegesten
- **Annahme:** Lochkameramodel mit Blick auf eine Ebene
- Die Beziehung zwischen Ebenenkoordinatensystem (X, Y) und Bildkoordinatensystem (u, v) lautet:

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = T \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

wobei $T_{3\times3}$ eine homogene Matrix mit $t_{33} = 1$ ist

Unkalibrierte Stereo-Vision (cont.)

- Um T zu bestimmen müssen mindestens vier Punkte beobachtet werden
- Man definiert die Grenzen der Arbeitsebene mit $(0, 0)$,$(0, 1)$,$(1, 0)$ und $(1, 1)$
- Für die beiden Kameras werden die Transformationen T und T' bestimmt

Bestimmung des Zeigepunktes

Notation:

l_w: Längsachse des Zeigers in der Welt

l_i: Projektion von l_w auf die Bildebene

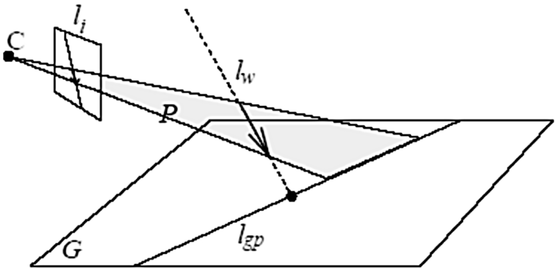
l_{gp}: Projektion von l_w auf Ebene G

Verfahren:

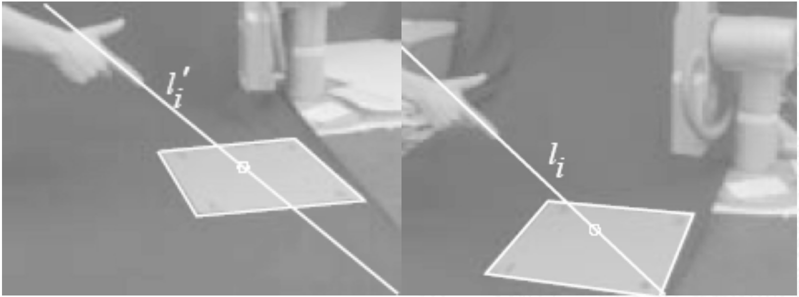
- Mit dem Bild der zweiten Kamera erhält man eine Projektion l'_{gp} deren Schnittpunkt mit l_{gp} der Zeigepunkt ist
- l_i ist das Bild von l_{gp}, d.h. $l_i = Tl_{gp}$
- Daraus folgt

$$l_{gp} = T^{-1} l_i \qquad \text{und} \qquad l'_{gp} = T'^{-1} l_i$$

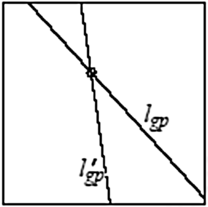
Bestimmung des Zeigepunktes (cont.)



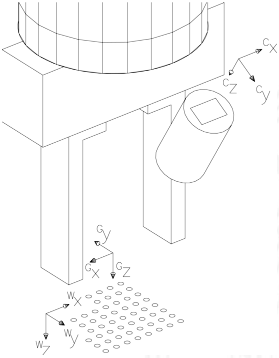
Bestimmung des Zeigepunktes (cont.)



Bestimmung des Zeigepunktes (cont.)



Handkamera-Kalibrierung



Kamera-, Greifer- und Weltkoordinatensystem

Handkamera-Kalibrierung (cont.)

Aufgabe:

Bestimmung der festen räumlichen Relation zwischen Kamera- (C) und Greiferkoordinatensystem (G) repräsentiert durch die homogene Transformation ${}^{C}H_{G}$

Idee:

Direkte Bestimmung von ${}^{C}H_{G}$ über modellbasierte Lokalisation sichtbarer Greifermerkmale

Handkamera-Kalibrierung (cont.)

Lösung:

- Positionierung des Greifers auf einem planaren Kalibrationsobjekt mit mehreren Messpunkten
- Greifer- und Weltkoordinatensystem fallen zusammen (*Ebenenkoinzidenz*)
- Ebenenkoinzidenz ermöglicht Problemkomposition

$$^{C}H_G = ^{C}H_W^{W}H_G$$

Handkamera-Kalibrierung (cont.)

Vorgehensweise:

1. Bestimmung der internen und externen Kameraparameter mittels Kalibrationsobjekt $\Rightarrow \quad {}^{C}H_{W}$
2. Bestimmung der Parameter der 2D-Transformation ${}^{W}H_{G}$ mittels der sichtbaren Greifermerkmale

Handkamera-Kalibrierung (cont.)

Vorteile der „Selbstsichtbarkeit“:

- „Selbstsichtbarkeit“ ermöglicht Kalibrierung der Konfiguration ohne Testbewegungen des Manipulators im Gegensatz zu klassischen Verfahren
- Zwei punktförmige Greifermerkmale für Bestimmung von ${}^{C}H_{G}$ ausreichend, Lösung in geschlossener Form
- Online-Überwachung der relativen Position zwischen Greifer und Objekt
- Höhere Akkuratheit der Offline-Kalibrierung durch Ausschluß kinematischer Fehler
- Höhere Robustheit durch mögliche Online-Kalibrierung

Visuell geregeltes Greifen

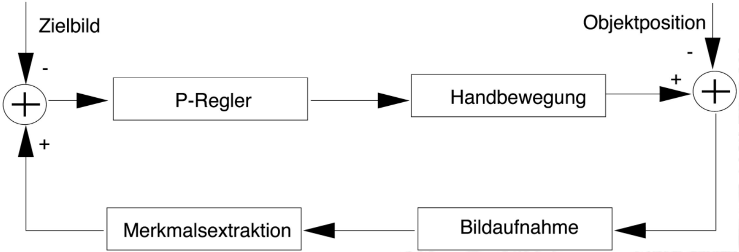
Aufgabe:

2D-Feinpositionierung einer Roboterhand oder eines Greifers bezüglich des zu greifenden Objektes

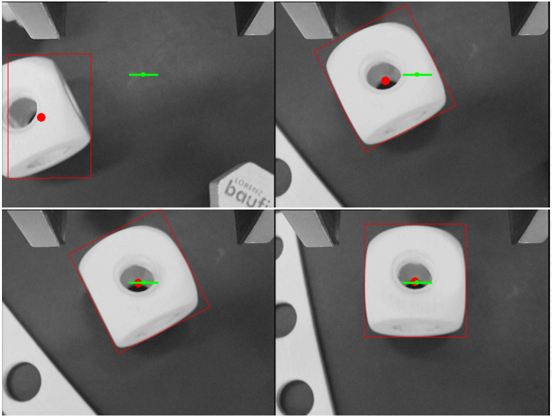
Verfahren:

1. Offline-Vorgabe der Ziellage (z.B. Objektmerkmale aus Stereo-Bildverarbeitung)
2. Online Transformation der aktuellen Differenz zur Zielvorgabe (z.B. mit Handkamera)

Visuell geregeltes Greifen (cont.)

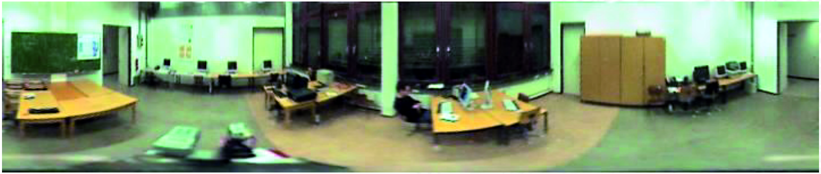


Visuell geregeltes Greifen (cont.)



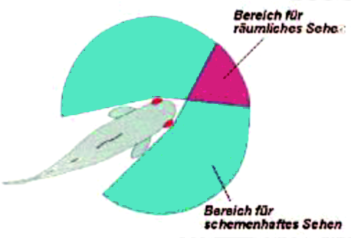
Omnidirektionale Sichtsysteme

- Generierung von Panoramabildern mit **einer** Kamera
- 360° Rundumsicht und etwa 30° vertikales Sichtfeld



Einführung

- **Biologische Inspiration:** Großes Sichtfeld vieler Tiere
- **Beispiel:** Sichtfeld eines Fisches



- **Problem:** Technische Umsetzung erfordert spezielle Optiken

Kamerasysteme mit erweitertem Blickwinkel

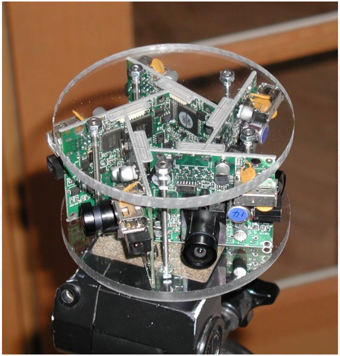
1. Verwendung mehrerer Kameras gleichzeitig

Probleme:

- unterschiedliche Verzeichnungen
- unterschiedliche Farben
- unterschiedliche Helligkeiten

Beispiel: RingCam (Microsoft-Research)

- Fünf IEEE1394-Kameras
- Auflösung von 640x480 Pixel
- Gesamtauflösung 3000x480 Pixel



http://research.microsoft.com/~rcutler/ringcam/ringcam.htm

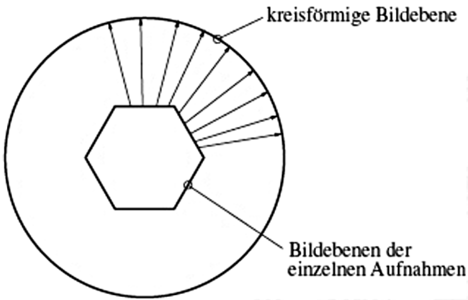
Kamerasysteme mit erweitertem Blickwinkel

2. Eine schwenkbare oder rotierende Kamera

Probleme:

- Ungenauigkeiten bei Rotation um das optische Zentrum
- zusätzlicher Energieaufwand für Rotation
- schwierige Verarbeitung der Bilder bei dynamischen Szenen
- aufwendige Umrechnungen in eine Panoramadarstellung

Kamerasysteme mit erweitertem Blickwinkel (cont.)



Kamerasysteme mit erweitertem Blickwinkel (cont.)

3. Weitwinkellinsen (Fischaugen-Objektiv)

Probleme:

- starke Verzeichnung
- teuer

4. Konvexe Spiegel und darunter angebrachte Kamera

Diese, als **catadioptrische Kamerasysteme** bezeichneten, omnidirektionalen Sichtsysteme werden im folgenden näher betrachtet. . .

Catadioptrische Kamerasysteme

Kombinationen aus Kamera, Linsen und Spiegeln nennt man catadioptrische Sichtsysteme:

- *dioptrics* $\rightarrow$ Linsen
- *catoptrics* $\rightarrow$ Spiegel

Kurze Geschichte omnidirektionaler Sichtsysteme

1970: U.S.-Patent von Rees

- Kamera und hyberboloider Spiegel

1990: Echtzeit-Verarbeitung der Bilddaten möglich

- konische, spharische und hyperboloide Spiegel

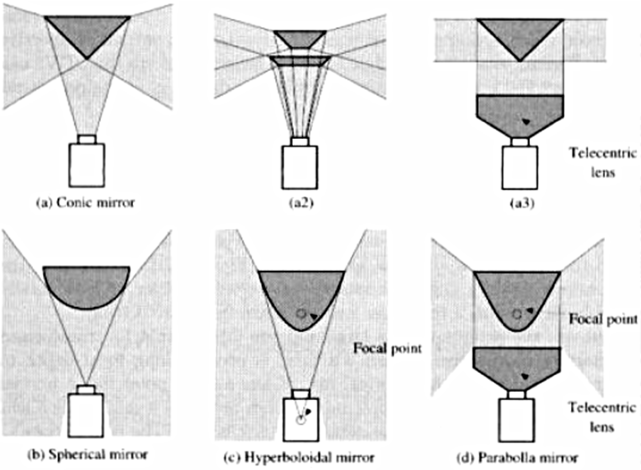
1997: Theoretische Analyse von Nayar und Baker

- neu: paraboloider Spiegel mit telezentrischer Linse

Ausführlichere Geschichte:

`http://www.math.drexel.edu/~ahicks/design/`

Spiegeldesign



Nachteile der verschiedenen Spiegelformen

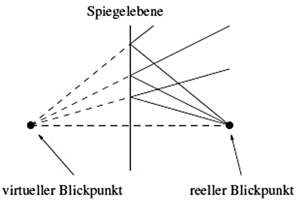
- Teilweise müssen telezentrische Linsen verwendet werden
- Der vertikale Blickwinkel ist unterschiedlich
- Die volle Auflösung der Kamera wird nicht genutzt
- Nicht jedes omnidirektionalen Bild kann perspektivisch korrekt in ein Panoramabild umgerechnet werden

⇒ Um ein perspektivisch korrektes Abbild aus einem Spiegelbild zu erzeugen, benötigt man einen festen Blickpunkt (engl. fixed viewpoint)

Effektiver Blickpunkt

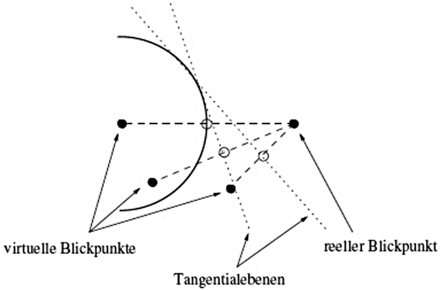
- Von diesem (virtuellen) Blickpunkt scheint man die Szene zu betrachten
- In ihm schneiden sich alle Lichtstrahlen die vom Spiegel reflektiert werden
- Dieser Standort des Betrachters entspricht dem Loch einer Lochkamera
- Bei einer Kamera mit Linsensystem befindet sich dieser Punkt im hinteren Brennpunkt

Effektiver Blickpunkt (cont.)



- Bei einem planaren Spiegel liegt der effektive Blickpunkt des Betrachters auf der gegenüberliegenden Spiegelseite
- Der Betrachter hat den Eindruck als sehe er die Szene vom effektiven Blickpunkt aus
- Der Abstand vom Spiegel zum Betrachter ist der gleiche wie vom Spiegel zum virtuellen Blickpunkt

Effektiver Blickpunkt (cont.)



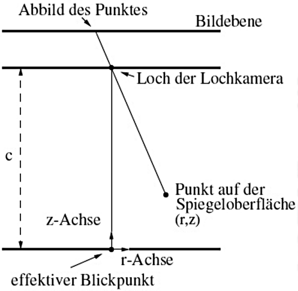
- Ist die Spiegeloberfläche gewölbt, muss an jeden Punkt der Oberfläche die Tangentialebene angelegt werden
- Auf der Senkrechten durch den realen Blickpunkt liegt auch der virtuelle Blickpunkt

Effektiver Blickpunkt (cont.)

- Als Konsequenz ergeben sich mehrere oder unendlich viele virtuelle Blickpunkte $\rightarrow$ es kann kein perspektivisch korrektes Bild berechnet werden

- **Hinweis:** Beispiele aus der Robotik zeigen, dass diese Bedingung nicht für jede Anwendung notwendig ist
- Perspektivisch korrekte Bilder sind allerdings für den Menschen besser zu verarbeiten
- Durch geschickte Konstruktion der Spiegel lässt sich aber **ein** effektiver Blickpunkt erzeugen

Fixed Viewpoint Constraint



Gesucht: Spiegeloberfläche $z(r)$, so dass ein effektiver Blickpunkt entsteht

Fixed Viewpoint Constraint (cont.)

Nayar und Baker, 1997:

Spiegelformen, die sich aus Lösungen der folgenden quadratischen Differentialgleichung ergeben, haben einen effektiven Blickpunkt

Fixed Viewpoint Constraint

$$r(c-2z)\left(\frac{dz}{dr}\right)^2 - 2(r^2+cz-z^2)\frac{dz}{dr} + r(2z-c) = 0$$

Allgemeine Lösung des Fixed Viewpoint Constraints

$$\left(z - \frac{c}{2}\right)^2 - r^2\left(\frac{k}{2} - 1\right) = \frac{c^2}{4}(k - 2k) \qquad (k \geq 2)$$

$$\left(z - \frac{c}{2}\right)^2 + r^2\left(1 + \frac{c^2}{2k}\right) = \left(\frac{2k + c^2}{4}\right) \qquad (k > 0)$$

Spezielle Lösungen des Fixed Viewpoint Constraints

Fünf Lösungen:

1. $k = 2$ und $c > 0$: **Planarer Spiegel**
   $\Rightarrow$ keine omnidirektionalen Bilder

$$z = c/2$$

2. $k \geq 2$ und $c = 0$: **Konischer Spiegel**
   $\Rightarrow$ Blickpunkt in Kegelspitze

$$z = \sqrt{\frac{k-2}{2}\ r^2}$$

Spezielle Lösungen des Fixed Viewpoint Constraints (cont.)

3. $k > 0$ und $c = 0$: **Sphärischer Spiegel**
   $\Rightarrow$ effektiver und realer Blickpunkt in Kugelmitte

$$z^2 + r^2 = k/2$$

4. $k > 0$ und $c > 0$: **Ellipsoider Spiegel**
   $\Rightarrow$ effektiver und realer Blickpunkt innerhalb der Ellipse

$$\frac{1}{a_e^2}\left(z - \frac{c}{2}\right)^2 + \frac{1}{b_e^2} r^2 = 1$$

$$a_e = \sqrt{\frac{2k + c^2}{4}} \qquad b_e = \sqrt{\frac{k}{2}}$$

Spezielle Lösungen des Fixed Viewpoint Constraints (cont.)

5. $k > 2$ und $c < 0$: **Hyperboloider Spiegel**

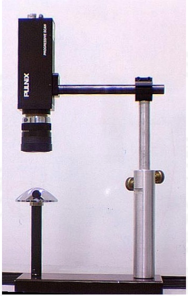
$$\frac{1}{a_h^2}\left(z - \frac{c}{2}\right)^2 + \frac{1}{b_h^2} r^2 = 1$$

$$a_h = \frac{c}{2}\sqrt{\frac{k-2}{k}}$$

$$b_h = \frac{c}{2}\sqrt{\frac{2}{k}}$$

Aufbau omnidirektionaler Sichtsysteme

1. Möglichkeit: Spiegel wird mit Halterung über der Kamera montiert



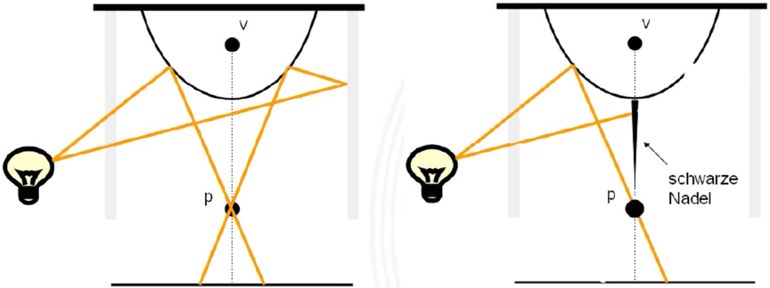
Quelle: `http://cmp.felk.cvut.cz/demos/Omnivis/Photos/fotohyper2.jpg`

Problem: Halterung ist immer im Panoramabild zu sehen

Aufbau omnidirektionaler Sichtsysteme (cont.)

2. Möglichkeit: Spiegel wird mit Glaszylinder oder -halbkugel über der Kamera montiert

   Problem: Reflexionen an der Innenseite des Zylinders/der Halbkugel



Aufbau omnidirektionaler Sichtsysteme (cont.)

Omnidirektionale Aufname:

Links mit Reflexionen, rechts ohne Reflexionen

Hersteller hyperboloidaler Spiegel

- PANORAMA EYE®: Seiwapro Co., Ltd.,
  `http://www.accowle.com/englisch/`

- „Mirror-Lens“ Single-Shot Panoramaoptik:
  Panorama-Hardware, Hamburg, Deutschland
  `http://www.panorama-hardware.de`

Panorama-Berechnung

Zwei Verfahren zur Umrechung auf ein Panoramabild

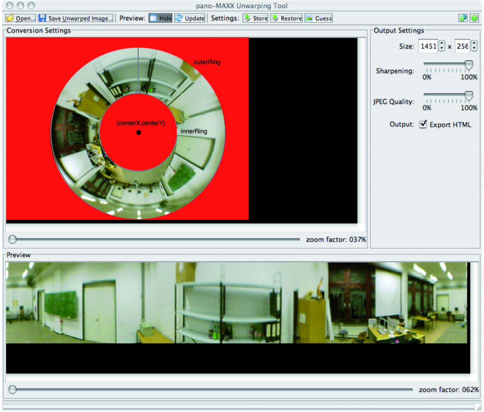
- **Einfache direkte Umwandlung**

  Das Bild aus dem omnidirektionalen Sichtsystem wird gleichmäßig abgetastet

- **Hyperboloidale Projektion**

  Eine Panorama ist eine Projektion auf einen Zylinder um den Spiegel

Einfache Umrechnung



z.B. mit der kostenlosen Software *PUT* von `www.panorama-hardware.de`

Einfache Umrechnung (cont.)

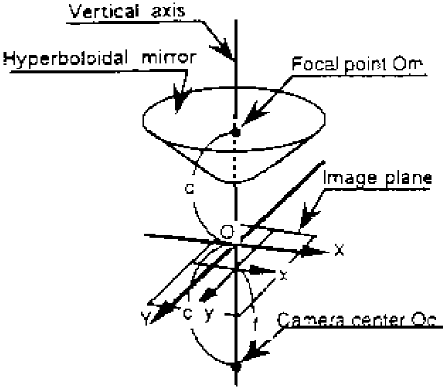
Für ein Panorama der Größe *width* × *height* wird für jeden Punkt der Punkt im omnidirektionalen Bild ermittelt

```
Image createPanorama (int width, int height, Image omni)
{
  Image panorama = new Image (width, height);
  for (int i=0; i<width; i++) {
    for (int j=0; j<height; j++) {
      panorama.setPixel
        (i, j, getPixelFromOmnidirectional (i, j, omni));
    }
  }
}
```

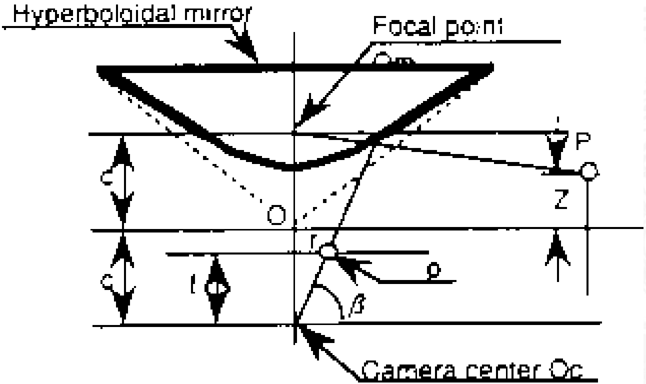
Einfache Umrechnung (cont.)

```
Pixel getPixelFromOmnidirectional (int i, int j, Image omni)
{
  // Polarkoordinaten bestimmen
  double radius = outerRing -
                 ((j/height) * (outerRing-innerRing));
  double alpha  = - (i/width) * (2*PI);

  // Koordinaten im omnidirektionalen Bild
  double x = centerX - radius * sin(alpha);
  double y = centerY + radius * cos(alpha);

  // Interpolierten Punkt zurueckgeben
  return omni.getInterpolatedPixel (x, y);
}
```

Hyperboloidale Projektion



Reduktion auf 2D-Problem



Reduktion auf 2D-Problem (cont.)

Hyperbel-Gleichung:

$$\frac{R^2}{a^2} - \frac{Z^2}{b^2} = -1$$

$$c = \sqrt{a^2 + b^2}$$

Hyperboloidale Projektion

Im Wesentlichen sind zwei Schritte erforderlich
(nach *Yamazawa et. al., 1993*)

1. Projektion eines Zylinderpunktes auf die Spiegeloberfläche
2. Projektion von der Spiegeloberfläche auf die Bildfläche

$$
\begin{aligned}
r &= f \cdot \tan(\beta) \\
\beta &= \tan^{-1}\left(\frac{(b^2+c^2)\cos(\alpha) - 2bc}{(b^2-c^2)\cos(\alpha)}\right) \\
\alpha &= \tan^{-1}\left(\frac{(b^2+c^2)\cos(\beta) - 2bc}{(b^2-c^2)\cos(\beta)}\right) \\
\alpha &= \frac{R}{Z-c}
\end{aligned}
$$

Kamerakalibrierung

Um die hyperboloidale Projektion durchzuführen ist eine Kamerakalibrierung notwendig:

- Bekannt: a und b der Hyperbel und Spiegedurchmesser
- Es wird berechnet: extrinsische Parameter (Translation und Rotation zwischen Spiegel und Kalibrierungsgrid) und intrinsische Parameter (Fokus, Verzeichnung und Versatz)

Für die Kalibrierung gibt es ein MATLAB-Toolkit:

`http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/links.html`

Symmetrie

„Symmetry is what we see at a glance; [. . .]"

aus: *Pensées - Section 1: Thoughts on mind and on style*
von: Blaise Pascal, 1660

Was ist Symmetrie?

- Symmetrie ist überall
- Symmetrie beeinflußt unsere Wahrnehmung:
  - Symmetrische Gesichter sind schön
  - Symmetrische Architektur ist schön
  - Symmetrische Schneeflocken fallen aus den Wolken

Was ist Symmetrie? (cont.)

- griech.: **symmetros** = regelmäßig, ebenmäßig
- Symmetrie ist vielfältig, aber es steckt immer
  - **Regelmäßigkeit** oder
  - **wiederkehrendes Element**

  dahinter
- Symmetrie hat etwas mit dem zu tun, was wir sehen:
  - mit Formen
  - mit Mustern
  - mit dem Aussehen der Welt

Was ist Symmetrie? (cont.)

Beispiel für Symmetrie: **Die Schneeflocke**

- Man kann jede Schneeflocke um 60° drehen ...
- und dann sieht sie aus wie vorher!
- Schneeflocken sind **drehsymmetrisch**
- Die Drehung um 60° wird als **Symmetrieoperation** bezeichnet

Symmetrie-Operation

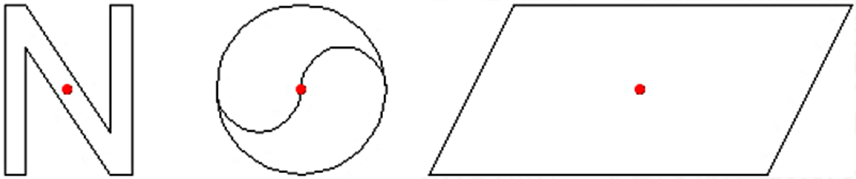
Eine Symmetrieoperation macht etwas mit einem Ding, und danach sieht man nicht, das etwas gemacht wurde

Im Zweidimensionalen sind vor allem die folgenden Symmetrien von Bedeutung

- Drehsymmetrie
- Spiegelsymmetrie
- Translationssymmetrie

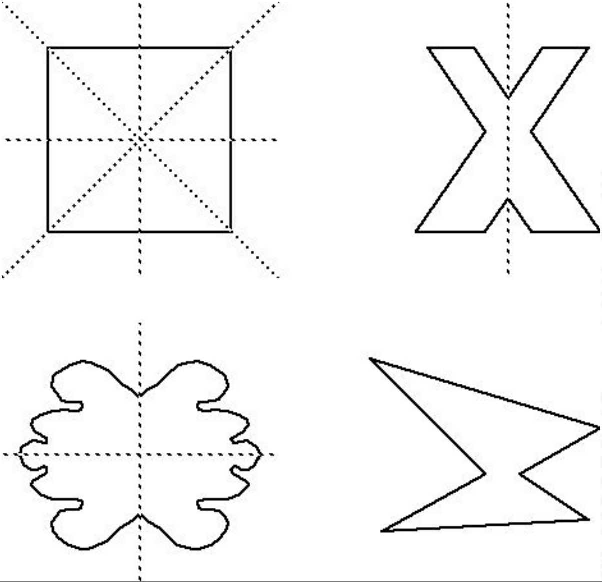
Drehsymmetrie

- auch: Radialsymmetrie oder Radiärsymmetrie
- Die Drehung eines Objektes um einen gewissen Winkel, um einen bestimmten Punkt (2D) bzw. eine Gerade (3D) bringt das Objekt wieder mit sich selbst zur Deckung
- Spezialfall: **Punktsymmetrie**
  - Punktspiegelung
  - Drehung eines Objektes um 180° um ein Zentrum



Spiegelsymmetrie

- auch: Bilaterale Symmetrie oder Achsensymmetrie
- häufigste Form der Symmetrie
- 2D: Symmetrieachse
- 3D: Symmetrieebene
- Zwei Punkte sind identisch, wenn
  sie auf einer Senkrechten zur Achse/Ebene liegen und
  den gleichen Abstand zur Achse/Ebene haben



Translationssymmetrie

- Einfaches Muster: GGGGGGGGGGGGG
- Symmetrieoperation: „Alle Gs einen Platz nach links rücken!“
- Muster bleibt wie oben
- Teilweise feste Verschiebung notwendig, teilweise beliebig

[M.C. Escher]

Andere Formen der Symmetrie

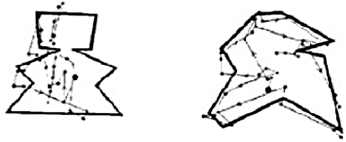
Sprache: Palindrome wie Otto und Anna

Musik: Johann Sebastian Bach

Contrapunktus: Die Noten scheinen an der Horizontalen gespiegelt

Wahrnehmung von Symmetrie

- Symmetrie ist wichtig für das menschliche Sehen



[Locher & Nodine (1989)]

- Symmetrie erzeugt Aufmerksamkeit und unterstützt visuelle Exploration
- **Besonders bedeutend:**
  Horizontale und vertikale Spiegelsymmetrien
  [Palmer & Hemenway (1978)]

Symmetrie-Detektoren

In der Bildverarbeitung gibt es viele Detektoren, die Symmetrien in Bildern ermitteln:

- Generalized Symmetry Transform
- Discrete Symmetry Transform
- Fast Radial Symmetry Transform
- 1-dimensionaler Symmetrie-Detektor
- Quantitativer bilateraler Symmetrie-Detektor

Generalized Symmetry Transform

- Reisfeld, Wolfson und Yeshurun (1995)
- Lokale Operation auf Kantenbild
- Sei $p_k = (x_k, y_k)$ ein Bildpunkt und $\nabla p_k = (\frac{\partial}{\partial_x} p_k, \frac{\partial}{\partial_y} p_k)$ der Gradient der Intensität des Punktes
- Sei $v_k = (r_k, \theta_k)$ für jeden Punkt mit
  - $r_k = \log(1+ \parallel \nabla p_k \parallel)$ und
  - $\theta_k = \arctan(\frac{\partial}{\partial_x} p_k, \frac{\partial}{\partial_y} p_k)$
- Sei des Weiteren l eine Linie durch zwei beliebige Punkte p_i und p_j und α_{ij} der Winkel zwischen l und der Horizontalen gegen den Uhrzeigersinn

Generalized Symmetry Transform (cont.)

- Es wird eine Nachbarschaftsbeziehung definiert:

$$\Gamma(p) = \left\{ (i,j) \middle| \frac{p_i + p_j}{2} = p \right\}$$

- Außerdem eine Gewichtung der Distanz eines Punktepaares zu p:

$$D_\sigma(i,j) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{\|p_i - p_j\|}{2\sigma}}$$

- Sowie eine Gewichtung der Phasen:

$$P(i,j) = (1 - \cos(\theta_i + \theta_j - 2a_{ij}))(1 - \cos()\theta_i - \theta_j)$$

Generalized Symmetry Transform (cont.)

- Es wird definiert, dass jedes Punktepaar

$$C(i,j) = D_\sigma(i,j)P(ij)r_i r_j$$

  zum Symmetriewert eines Punktes p liefert
- Der Symmetriewert ist dann:

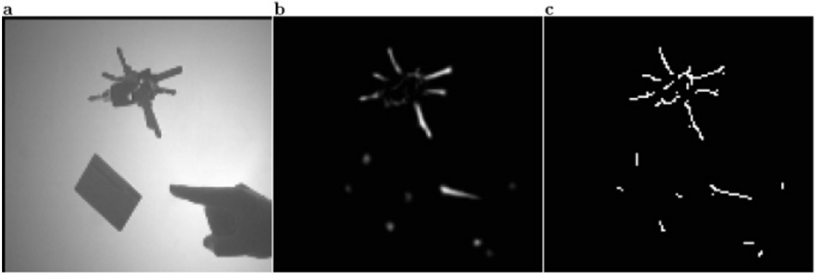
$$M_\sigma(p) = \sum_{(i,j)\in\Gamma(p)} C(i,j)$$

- Die Symmetrierichtung wird definiert als $\phi(p) = \varphi(i,j)$, so dass $C(i,j)$ maximal für alle $(i,j) \in \Gamma(p)$ ist

Generalized Symmetry Transform (cont.)

Abschließend wird die Symmetrie eines Punktes p definiert als:

$$S_\sigma(p) = (M_\sigma(p), \phi(p))$$



Discrete Symmetry Transform

- Di Gesù und Valenti (1995)
- Zweiteiliger Algorithmus zur Berechnung der Kreissymmetrie (alle Punkte auf einen Kreis identisch):

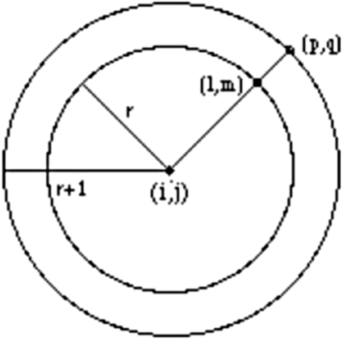
1. **Auswahl nicht-uniformer Zonen:**

   Berechnung eines Filters, so dass

$$E(i,j) = \sum_{(l,m)\in C_r,(p,q)\in C_{r+1}} |g(l,m) - g(p,q)|$$

   mit $g(i,j)$ als Grauwert eines Bildpunktes, C_r und C_{r+1} als digitale Kreise und so miteinander verbundenen Punkten, dass $(l-p)^2 + (m-q)^2 = 1$ gilt

Discrete Symmetry Transform (cont.)



Discrete Symmetry Transform (cont.)

2. **Berechnung der Symmetriewerte:**

   Überall wo $E(i,j) > 0$:

$$DST(i,j) = E(i,j) \times T(i,j)$$

mit:

$$T(i,j) = 1 - \sqrt{\frac{\sum_k (T_k(i,j))^2}{n} - \left(\frac{\sum_k (T_k(i,j))}{n}\right)^2}$$

wobei

$$T_k(i,j) = \sum_{(l,m)\in C_r} \left|(i-l)\sin\left(\frac{k\pi}{n}\right) - (j-m)\cos\left(\frac{k\pi}{n}\right)\right| \times g(l,m)$$

Anwendung der Discrete Symmetry Transform

Extraktion von kreisförmigen Merkmalen aus Bildern:

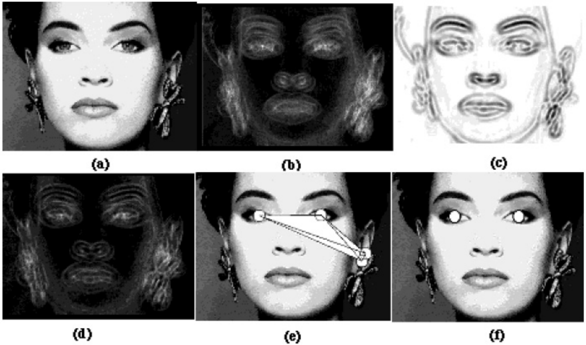
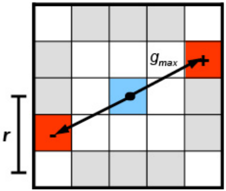


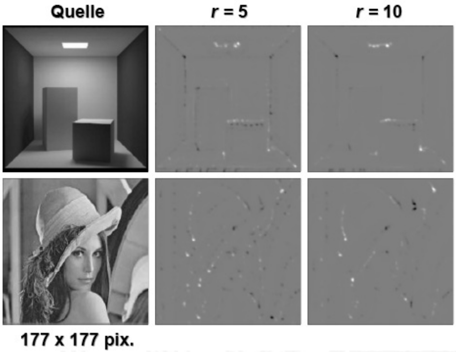
Fig. 8. a) Input image F; b) the image $E(F)$; c) the image $T(F)$; d) the image $DST(F)$; e) candidate interest points; f) final result.

Fast Radial Symmetry Transform

- Loy und Zelinsky (2003)

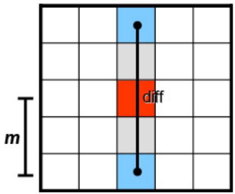


- Richtung des maximalen Gradienten



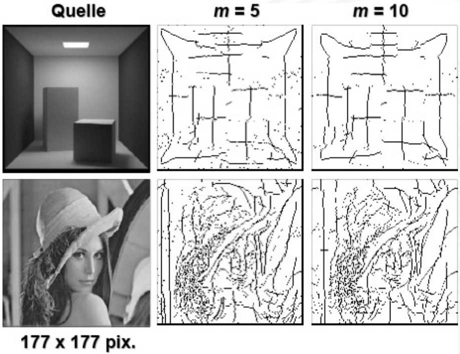
1-dimensionaler Symmetrie-Detektor

- Hübner und Zhang (2003)



- Normalisierter MSE

$$S^{v}(p_{x,y}, m) = 1 - \frac{1}{C \cdot m} \sum_{i=1}^{m} \sigma(i, m) \cdot g^{v}(p_{x-i,y}, p_{x+i,y})^2$$



Diskussion der bis hierher vorgestellten Ansätze

Symmetrie ist ein regionales Merkmal

- Jedes Resultat stark abhängig von der Wahl der Operatorgröße
- Symmetriewerte nur relativ ausgedrückt
- Mit der Größe der betrachteten Region wächst die Aussagekraft, aber auch der Rechenaufwand
- Große Operatoren reichen oft über den Bildrand hinaus
- Große Operatoren anfälliger für Beleuchtungsschwankungen, Verdeckungen und perspektivische Verzerrung
- Hierarchien unterschiedlicher Operatorgrößen zu nutzen, ist zu aufwändig
- Ohne Vorwissen keine optimale Operatorgröße möglich

Diskussion der bis hierher vorgestellten Ansätze (cont.)

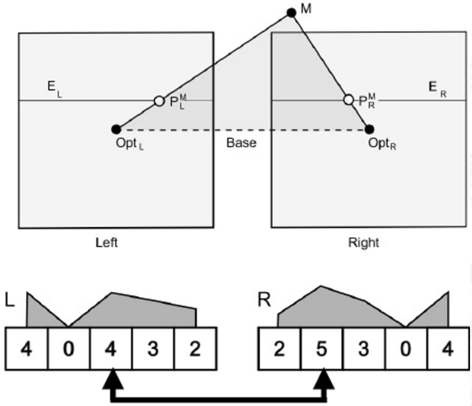
- Bisherige Ansätze alle **qualitativ**
- Wünschenswert wären **quantitative** Aussagen

Qualitative Symmetrie	Quantitative Symmetrie
„Symmetrie ist **[hoch]** in einer Umgebung von 10 Pixeln"	„Symmetrie ist hoch in einer Umgebung von **[10 Pixeln]**"
• benötigt a-priori Operatorgröße • gibt einen relativen, für gewöhnlich normalisierten Grad der Symmetrie in der betrachteten Umgebung wieder	• benötigt ein Abbruchkriterium • gibt eine absolute Größe zurück, die die symmetrische Umgebung beschreibt
Beispiele: Reisfeld operator, DiGesù operator, Loy operator, 1D operator, ...	Probleme: Ungenauigkeit, Fehlermaß Ansatz: **Dynamische Programmierung**

Dynamisches Programmieren

- Effiziente Methode zur Lösung des Stereomatchingproblems
- Distanzabschätzung durch Beobachtung der Verschiebung korrespondierender Bildmerkmale
- Vereinfachung durch Nutzung der Epipolarlinien
- Lösung im 2D-Suchraum

Dynamisches Programmieren (cont.)



Dynamisches Programmieren (cont.)

Links

Rechts

[Cox et al. (1996)]

[Bobick and Intille (1999)]

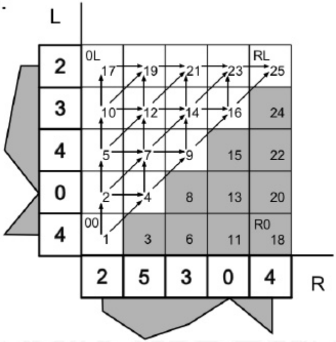
Dynamisches Programmieren (cont.)

Berechnung von Kosten und optimalem Pfad:
[Žganec et al., 1996]

$$C(l,r) = min(C_1(l,r), C_2(l,r), C_3(l,r))$$

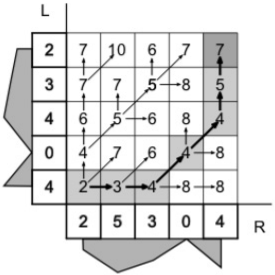
$$C_1(l,r) = C(l-1,r) + W_1T(l,r)$$
$$C_2(l,r) = C(l,r-1) + W_2T(l,r)$$
$$C_3(l,r) = C(l-1,r-1) + W_3T(l,r)$$

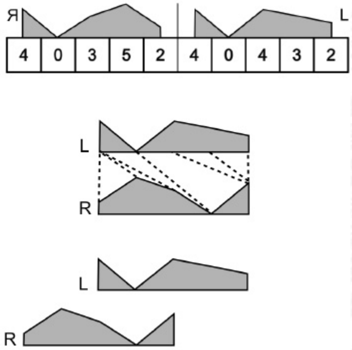
$$T(l,r) = |L_l - R_r|$$



Quantitative bilaterale Symmetrie durch Dynamische Programmierung

Berechnung des gesamten Suchraums:

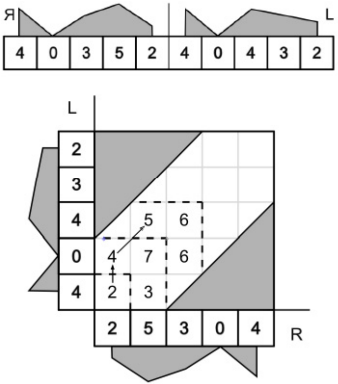




DPS Algorithmus

Iterative Berechnung mit Schwellwert:

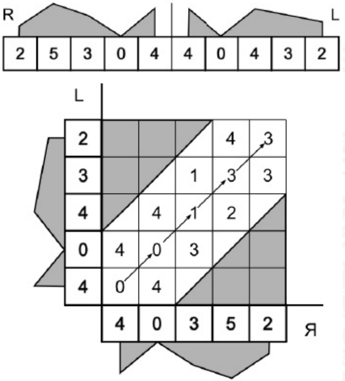
- Parameter:
  - Bandmaske Weite $w_B = 2$
  - Absoluter Schwellwert $T = 5$
  - Gewichte $W_i = 1$
- Ergebnis:
  - Iterationsstop bei $I = 2$
  - Mapping index $(1 / 2) \rightarrow S = 3$



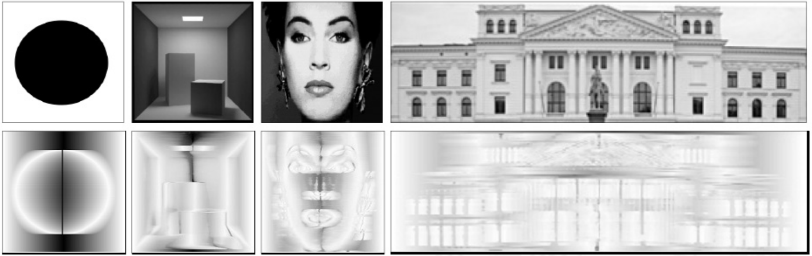
DPS Algorithmus (cont.)

Iterative Berechnung mit Schwellwert:

- Parameter:
  - Bandmaske Weite $w_B = 2$
  - Absoluter Schwellwert $T = 5$
  - Gewichte $W_i = 1$
- Ergebnis:
  - Iterations stop bei $l = 4$ (max)
  - Mapping index (4 / 4) $\rightarrow$ S = 8

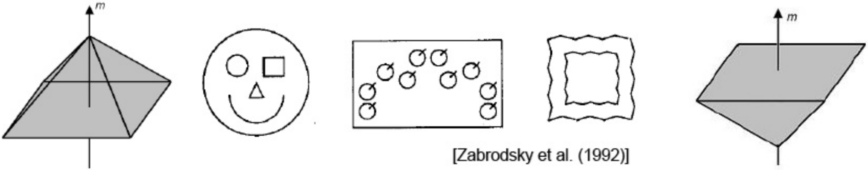


DPS Algorithmus - Beispiele



Symmetrie-Hierarchien

In **qualitativen** Ansätzen werden Hierarchien auf Basis unterschiedlicher Operatorgrößen betrachtet (**multiskalar**)

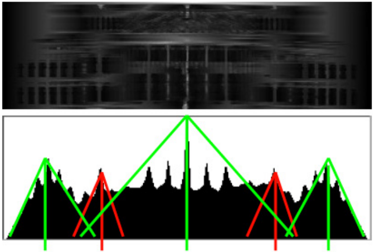


[Zabrodsky et al. (1992)]

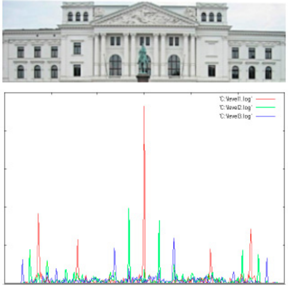
Multiskalare Hierarchie: $\{S(I, m_1), \ldots, S(I, m_n)\} \Rightarrow H_{ms}(I)$

Symmetrie-Hierarchien (cont.)

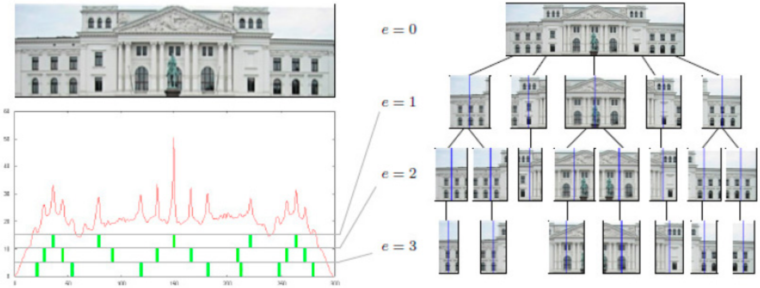
Bei quantitativen Operatoren können Hierarchien als Ordnung der Symmetrien selbst gesehen werden



- $S = S_L + S_R$
- Rekursive top-down Hierarchisierung $S(I) \Rightarrow H_{so}(I)$



Symmetrie-Hierarchien (cont.)



Symmetrie-Signaturen

Vierstufiger Signatur-Algorithmus

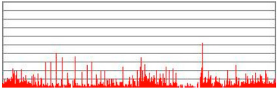
1. Erzeugen eines Panoramas
2. DPS Algorithmus
3. Non-Maxima-Suppression
4. Aufsummieren der Spalten zu Signatur-Vektor

Symmetrie-Signaturen (cont.)

(a) Panoramic image.

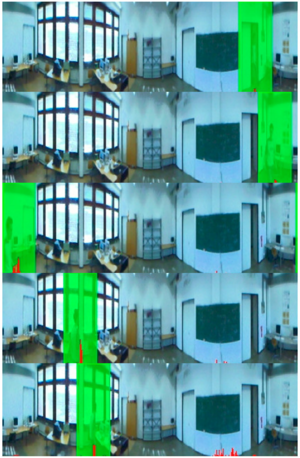
(b) Vertical symmetry image.

(c) Vertical symmetry image after non-maxima suppression.



(d) Visualization of the symmetry image signature.

Tracking über Symmetrie-Signatur-Fehlern



[CHH94] Roberto Cipolla, Paul A. Hadfield, Nicholas J. Hollinghurst:
Uncalibrated stereo vision with pointing for a man-machine interface.
In: *In Proc. IAPR Workshop on Machine Vision Applications (MVA94)*
(1994), Dezember, S. 163–166. –
Kawasaki. –
URL `mi.eng.cam.ac.uk/~cipolla/publicats.html`

[DESY00] *Was ist Symmetrie?*, 2000. –
URL
`www.desy.de/expo2000/deutsch/dhtmlbrowser/webthemen/08_symmetrie/symmetrie_druck.htm`

[GD01] C. Geyer, K. Daniilidis:
Catadioptric projective geometry.
In: *International Journal of Computer Vision* 43 (2001), S. 223–243. –
URL `citeseer.ist.psu.edu/article/geyer02catadioptric.html`

[GV95] V. Di Gesù, C. Valenti:
The Discrete Symmetry Transform in Computer Vision.
DMA Università di Palermo, 1995, Forschungsbericht

[Hue01] Kai Huebner:
Methods for Range Estimation and Situation Recognition using an Omnidirectional Vision System for Mobile Robots - Symmetry as a Natural Feature,
November 2001. –
URL `www.informatik.uni-bremen.de/~khuebner/diploma/pdf/index_d.html`

[HWZ05] K. Huebner, D. Westhoff, J. Zhang:
Optimized Quantitative Bilateral Symmetry Detection.
In: *International Journal of Information Acquisition (IJIA)*
2 (2005), September, Nr. 3, S. 241–249

[LN89] P. Locher, C. Nodine:
The perceptual value of symmetry.
In: *Comput. Math. Applic.*
17 (1989), S. 475–484

[LZ03] G. Loy, A. Zelinsky:
Fast Radial Symmetry for Detecting Points of Interest.
In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* Bd. 25.
2003, S. 959–973

[McK93] Kap. 3 In: Phillip John McKerrow:
Introduction to Robotics.
korrigierte.
Addison-Wesley, 1993
(Electronic Systems Engineering Series), S. 131–174

[Nay97] Shree K. Nayar:
Catadioptric Omnidirectional Camera.
In: *1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97)*.
1997, S. 482

[RWY95] D. Reisfeld, H. Wolfson, Y. Yeshurun:
Context Free Attentional Operators: the Generalized Symmetry Transform.
In: *International Journal of Computer Vision*
14 (1995), S. 119–130

[Tsa86] Roger Y. Tsai:
An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision.
In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.
Miami Beach, Florida, USA, 1986, S. 364–374

[Tsa87] Roger Y. Tsai:
A versatile camera calibration technique for 3D machine vision metrology using off-the-shelf TV cameras and lenses.
In: *IEEE Journal of Robotics and Automation*
RA-3 (1987), August, Nr. 4, S. 323–344

[WCH92] Juyang Weng, Paul Cohen, Marc Herniou:
Camera Calibration with Distortion Models and Accuracy Evaluation.
In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*
14 (1992), Nr. 10, S. 965–980. –
ISSN 0162–8828.
URL `dx.doi.org/10.1109/34.159901`

[YYY93] K. Yamazawa, Y. Yagi, M. Yachida:
Omnidirectional imaging with hyperboloidal projection.
In: *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems '93, IROS '93.* Bd. 2.
Yokohama, Japan, 26-30 Juli 1993, S. 1029–1034

[YYY95] K. Yamazawa, Y. Yagi, M. Yachida:
Obstacle detection with omnidirectional image sensor HyperOmniVision.
In: *Proceedings of the 1995 IEEE International Conference on Robotics and Automation* Bd. 1.
Nagoya, Japan, 21-27 May 1995, S. 1062–1067

[ZR96] Kap. 2.II.A, 2.III, 2.IV, 3.I und 3.II In: Hanqi Zhuang, Zvi S. Roth:
Camera-aided robot calibration.
CRC Press Inc, 1996, S. 11–14, 20–27 und 63–68