

# Robotik

Priv.-Doz. Dr. Thomas Wiemann  
Institut für Informatik  
Autonome Robotik

SoSe 2021

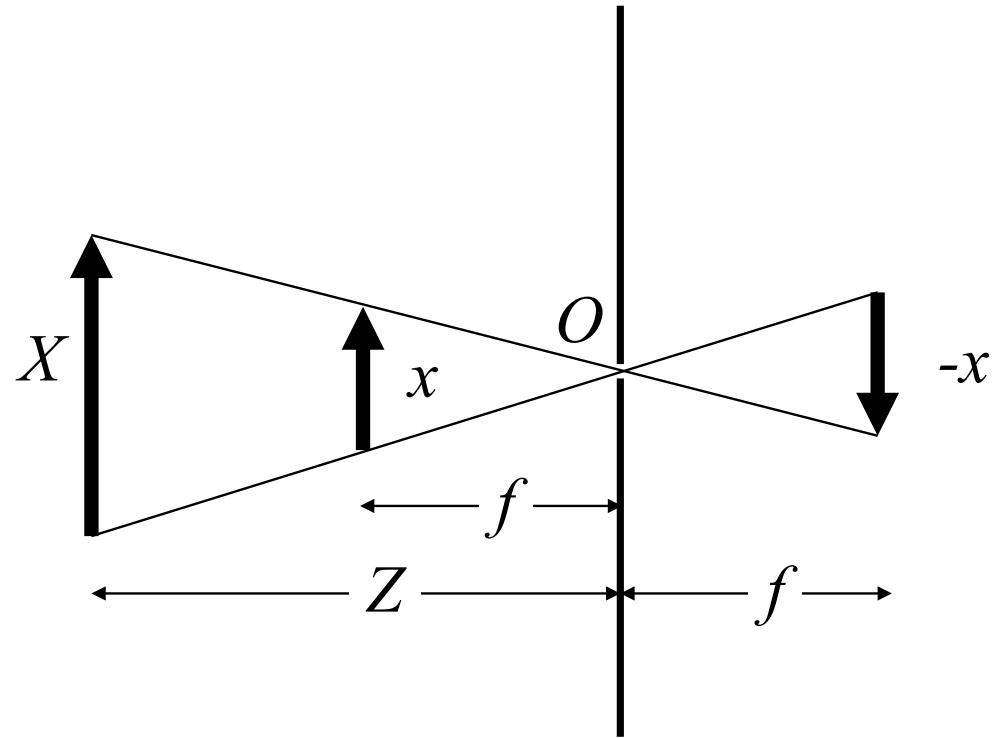




## Gliederung

- 3. Sensorik
  - 1. Allgemeines
  - 2. Bewegungsmessung
  - 3. Ausrichtungsmessung
  - 4. Globale Positionsbestimmung
  - 5. Entfernungsmessung
  - 6. Kamerarund Kameramodelle
  - 7. Farbsensoren
  - 8. Ausblick

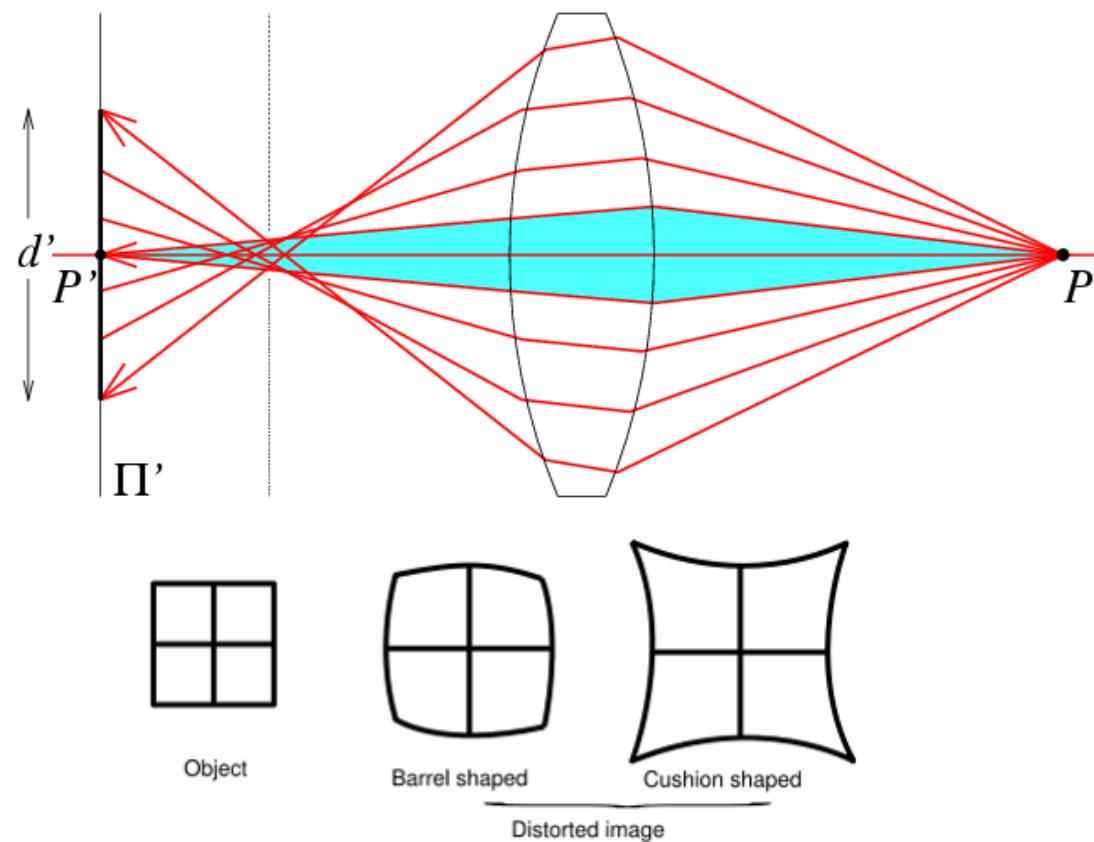
## Wdh.: Perspektivische Projektion



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

## Wdh.: Aberrationen

- ▶ Lichtstrahlen nicht-parallel zur optischen Achsen werden nicht exakt im Fokus gebündelt
- ▶ Die Brennweite sinkt in den äußeren Bereichen einer Linse



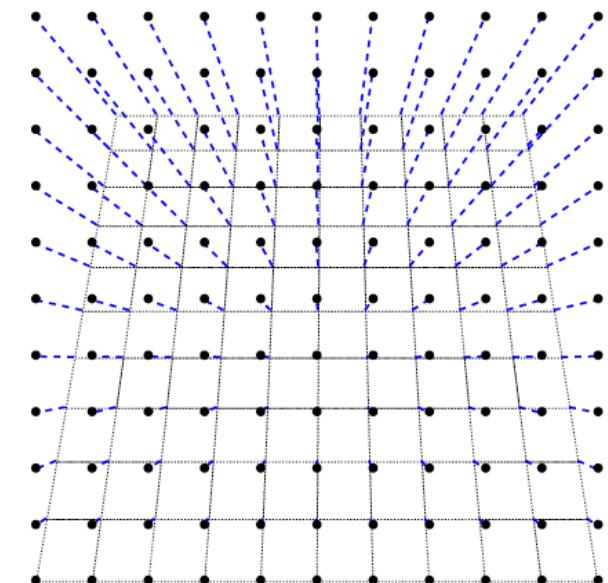
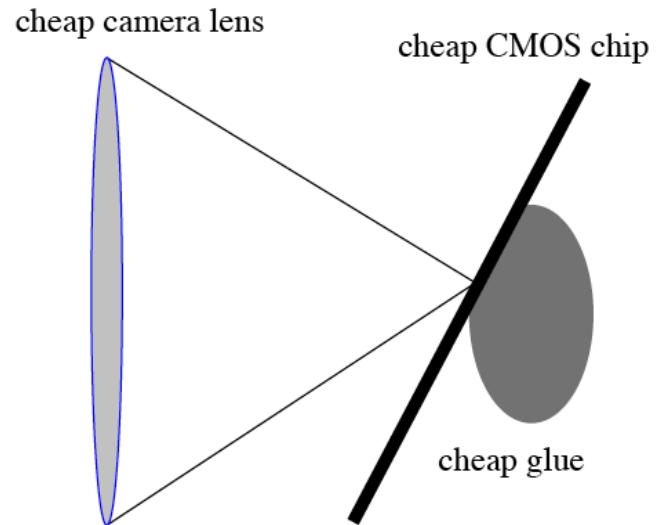
# Wdh.: Kamerakalibrierung

- Modell für radiale Verzerrung

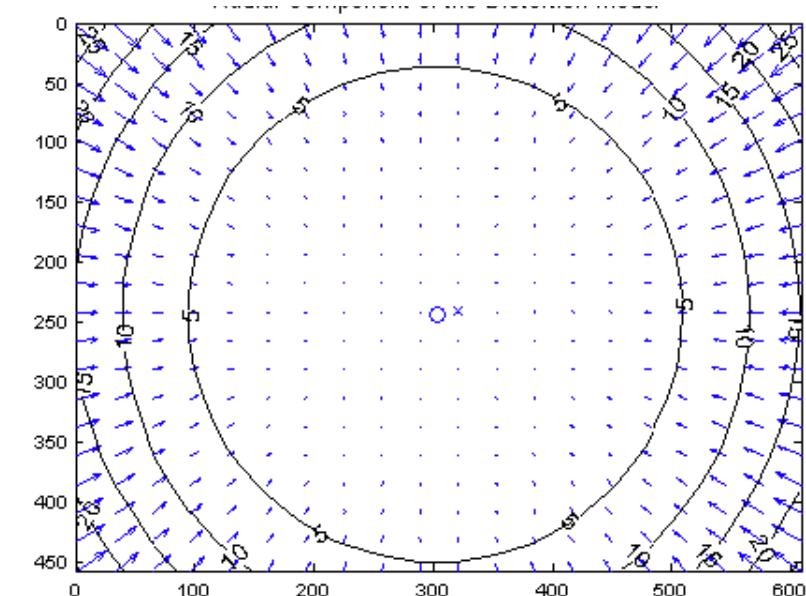
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_d \\ y_d \end{pmatrix} (1 + k_1 r^2 + k_2 r^4)$$

Abstand von der Bildmitte

- Tangentiale Verzerrung:



# Wdh.: Kamerakalibrierung in OpenCV



Pixel error	= [0.1174, 0.1159]
Focal Length	= (657.303, 657.744)
Principal Point	= (302.717, 242.334)
Skew	= 0.0004198
Radial coefficients	= (-0.2535, 0.1187, 0)
Tangential coefficients	= (-0.0002789, 5.174e-005)

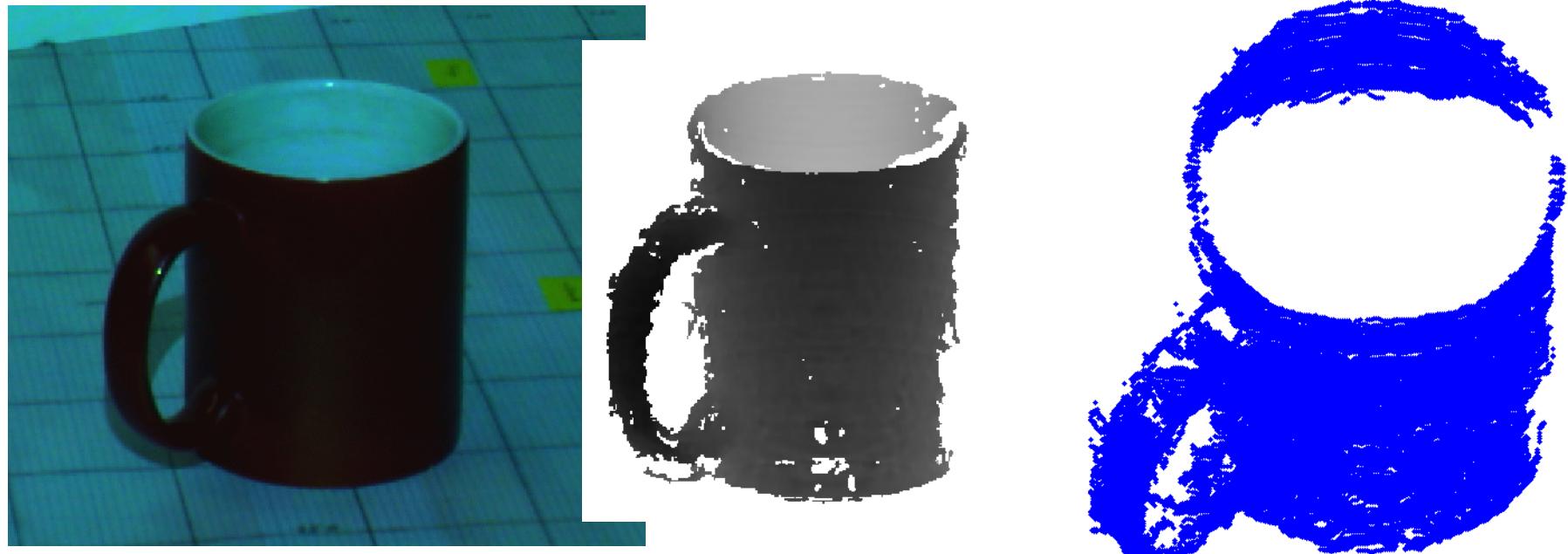


[Albrecht 2007]

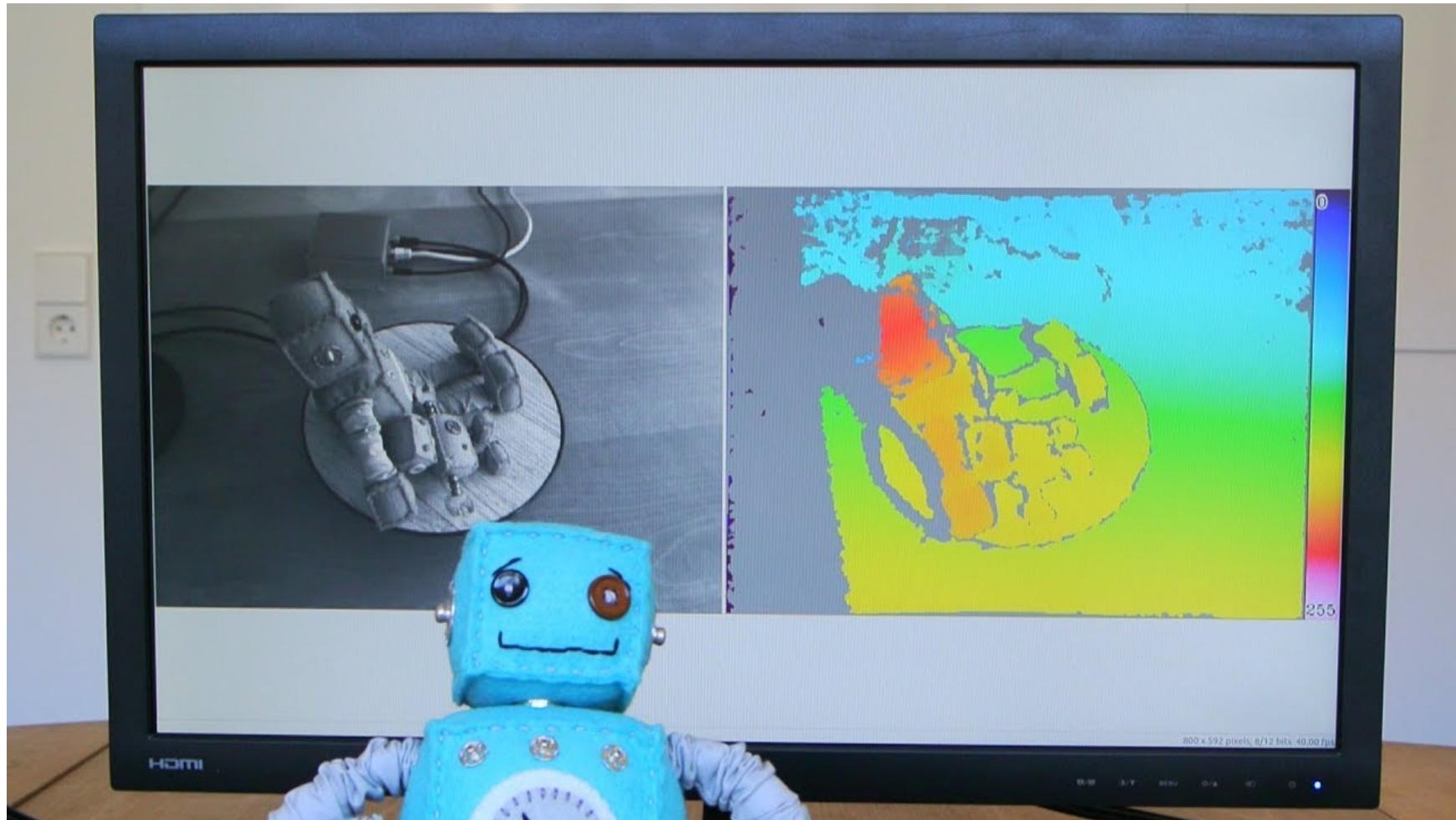
# Stereokameras



# Stereorekonstruktion



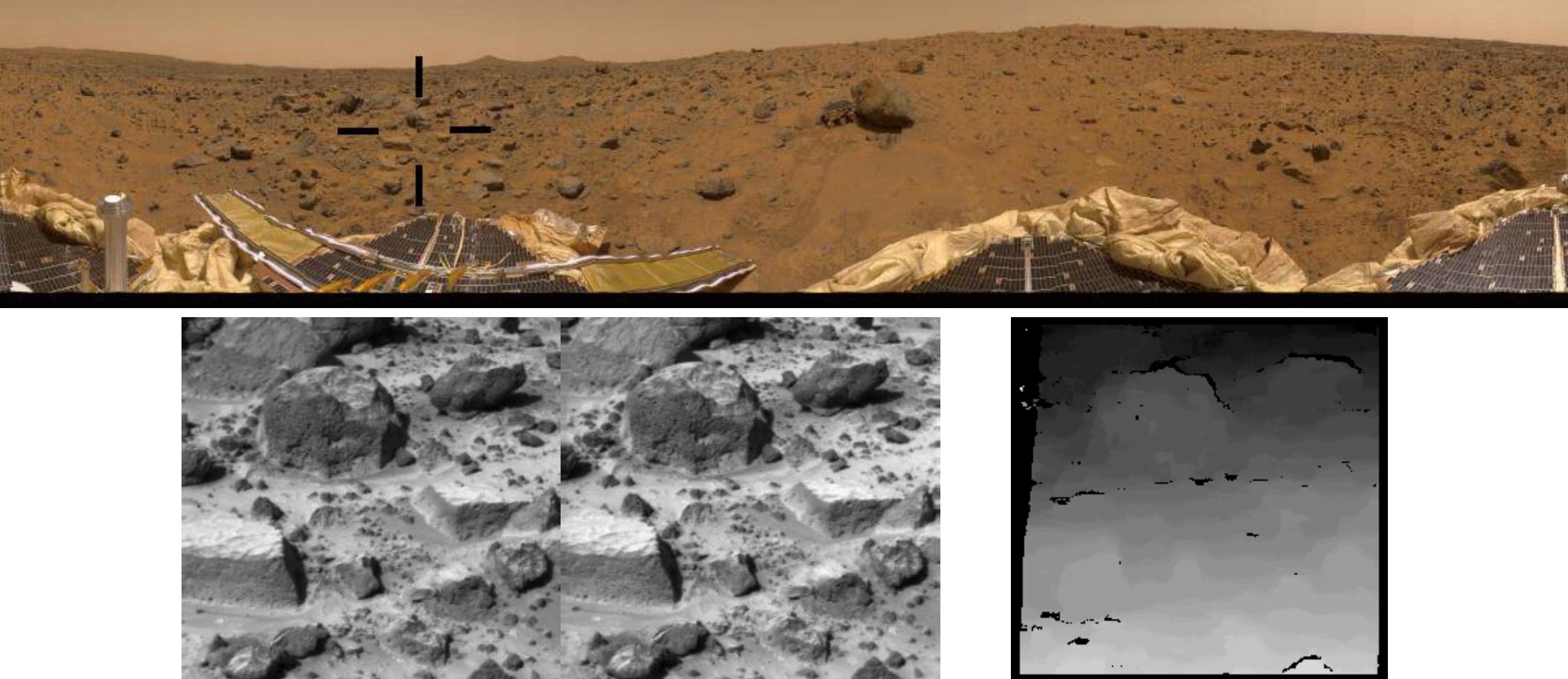
## Beispiel



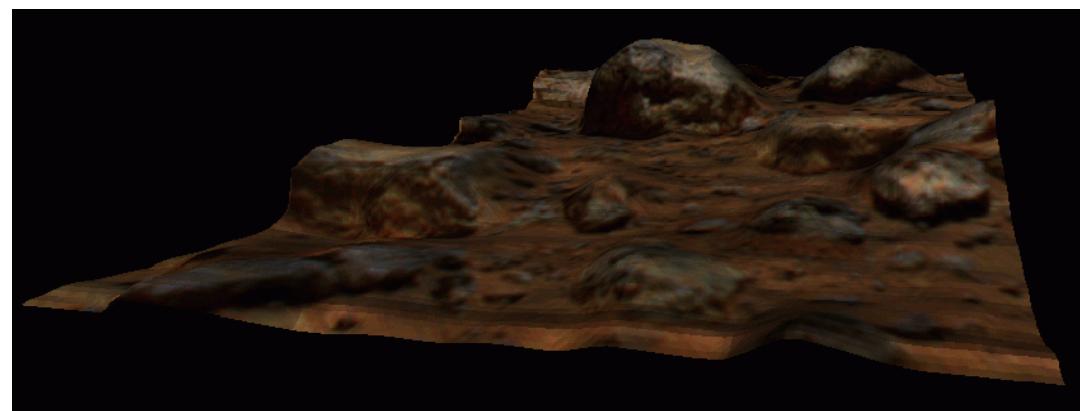
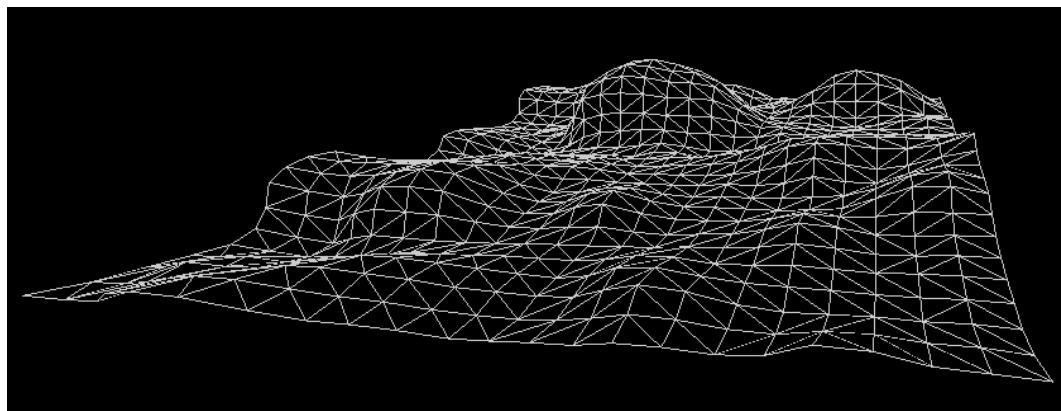
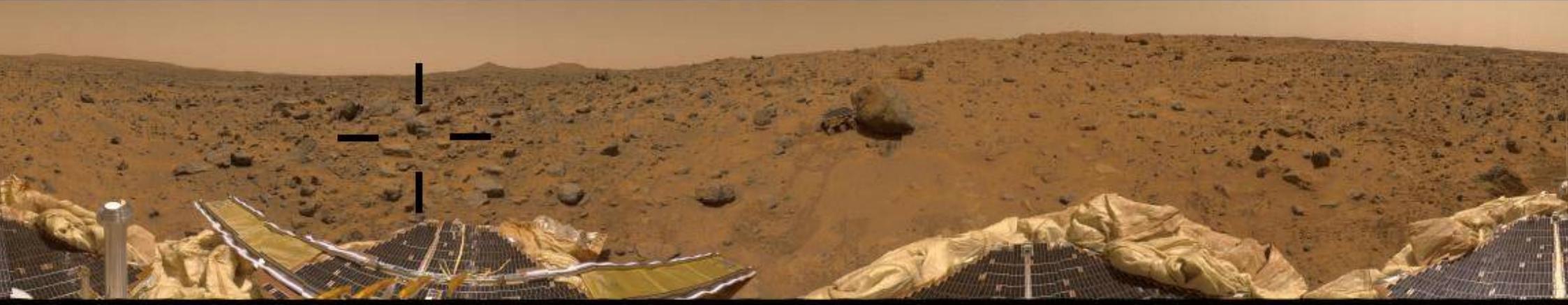
## Beispiel



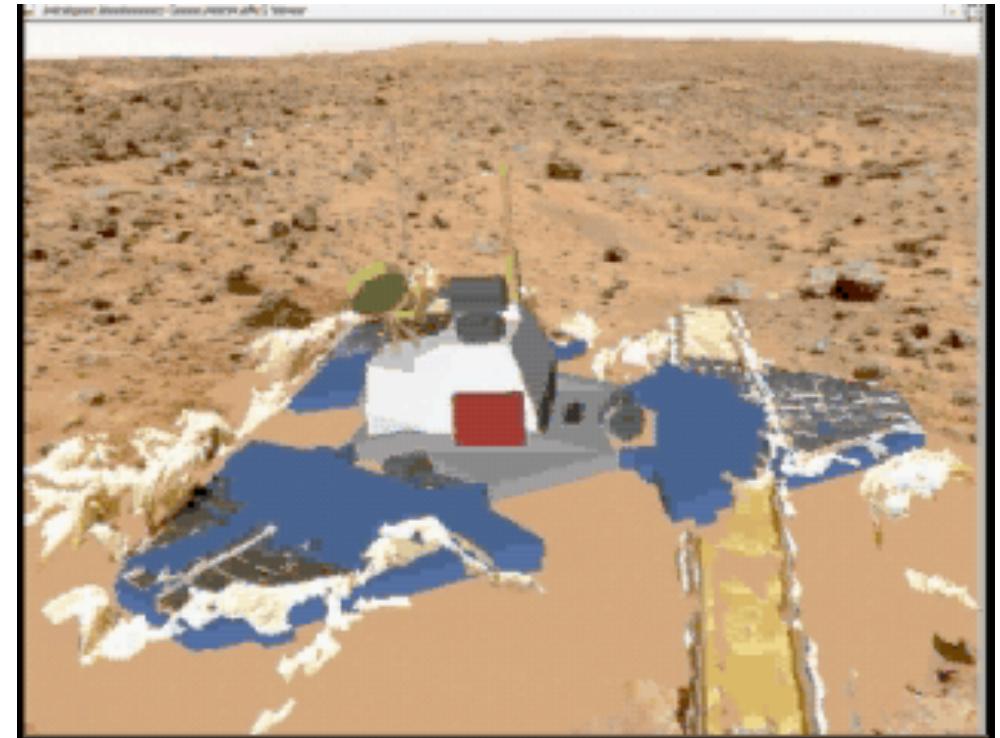
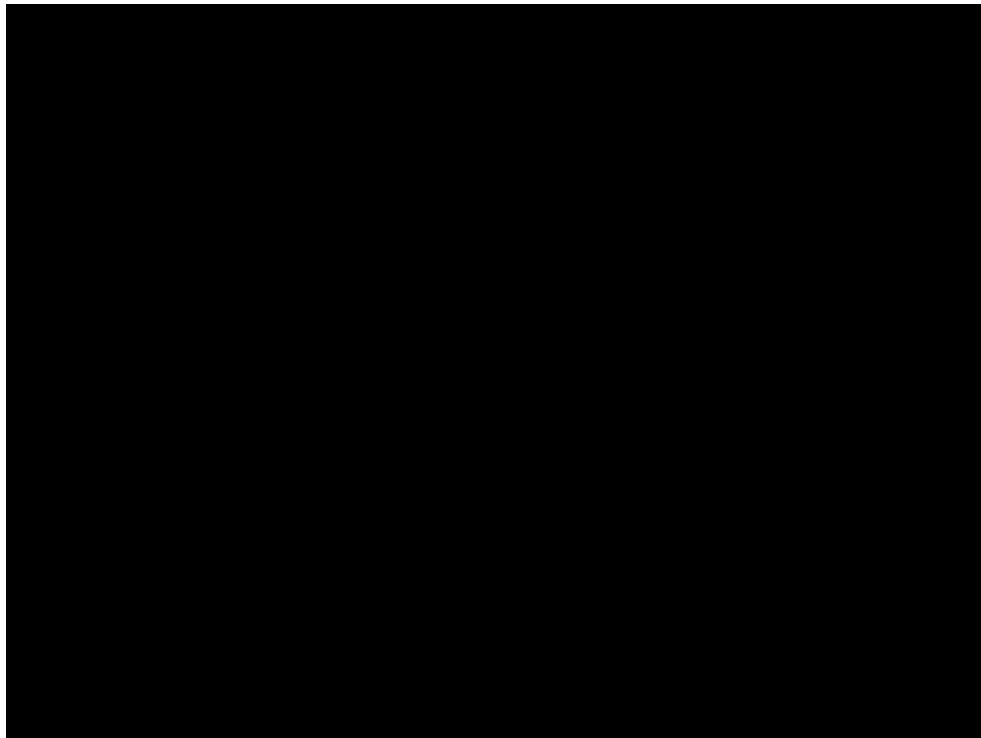
## Einsatzbeispiel



## Einsatzbeispiel

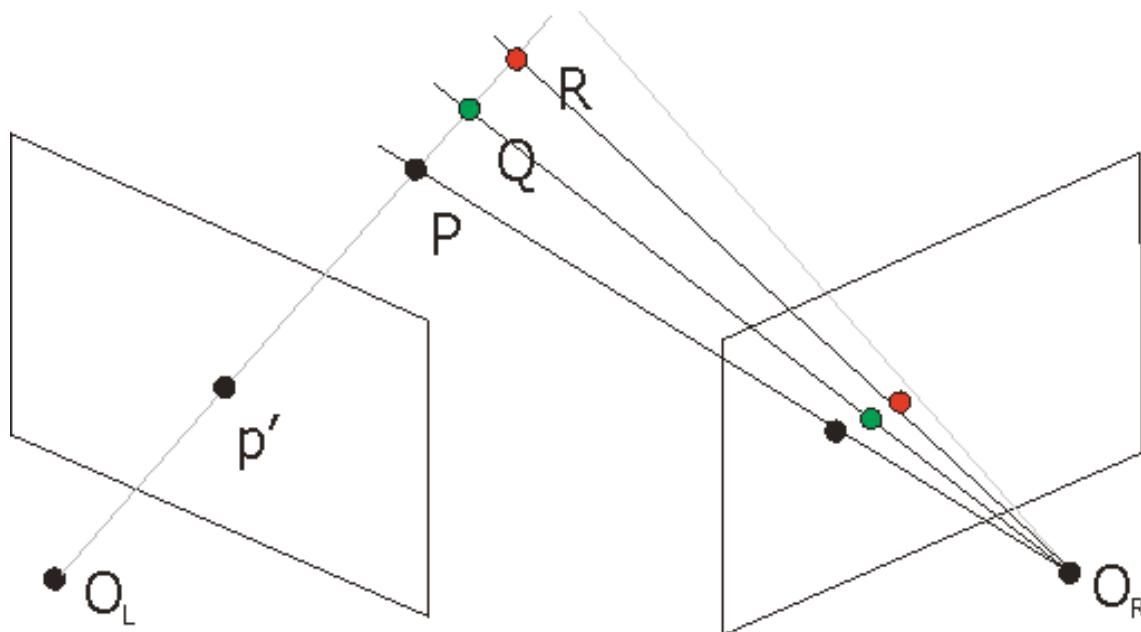


## Einsatzbeispiel

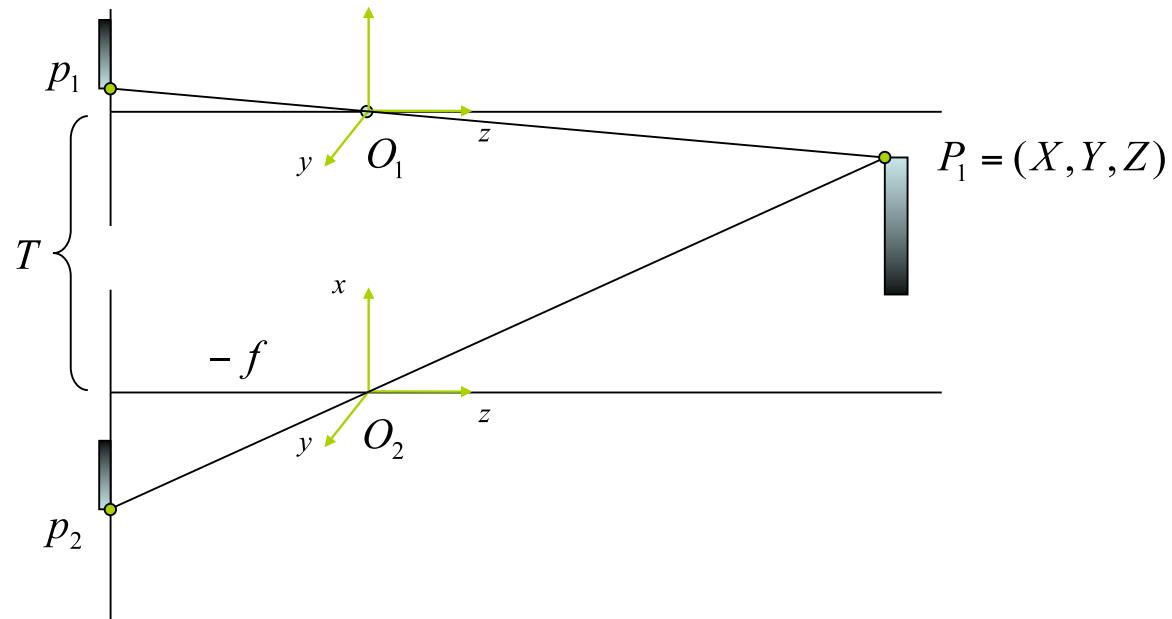


# Stereo - Geometrie (1)

- Ziel: Durch Betrachtung eines Punktes mit zwei Kameras die Tiefe eines Raumpunktes bestimmen
- Erinnerung:

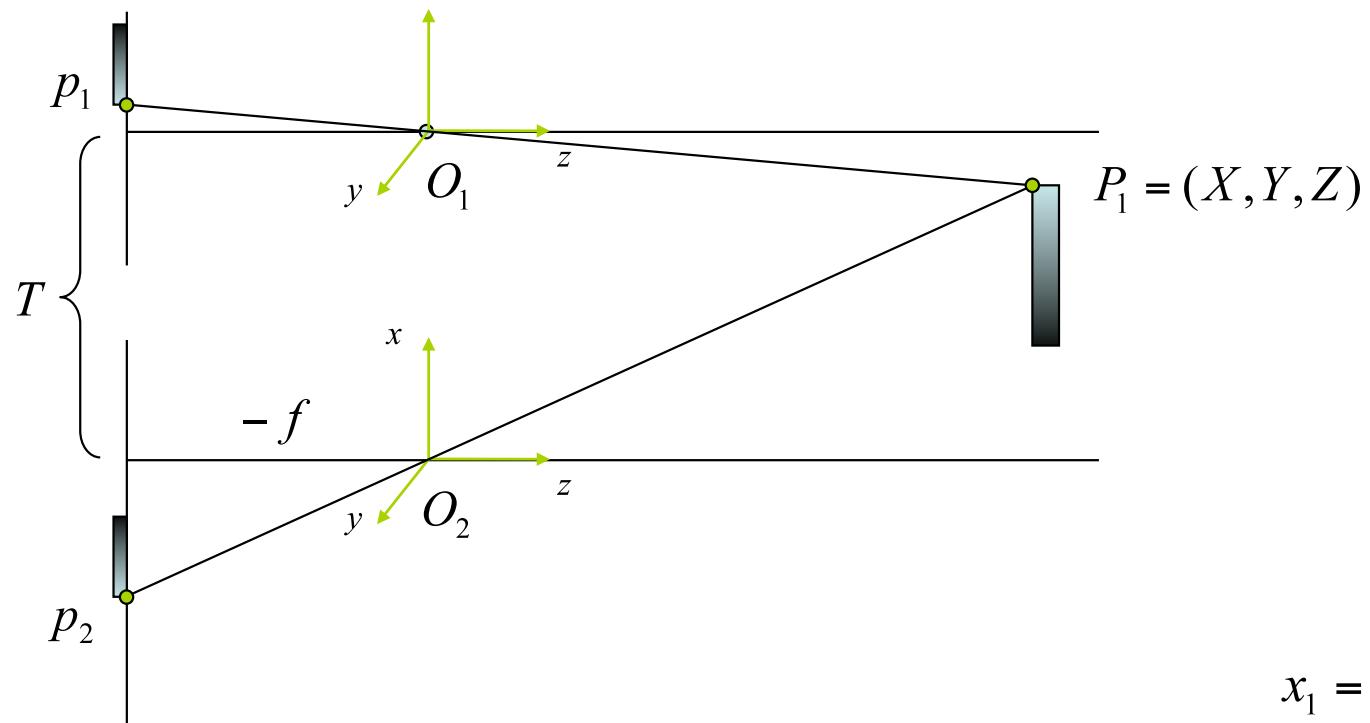


- Zwei Kameras:



- Finde einen Ausdruck für  $Z$  als  $f(x_1, x_2, f, T)$

► Zwei Kameras:

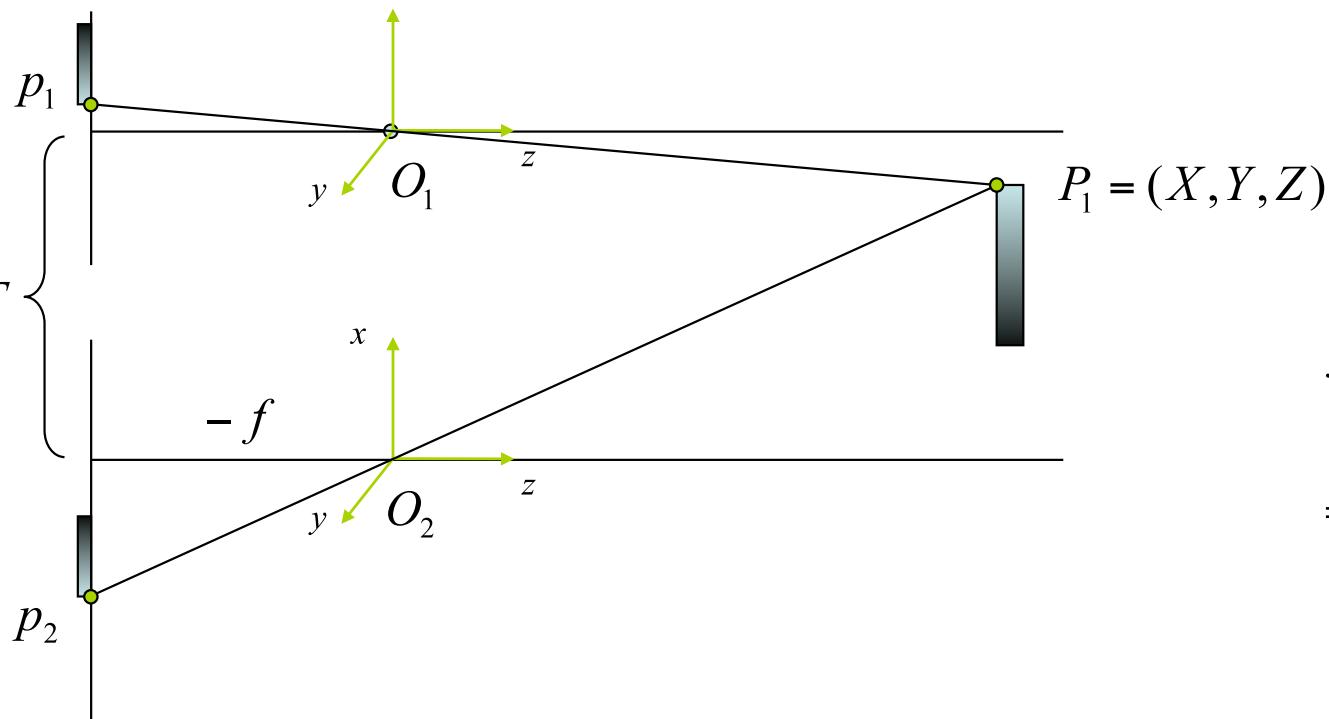


$$x_1 = -f \frac{X_1}{Z_1}, \quad x_2 = -f \frac{X_1 + T}{Z_1} = x_1 - f \frac{T}{Z_1}$$

$$\Rightarrow Z_1 = \frac{fT}{x_1 - x_2}$$

# Stereo - Disparität (1)

- Zwei Kameras:

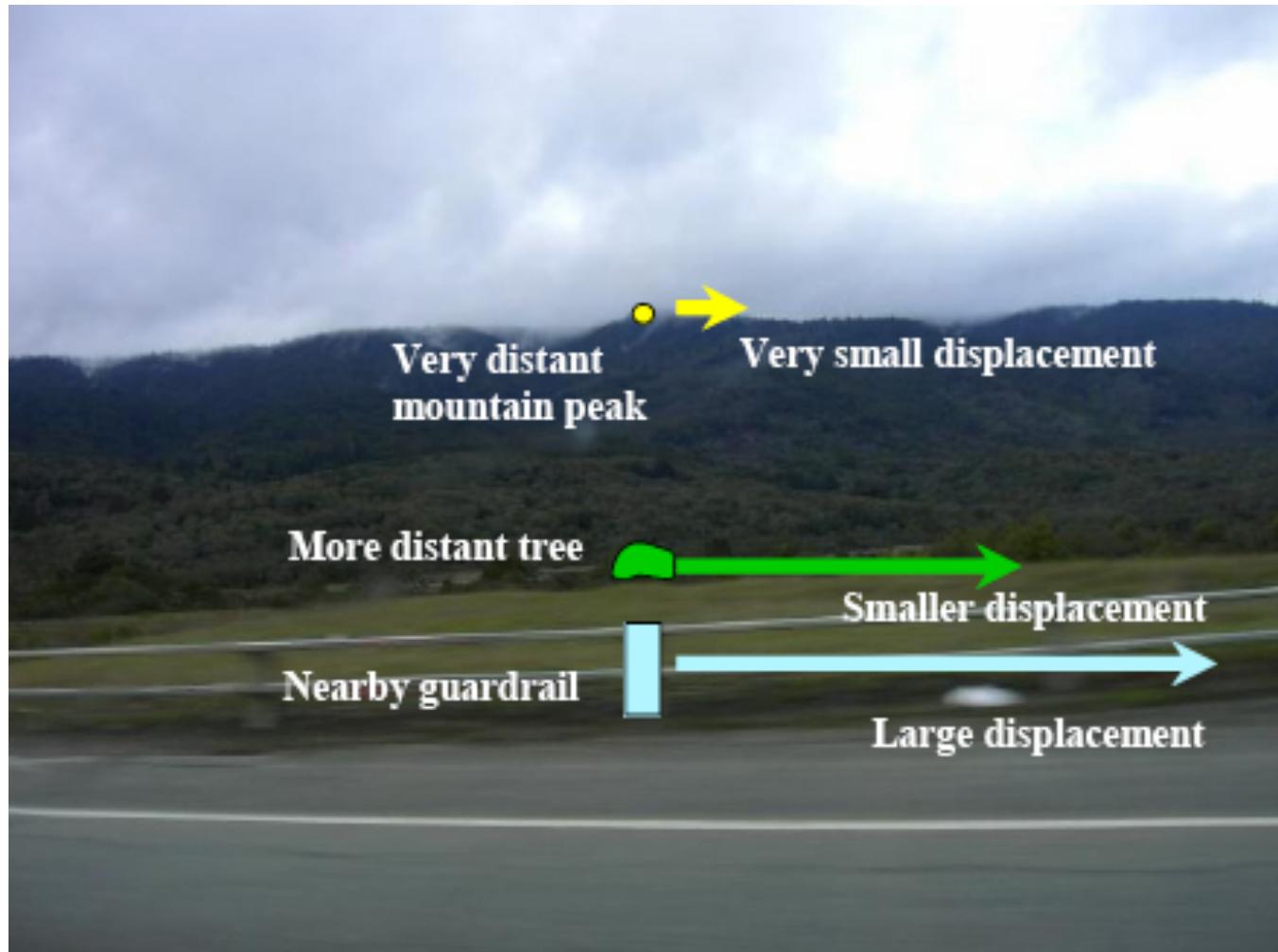


$$x_1 = -f \frac{X_1}{Z_1}, \quad x_2 = -f \frac{X_1 + T}{Z_1} = x_1 - f \frac{T}{Z_1}$$

$$\Rightarrow Z_1 = \frac{fT}{x_1 - x_2} = \frac{fT}{d}$$

Die erreichbare **Tiefenauflösung** hängt im Wesentlichen von der **Baseline  $T$**  und der detektierbaren **Disparität  $d = x_1 - x_2$**  ab.

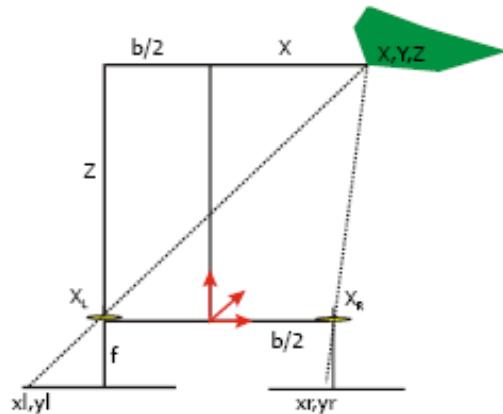
## Stereo - Disparität (2)



## Stereo - Disparität (3)



# Stereo - Auflösung (1)

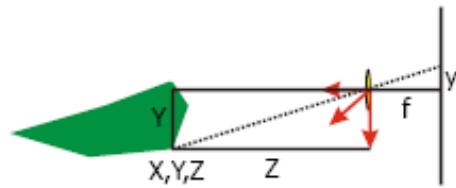


$$\frac{X + T/2}{Z} = \frac{x_l}{f} \Leftrightarrow X = \frac{Zx_l}{f} - \frac{T}{2}$$

$$\text{mit } Z = \frac{Tf}{x_l - x_r} \text{ ergibt sich: } X = \frac{Tfx_l}{f(x_l - x_r)} - \frac{T}{2} = \frac{Tx_l}{x_l - x_r} - \frac{T}{2}$$

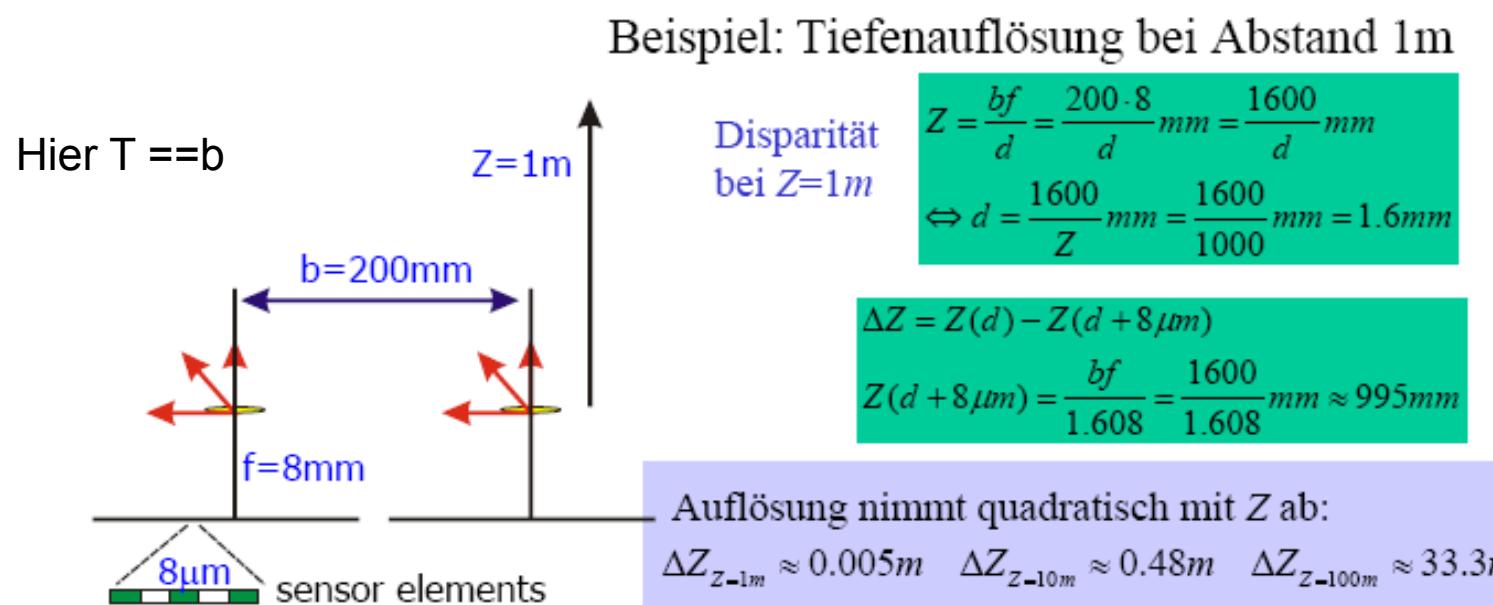
$$X = \frac{T}{2} \left( \frac{2x_l}{x_l - x_r} - \frac{x_l - x_r}{x_l - x_r} \right) = \frac{T}{2} \left( \frac{x_l + x_r}{d} \right)$$

$$\frac{Y}{Z} = \frac{y}{f} \Leftrightarrow Y = \frac{Zy}{f} = \frac{Tfy}{f(x_l - x_r)} = \frac{T}{2} \left( \frac{2y}{d} \right)$$

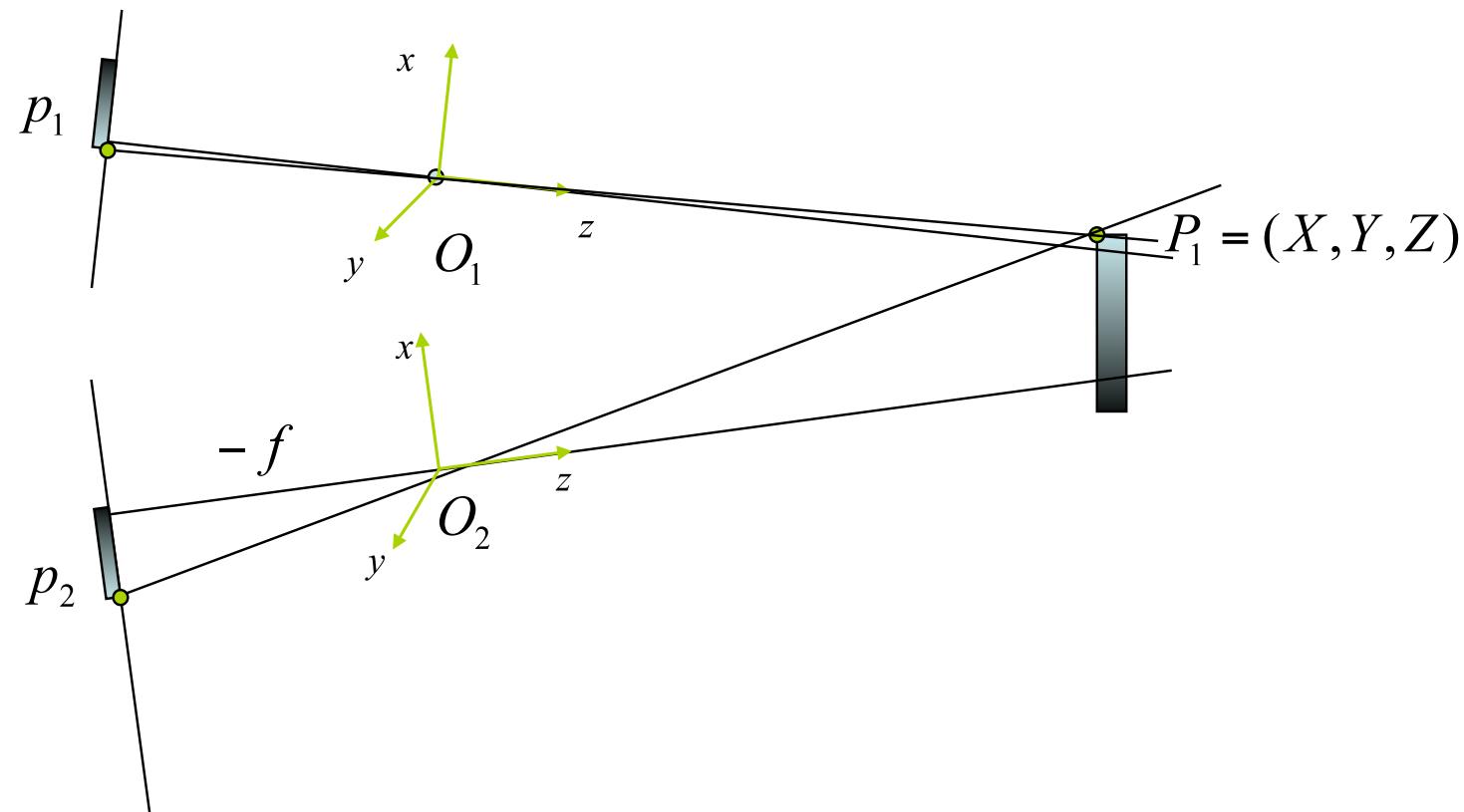


## Stereo - Auflösung (2)

$$Z = \frac{Tf}{d} \quad X = \frac{T}{2} \left( \frac{x_l + x_r}{d} \right) \quad Y = \frac{T}{2} \left( \frac{2y}{d} \right)$$

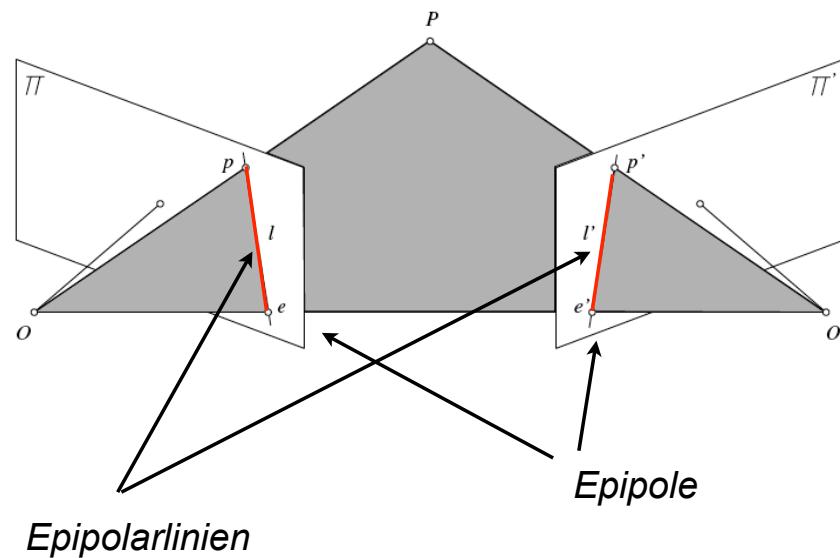


# Stereo - Die Realität

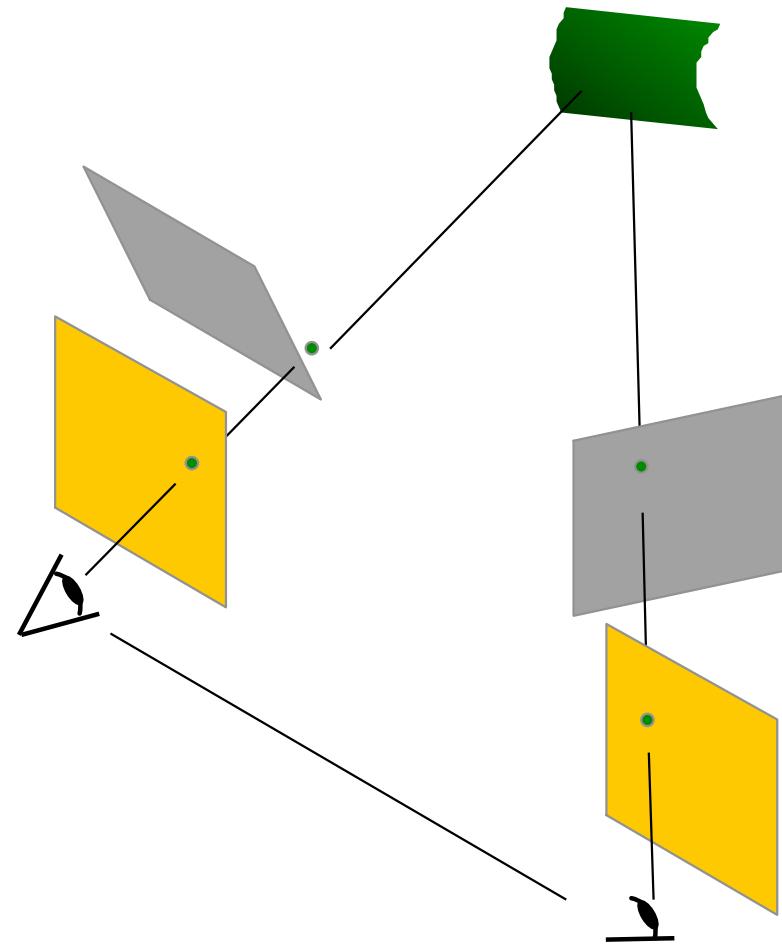


# Epipolargeometrie

- Die Suche nach Korrespondenzen kann auf die Epipolarlinien beschränkt werden



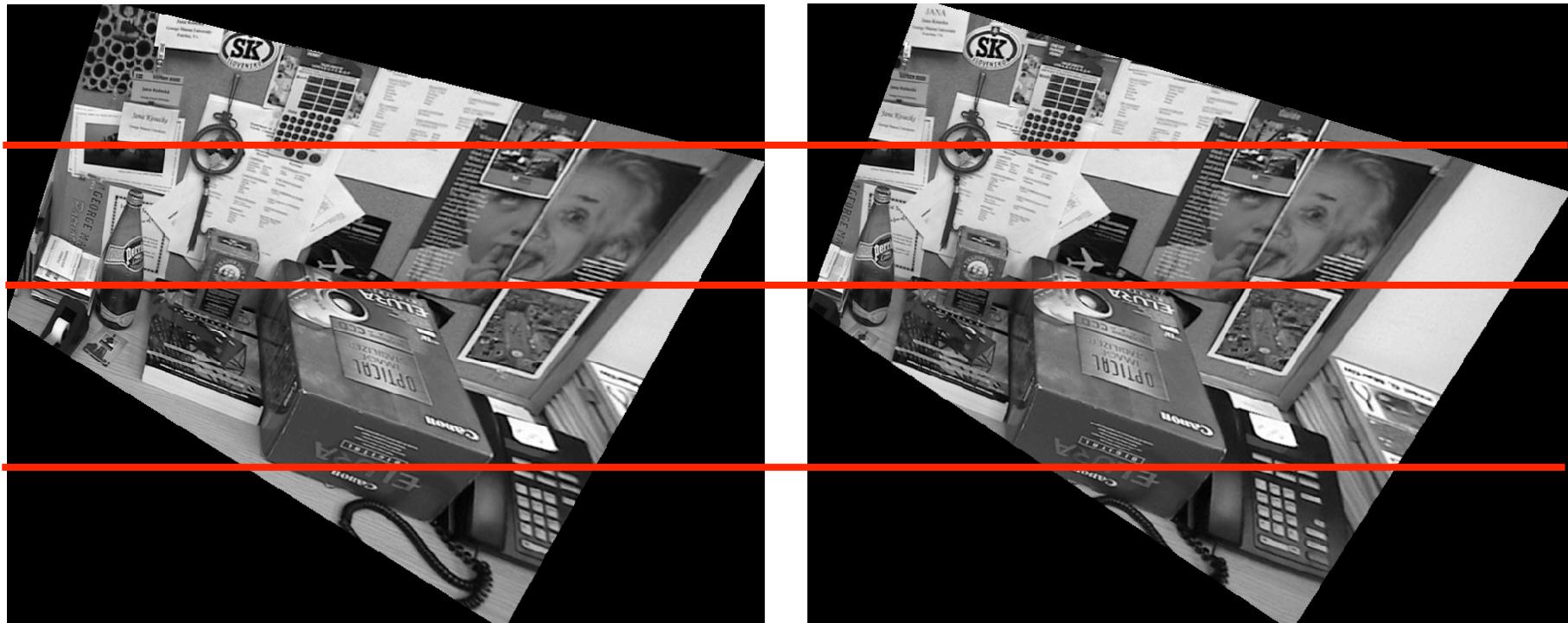
# Rektifizierung (1)



[Seitz]

## Rektifizierung (2)

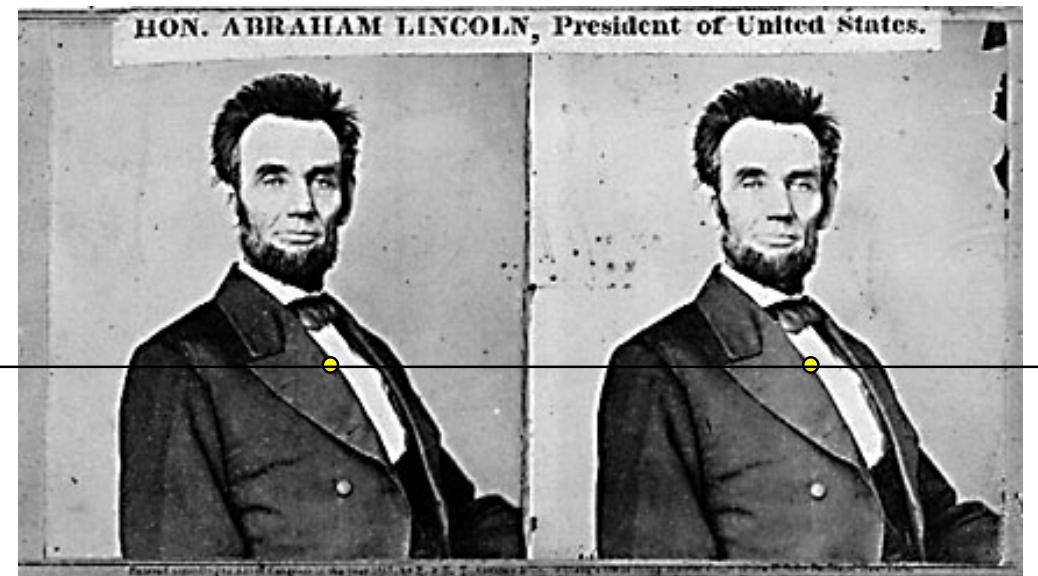
- ▶ Beispiel für ein rektifiziertes Bildpaar
- ▶ Epipolarlinien sind horizontal
- ▶ Korrespondenzsuche wird zu einem 1D-Problem



# Stereomatching

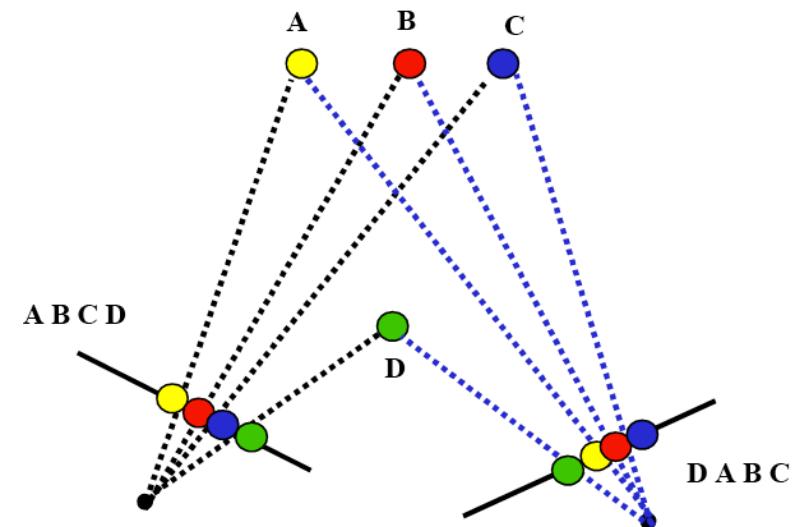
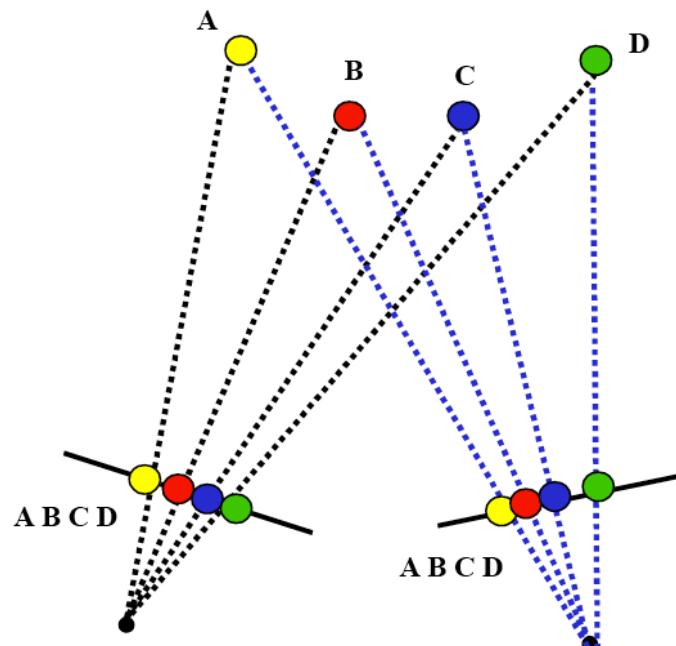
► Naiver Ansatz:

- Vergleiche alle Pixel der Epipolarlinie im linken Bild mit denen auf der entsprechenden Linien im Linken
- Wähle den Pixel mit minimalen Fehler
- ...geht so natürlich nicht, daher werden Regionen verglichen

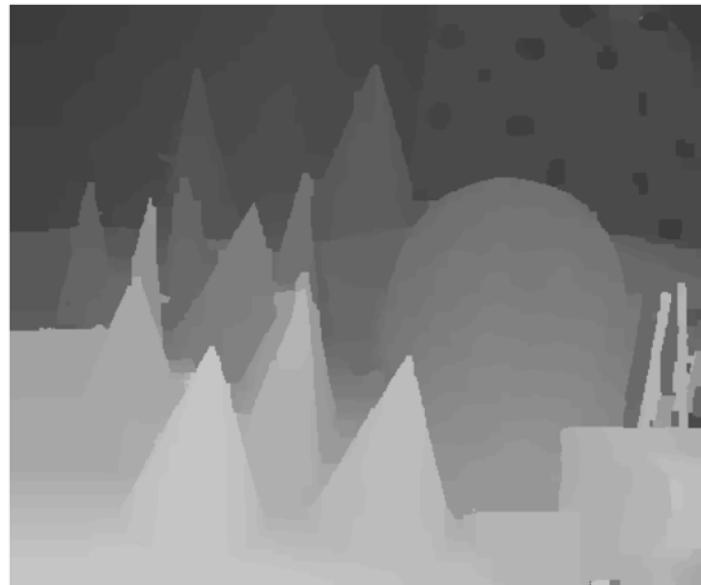


# Stereomatching

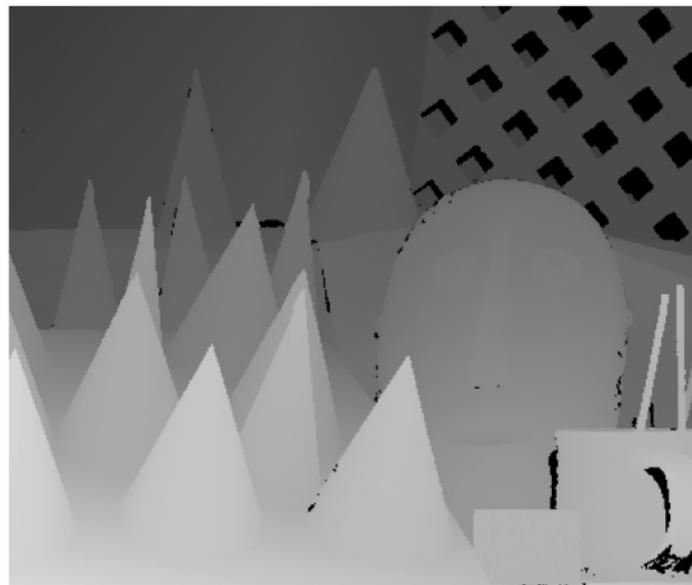
- ▶ Die Suche nach der besten Korrespondenz kann durch „Dynamic Programming“ realisiert werden
- ▶ Suche durch Randbedingungen, hier Punktanordnung, beschränken
- ▶ Verhindert Zyklen



## Stand der Technik (2007)



Algorithm Results



Ground truth

J. Sun, Y. Li, S.B. Kang, and H.-Y. Shum.  
“Symmetric stereo matching for occlusion handling”.  
IEEE Conference on Computer Vision and Pattern  
Recognition, June 2005.

## Zusammenfassung

- ▶ Stereokameras sind günstig realisierbar
- ▶ Gute Resultate für 3D Wahrnehmung im Nahbereich
- ▶ Software für alle notwendigen Schritte ist frei Verfügbar
- ▶ Aber:
  - Saubere Kalibrierung ist komplex und fummelig
  - Das Problem der Korrespondenzsuche beim Stereo-Matching ist komplex
  - Off-the-shelf-Lösungen lassen sich die Präzision und Algorithmik bezahlen
- ▶ Oft auch “aktives Stereo”, Kombination von Stereo-Ansätzen mit Referenzmustern
  - z.B. Kinect-Kamera

# Inhalt



## Gliederung

1. Einleitung
2. Robot Operating System
3. Sensorik
4. **Sensordatenverarbeitung**
5. Fortbewegung
6. Lokalisierung
7. Mapping
8. Navigation
9. Ausblick



## Gliederung

4. Sensordatenverarbeitung
- 2.1. Entfernungsdaten
- 3.2. Bildverarbeitung
4. Sensordatenverarbeitung
5. Fortbewegung
6. Lokalisierung
7. Mapping
8. Navigation
9. Ausblick

## Worum geht es

- ▶ Sensordaten sind i.A. fehlerhaft und (zu) viele
- ▶ Verfahren zur direkten Daten- und Fehlerreduktion
- ▶ Geringer Rechenaufwand Voraussetzung (online-fähig)
  - ▶ Rechenintensive Verfahren auf (riesen)großen Datensätzen in Nachbearbeitung off-line und/oder parallel
  - ▶ Oder auch Mischung (Edge vs. Cloud)
- ▶ Grob zwei Klassen:
  - ▶ Filter (primär Fehlerkorrektur) werfen bewusst Daten weg!
  - ▶ Merkmalsdetektoren (primär Datenreduktion) aggregieren Daten
- ▶ Verwende später (gefilterte) Originaldaten und/oder aggregierte Merkmale
- ▶ Hier:
  - ▶ Entfernungsdaten (Laserscans) und Bilddaten
  - ▶ Hauptsächlich 2D-Daten
  - ▶ 3D ➔ Masterveranstaltung 3D-Sensordatenverarbeitung



## Gliederung

4. Sensordatenverarbeitung
1. Entfernungsmessung
2. Robot Operating System
3. Sensorik
2. Bildverarbeitung
4. Sensordatenverarbeitung
5. Fortbewegung
6. Lokalisierung
7. Mapping
8. Navigation
9. Ausblick

## Reduktionsfilter

- ▶ Solange für Sequenz von Messwerten  $a_i, \dots, a_j$  gilt:  $\|a_i, a_j\| \leq \delta(i - j)$ , ersetze  $a_i, \dots, a_j$  durch

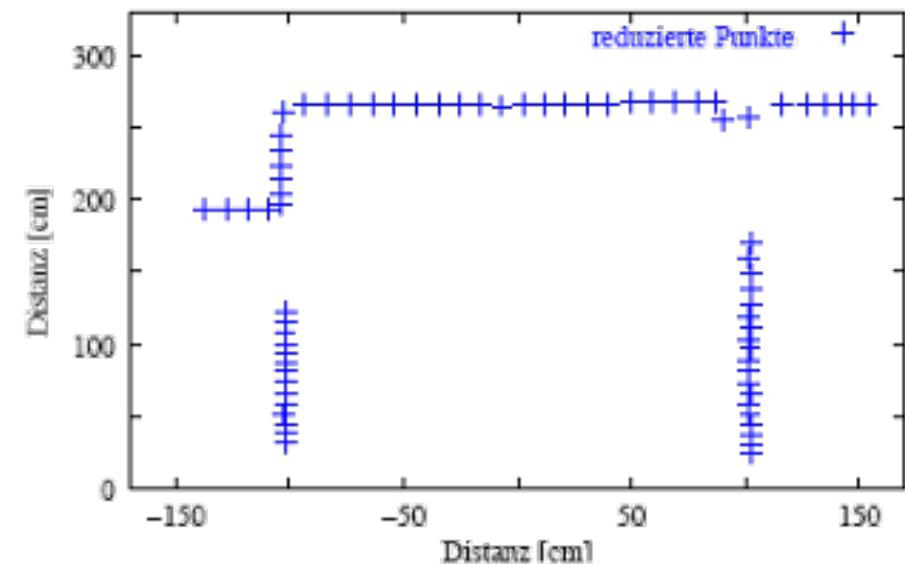
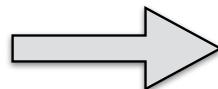
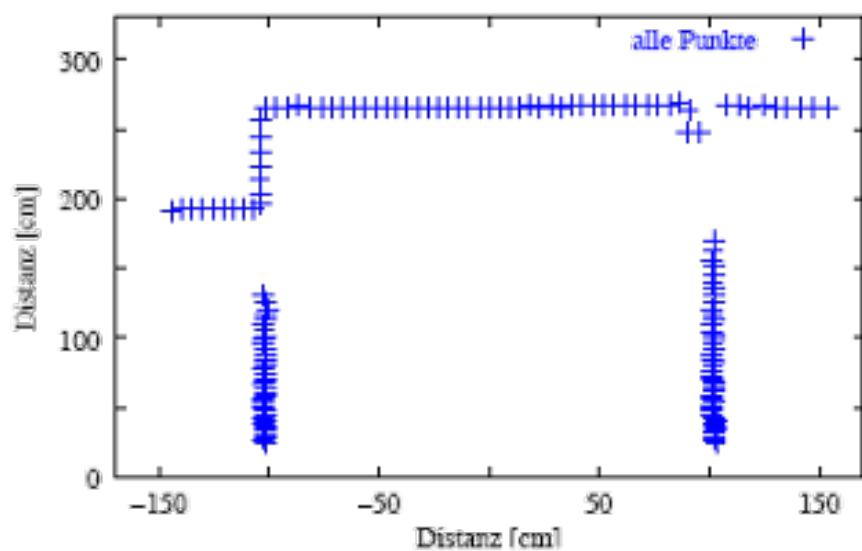
$$\frac{1}{j-i+1} \sum_{k=i}^j a_k$$

Ersetze eine Schar nah aneinander liegender Messwerte durch ihren Mittelwert

- ▶ Voraussetzungen:
  - ▶ Messwerte geordnet (wie beim Laserscanner)
  - ▶ Metrik  $\|\cdot, \cdot\|$  auf Messwerten
  - ▶  $\delta$ -Schranke (hängt möglicherweise von der Punktzahl ab)

# Eigenschaften des Reduktionsfilters

► Beispiel:



- Falls N Messwerte geordnet vorliegen (Laserscanner!), Zeitkomplexität  $\mathcal{O}(n)$
- Reduktionsfilter glättet als Nebeneffekt Messwertrauschen
- Ausreißer bleiben erhalten

## Medianfilter

- ▶ Ersetze jeden Messwert durch den Median seiner Umgebung aus  $k$  Werten:
- ▶ Ersetze für alle  $i$  den Wert  $a_i$  durch

$$\text{Median}\left(a_{i-\lfloor k/2 \rfloor}, \dots, a_i, \dots, a_{i+\lfloor k/2 \rfloor}\right)$$

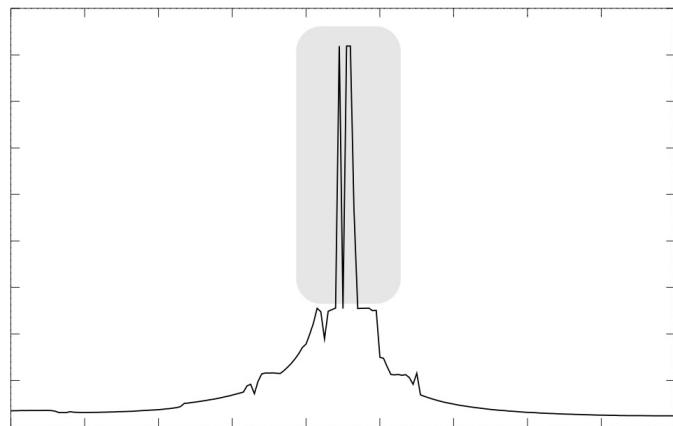
- ▶ Voraussetzungen:
  - ▶  $N$  Messwerte
  - ▶ Metrik auf Messwerten (z.B. Euklidische Distanz)

### Eigenschaften:

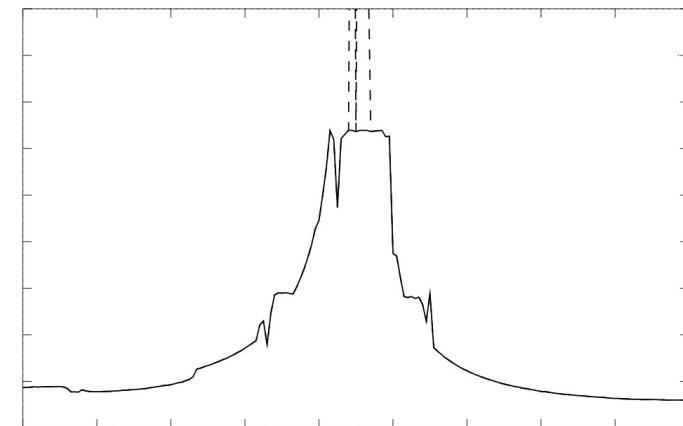
Für  $N$  Messwerte Zeitkomplexität  $\mathcal{O}(N \cdot k \cdot \log k)$  wegen der Sortierung nach Distanz für die Medianberechnung.  
Medianfilter verfälscht “echte” Sprünge in  $< k$  Messwerten.

# Beispiel Medianfilter

Originalscan

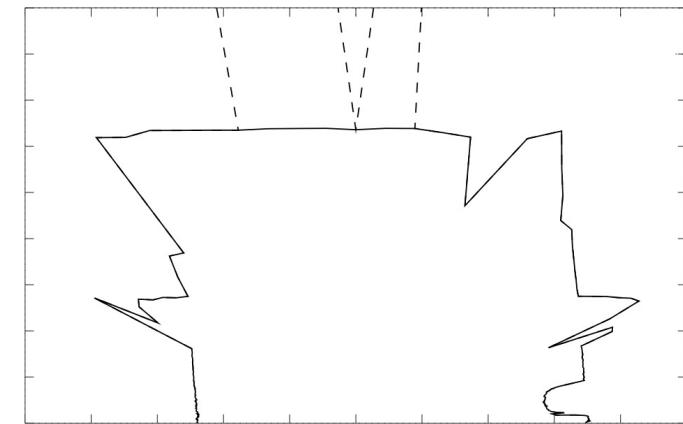
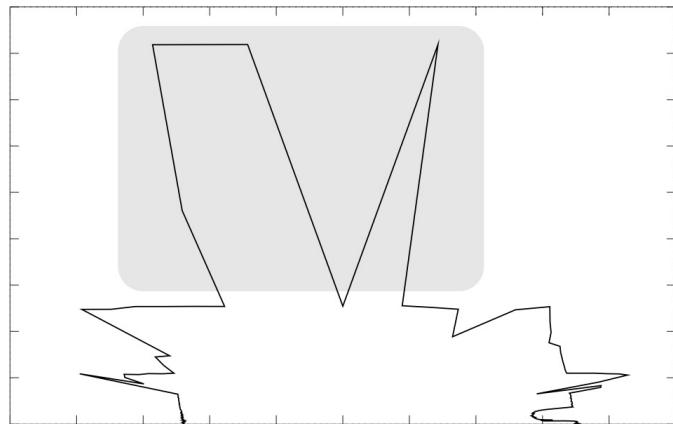


Scan nach Medianfilterung

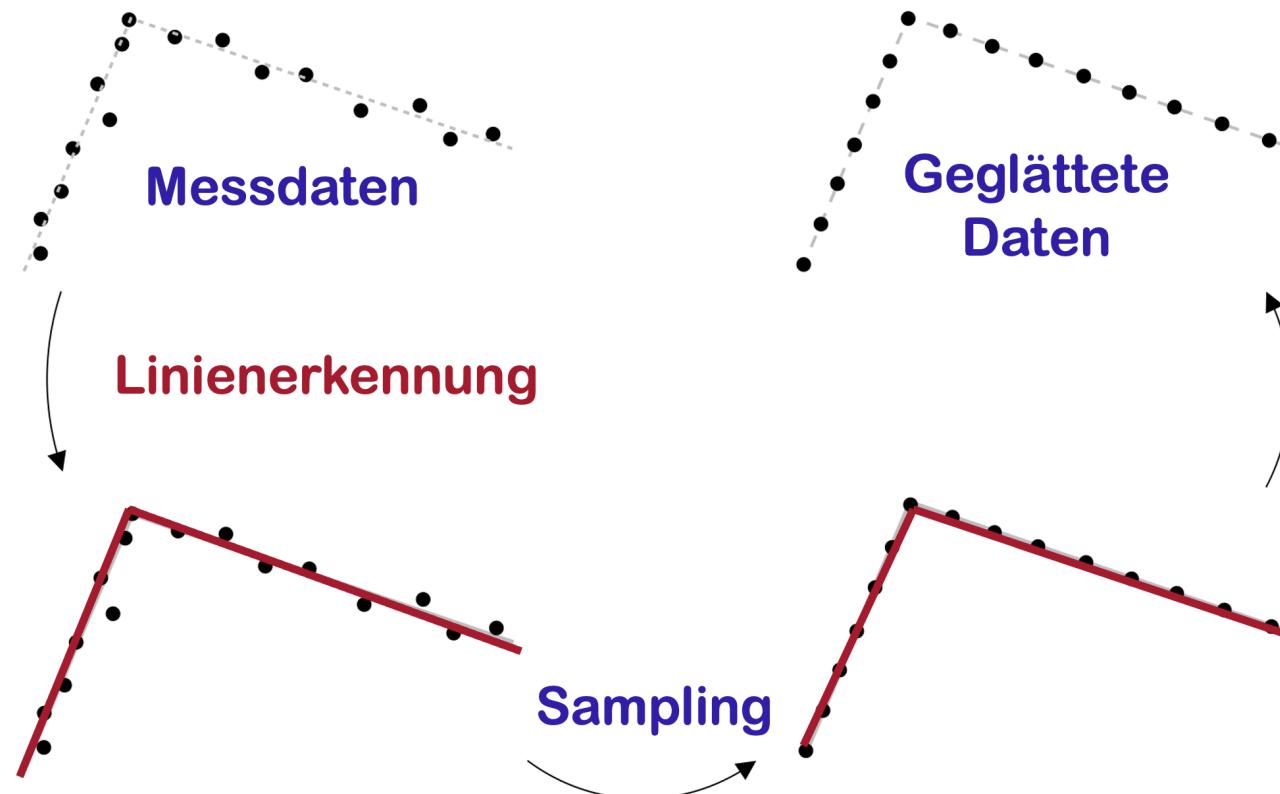


polar

karthesisch



# Merkmale auf Entfernungsdaten: Linien



Linien können als Merkmale dienen und außerdem als Rauschfilter verwendet werden!