Karlsruhe Institute of Technology

Robotik I: Einführung in die Robotik
Umweltmodellierung

Tamim Asfour

KIT-Fakultät für Informatik, Institut für Anthropomatik und Robotik (IAR)
Hochperformante Humanoide Technologien (H^2T)

Inhalt

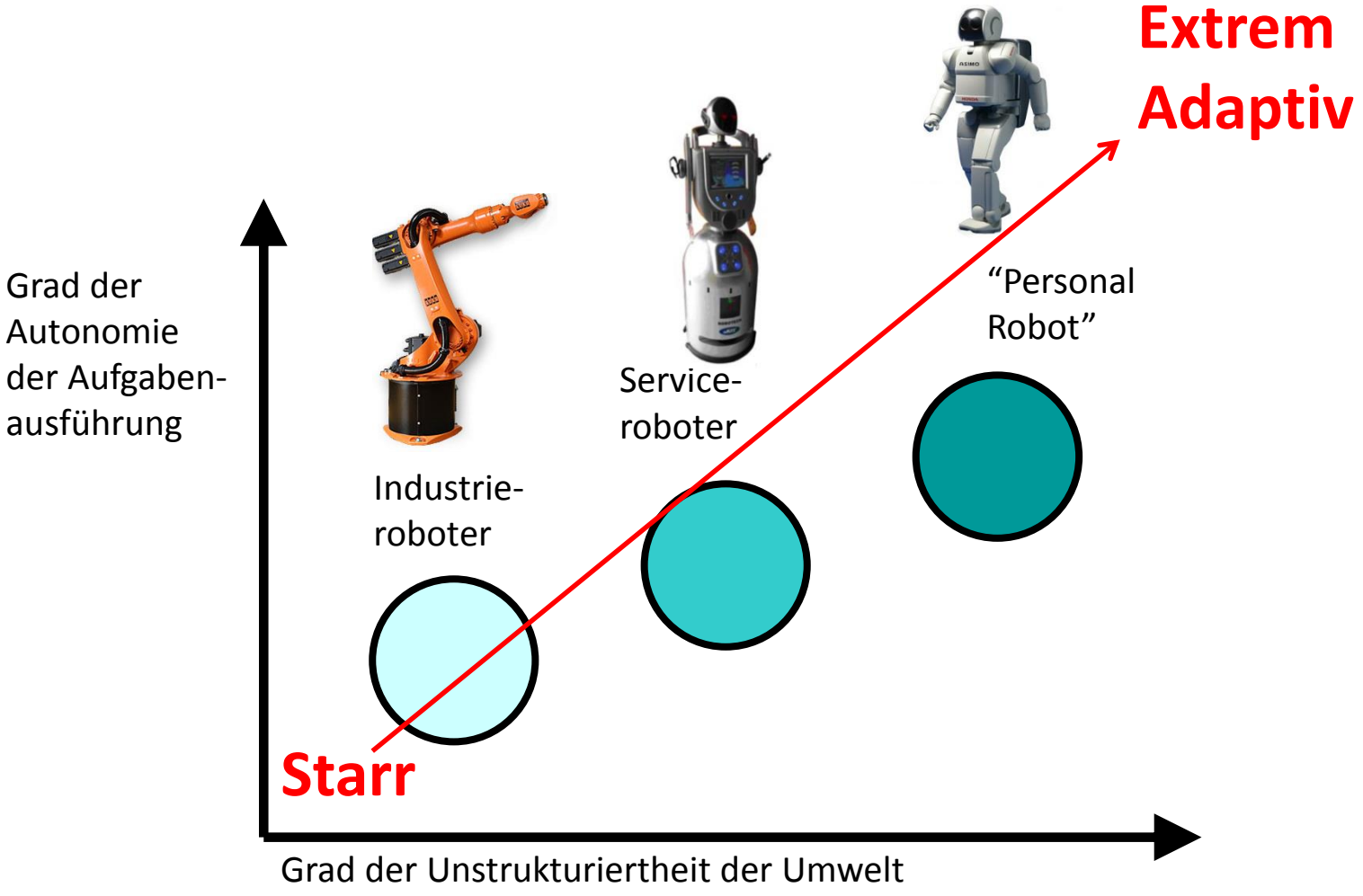
- **Motivation: Adaptive Roboteraufgaben**
- Objektmodelle
  - Geometrische Beschreibung
  - Zusätzliche Eigenschaften
- Szenenmodelle

Bisher: Starre Roboteraufgaben

- **Bisher in VL: un- oder marginal flexible Roboteraufgaben**
  - Feste Trajektorie programmiert:
    - Positionsregelung
    - Eventuell lokale Kraftregelung
    - Benötigte Modelle für Programmierung und Regelung:
      - Kinematisches Modell
      - Dynamisches Modell
    - Benötigte Sensoren:
      - Gelenkencoder
      - (eventuell) Kraft-Momenten-Sensor
  - Bahn absolut fest oder nur lokal-adaptiv, repetitiv ausgeführt

Starre Roboteraufgaben

Adaptionsanforderungen



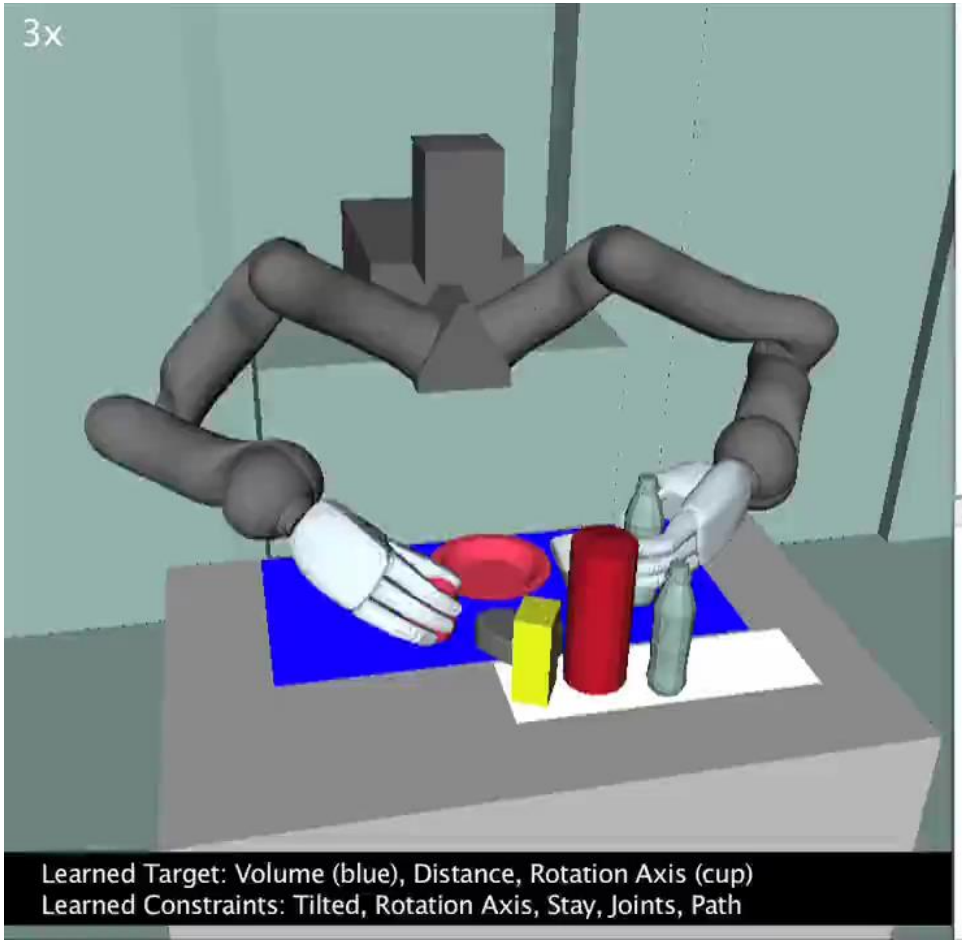
Adaptive Roboteraufgaben

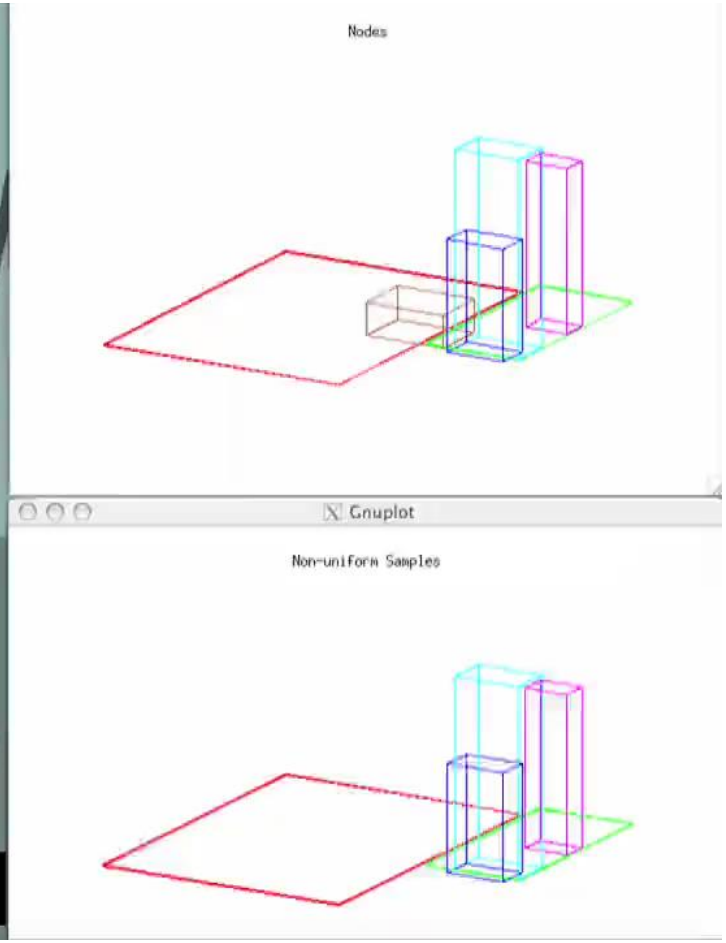
Bei adaptiven Roboteraufgaben andere Voraussetzungen!

- Keine feste Trajektorie programmierbar
  - Trajektorie muss flexibel vom Roboter bestimmt werden
  - Benötigte Modelle für Programmierung und Regelung:
    - Kinematisches Modell
    - Dynamisches Modell
    - **+ Umweltmodelle**
    - **+ Aufgaben/Planungsmodelle**
  - Benötigte Sensoren:
    - Gelenkencoder
    - Kraft-Momenten-Sensor
    - **+ Visuelle Sensoren**
    - **+ Taktile Sensoren**
- Bahn völlig frei, flexibel, nicht-repetitiv ausgeführt

Adaptive Roboteraufgaben

Bei adaptiven Roboteraufgaben andere Voraussetzungen!





Anforderungen für Adaption

Zusätzliche Anforderungen gegenüber starren Aufgaben

1. Visuelle und taktile Sensorik zur Umwelterfassung:
   - *Zentrales Thema der Vorlesung Robotik 3 im SoSe*

2. Datenrepräsentation der Umgebung/Umwelt:
   - *Diese Vorlesung*

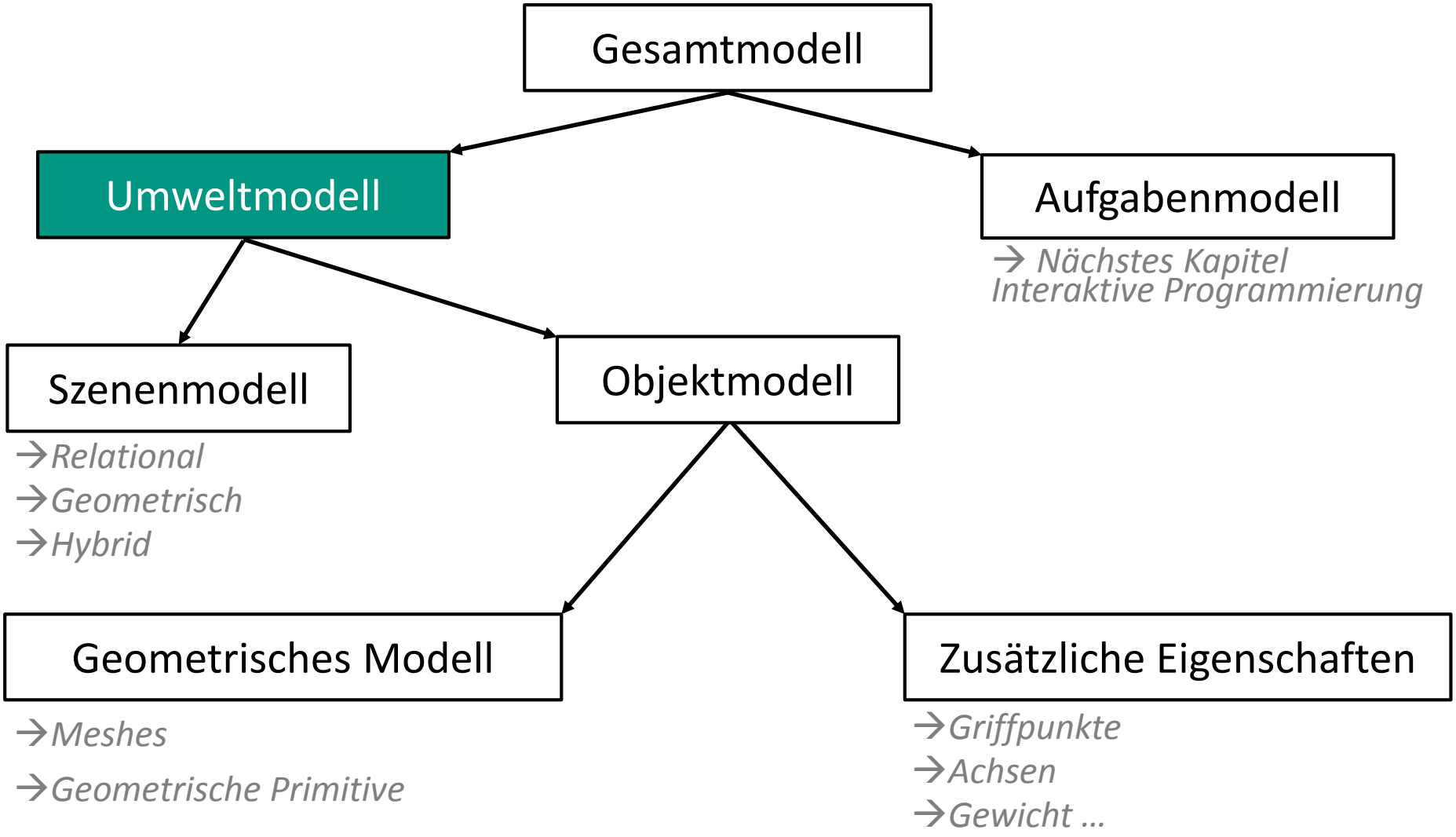
3. Planungsmethoden zur Aufgaben/Bewegungsberechnung
   - *Robotik 1*
     - *Kapitel 8: Bewegungsplanung*
     - *Kapitel 9: Greifplanung*
     - *Kapitel 12: Interaktive Programmierung*

Übersicht Modelle

Verschiedene Arten von Modellen in der Robotik

1. Modell der Kinematik (Manipulator, mobile Plattform)
2. Modell der Dynamik (Manipulator, mobile Plattform)
3. Modelle der Sensoren (Kamera, Kraft-Momenten-Sensor)
4. **Umweltmodell** (Umgebung: Einzelobjekte und Szenen)
5. Aufgabenmodell (Bewegungen und abstrakte Aufgaben)

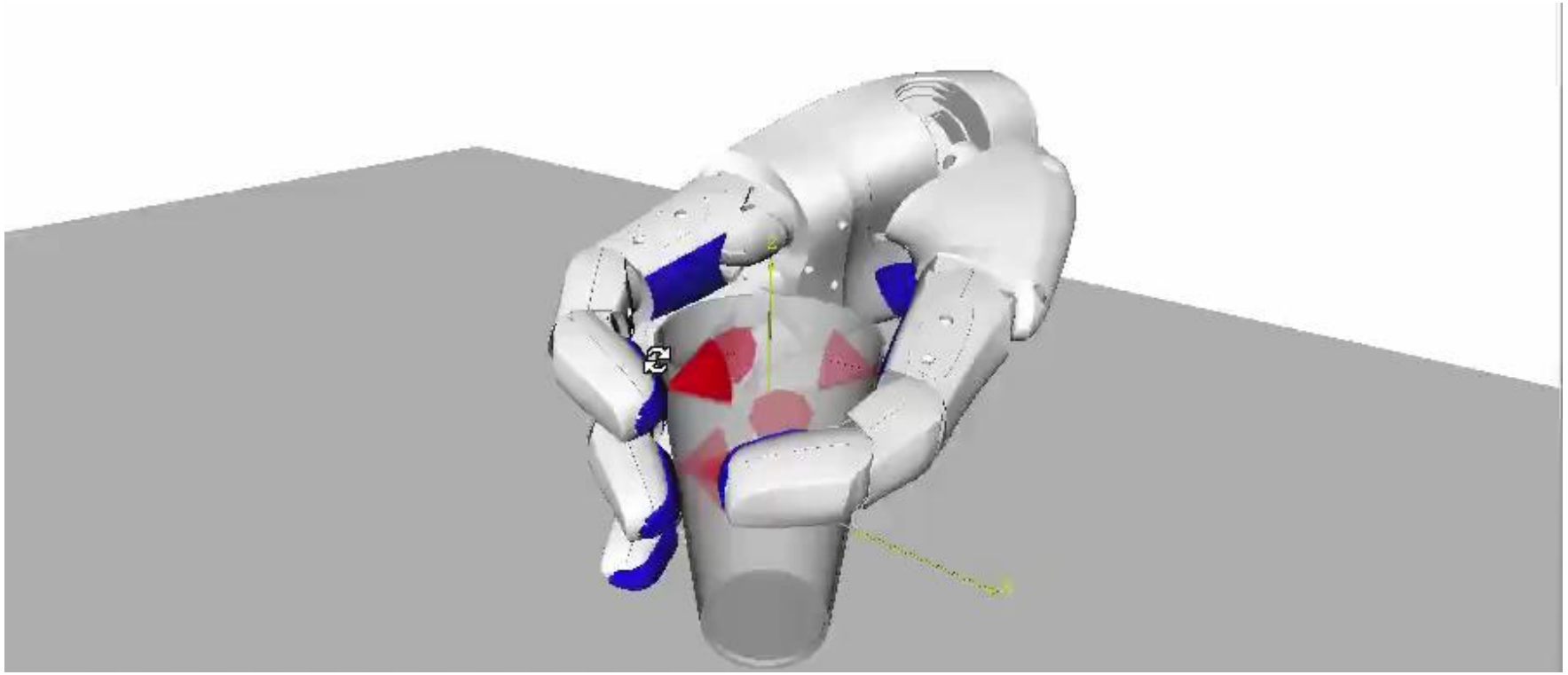
Übersicht Umweltmodellierung



Inhalt

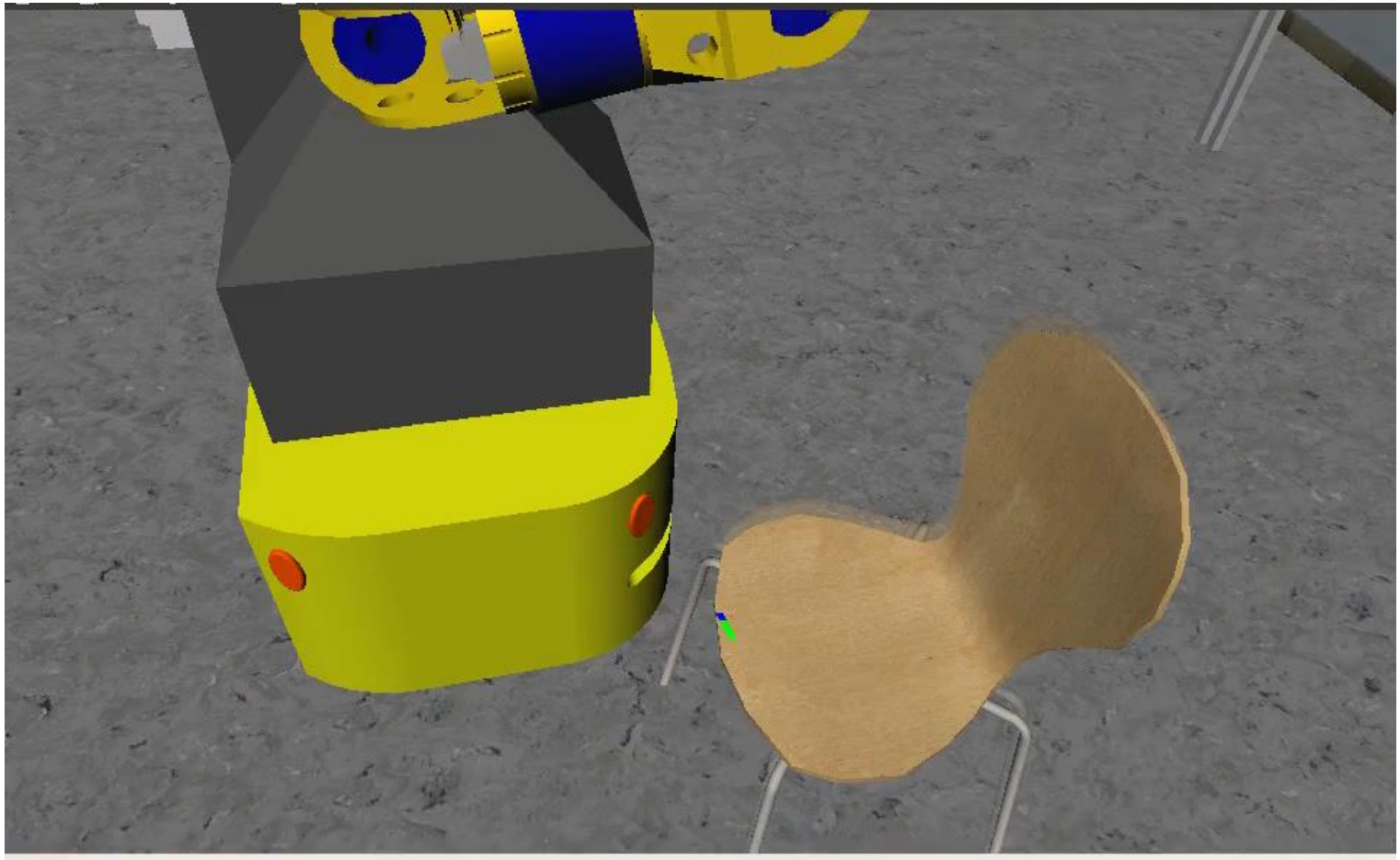
- Motivation: Adaptive Roboteraufgaben
- **Objektmodelle**
  - **Geometrische Beschreibung**
  - Zusätzliche Eigenschaften
- Szenenmodelle

Geometrisches Modell: Motivation



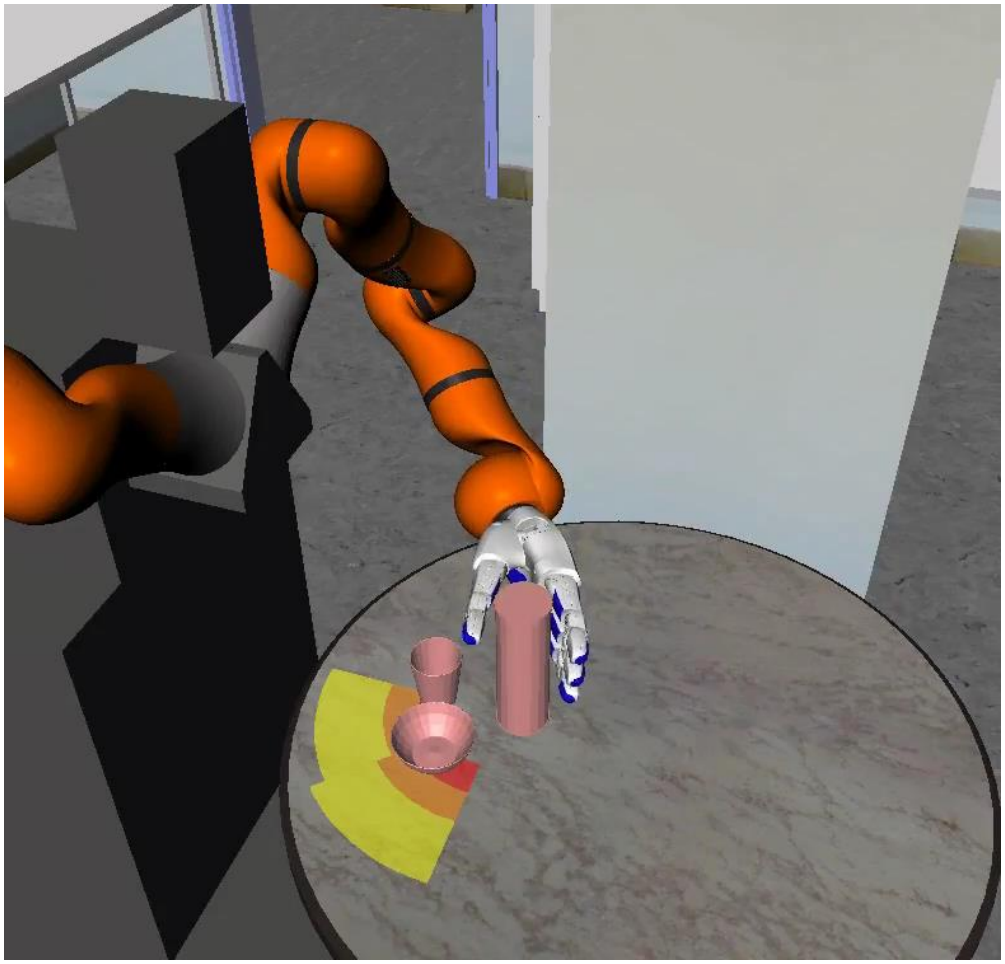
Geometrisches Modell: Motivation

Simulation der Effekte von Handlungen auf die Umwelt



Geometrisches Modell: Motivation

Simulation der Effekte von Handlungen auf die Umwelt



Geometrisches Modelle: Motivation

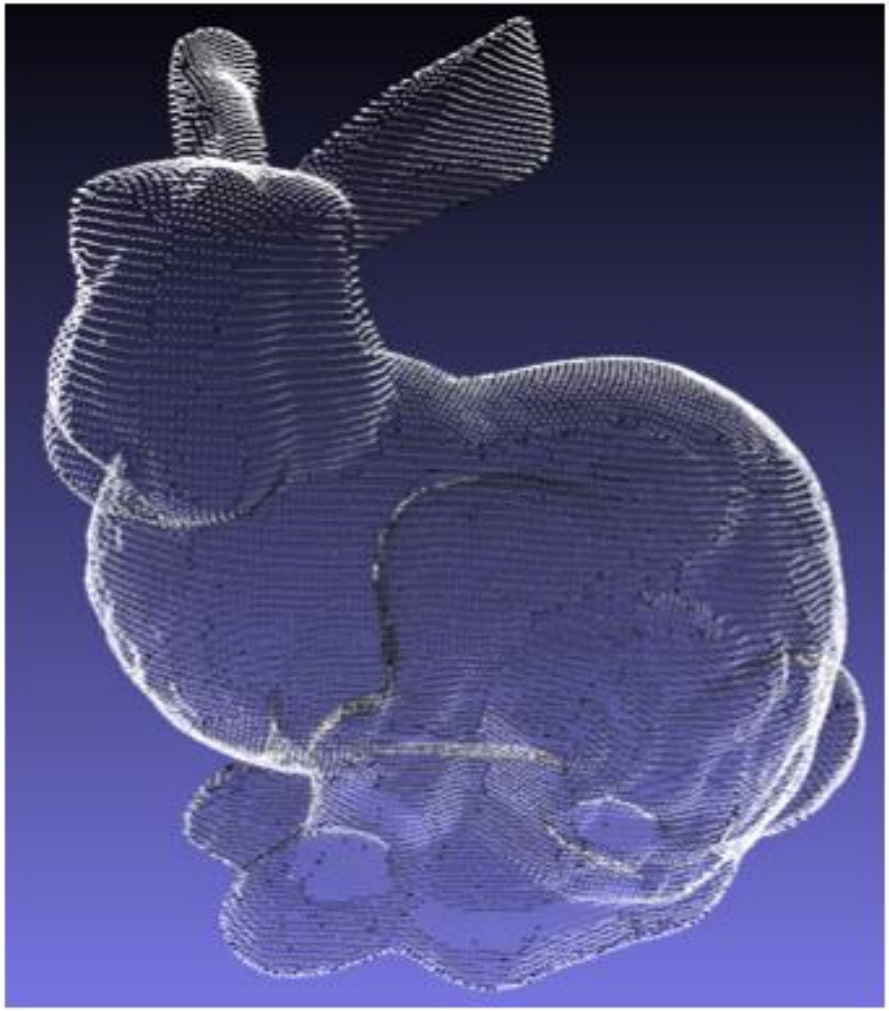
Anwendungsgebiete für das geometrische Modell

- Umweltwahrnehmung (Live)
  - Objektklassifizierung
  - Objektlokalisierung
  - Erkennung und Klassifikation von menschlichen Bewegungen
- Bewegungsplanung (Live, offline)
  - Pfadplanung (Baumsuche, A*)
  - Bewegungsplanung (PRM, RRT)
  - Greifplanung und Planung von Inhandmanipulationsaufgaben
- Abstrakte Aufgabenplanung (Live, offline)
- Dynamische Effektsimulation und –prädiktion (Live, offline)
- Bestimmung des Arbeitsraums (Offline)

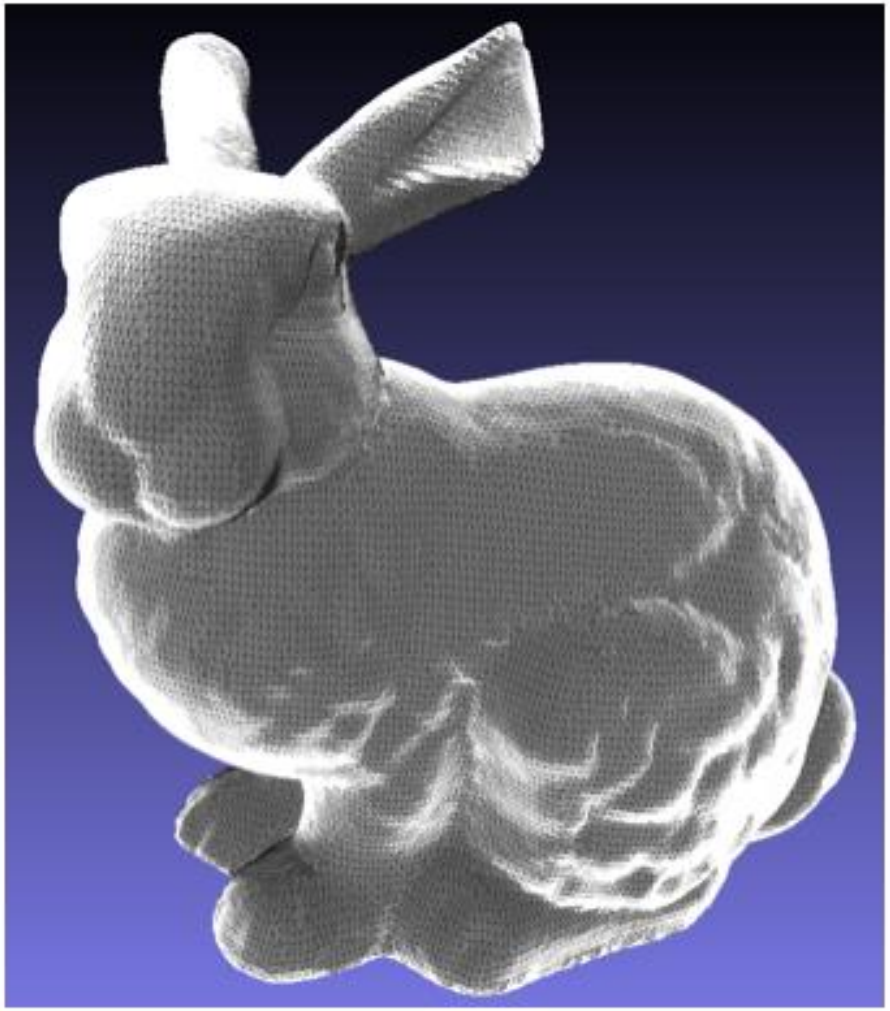
Geometrische Modelle: Übersicht

- Punktwolken
- Approximative Oberflächenmodelle (B-Rep)
  - Dreiecksflächen (Meshes)
  - Viereckflächen
  - Bezierflächen
- Approximative Zellenbelegung
  - Voxel
  - Octree
- Analytisch-parametrische Modelle
  - Constructive Solid Geometry (CSG)

Punktwolken: Beispiel



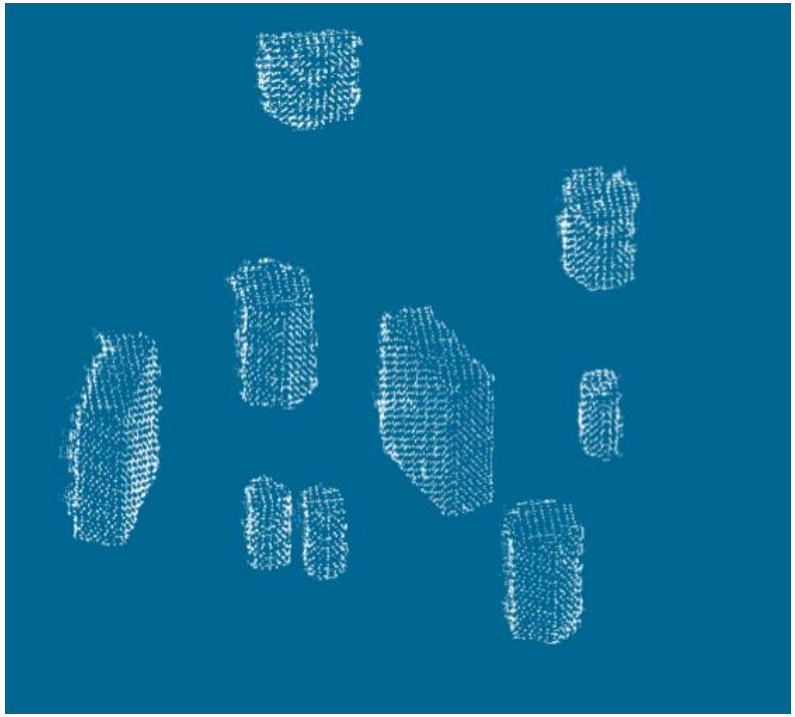
Stanford Bunny dargestellt mit Punkten



Stanford Bunny dargestellt mit Flächen

Punktwolken: Beispiel

Szene, mit Tiefenkamera aufgenommen

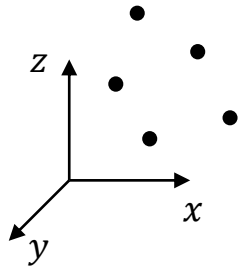


Aufgenommene Punktwolke

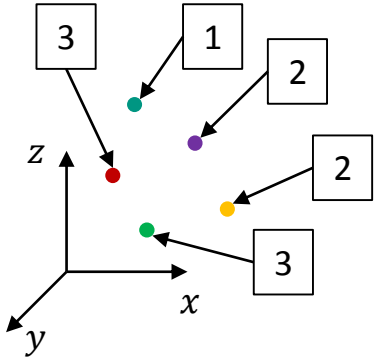
Tiefenkamera

Punktwolken: Repräsentation

- Punkt im 3D-Raum: $p = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3$



- Punktwolke (Point Cloud):
  - Menge von Punkten: $PC = \{p_1, p_2, \ldots, p_n\}$
  - n: Anzahl der Punkte
  - $p_i \in \mathbb{R}^3$: Enthaltene Punkte

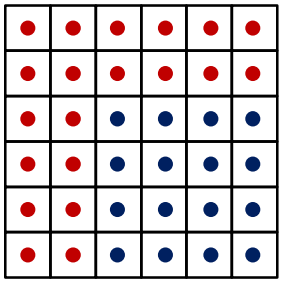


- Zusätzliche Daten pro Punkt
  - Farbe (RGB)
  - Label (1, 2, 3, …)

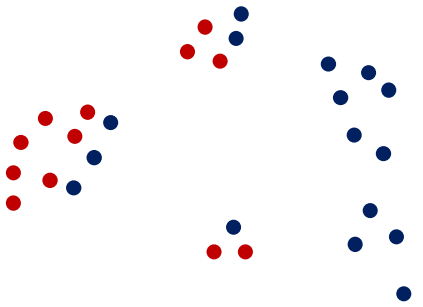
Punktwolken: Dense vs. Sparse

- Zwei Möglichkeiten zur Darstellung der Punktwolken im Speicher

- **Dense**
  - Feste 2D-Gitteranordnung (z. B. 640x480)
  - Ausgabe von Tiefenkameras: 2D-Bild (RGBD)
    ➔ Umwandlung in Punktwolke
  - Punkt kann einem Pixel im Farbbild zugeordnet werden



- **Sparse**
  - Ungeordnete Menge von Punkten
  - Entsteht z. B.
    - Filterung von dichten Punktwolken
    - Sampling von 3D-Modellen



Punktwolken: Segmentierung

- Aufgenommene Punktwolken enthalten keine Zuordnung zu Objekten
- Idee:
  - Bilde pro Objekt eine **Untermenge von Punkten**, die zum Objekt gehören
    - $O_1, O_2, \dots, O_m \subset PC$: Untermengen der Objekte
    - m: Anzahl der Objekte
  - Stelle Objektzugehörigkeit als **Label** (1, 2, …, m) dar
- Umsetzung: Segmentierungsalgorithmen (z. B. LCCP, Region Growing, …)

Punktwolke aufgenommen
mit einer Tiefenbildkamera

Segmentierte Punktwolke
Labels sind farbcodiert

Punktwolken: Eigenschaften

- Darstellung als Menge von Punkten im 3D-Raum mit
  - Farbinformation
  - Label
- Datenquellen
  - Aktuellen Tiefensensoren (3D-Sensoren)
  - Stereokamerasysteme
- Segmentierung auf Quelldaten notwendig
- Objekte sind werden approximiert dargestellt
  - Nur einzelne Punkte sind enthalten, Flächen können geschätzt werden
  - Unter- und Übersegmentierung kann auftreten

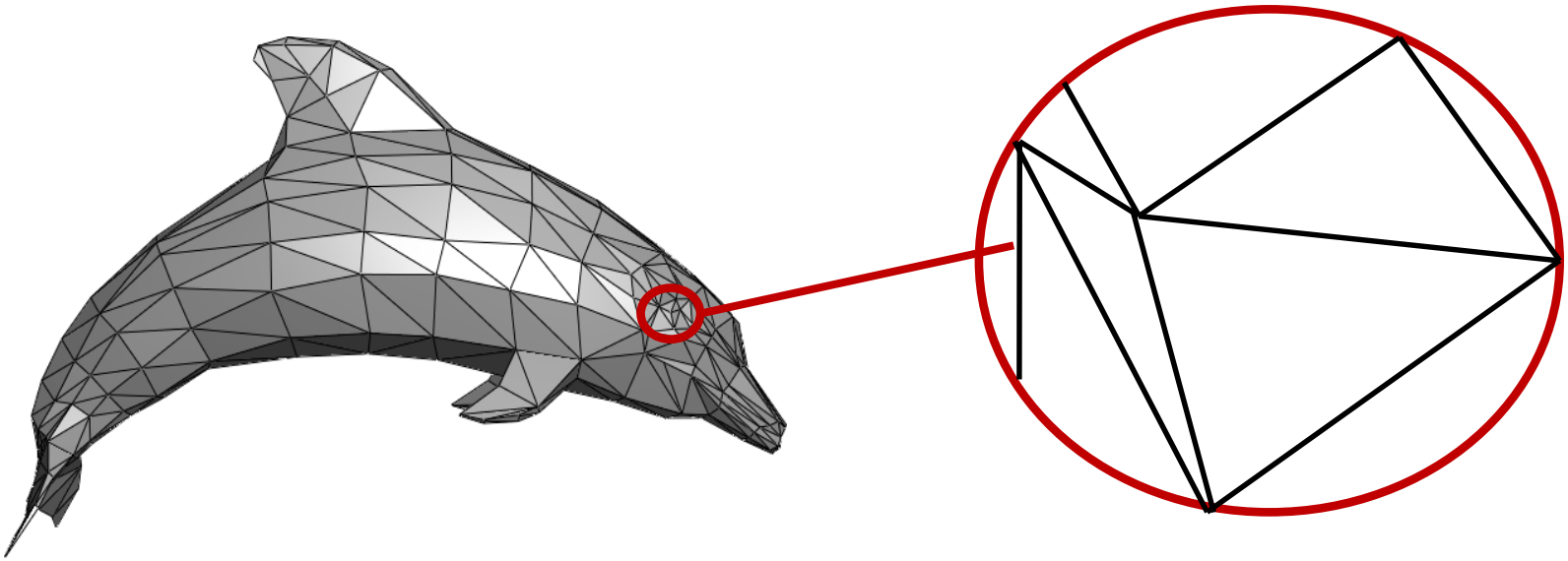
Approximative Oberflächen

- Bildung einer großen Fläche aus einem Netz („Mesh“) von einfachen Einzelflächen, z.B. Dreiecke, Vierecke

- **Vorteile**:
  + Definition sehr einfach
  + einfache Algorithmen
- **Nachteile**:
  - hoher Speicherbedarf
  - hoher Rechenaufwand

Dreiecksflächen: Motivation

- Approximation von Freiformflächen durch **Dreiecke**



Public Domain, Autor: Chrschn,
https://commons.wikimedia.org/wiki/File:Dolphin_triangle_mesh.png

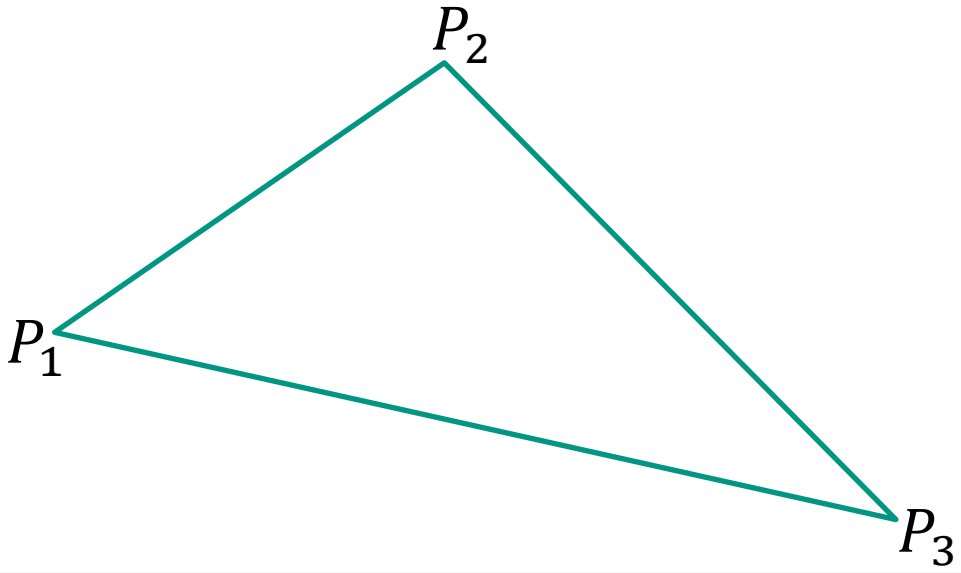
Dreiecksflächen: Definition

- Approximation von Freiformflächen durch **Dreiecke**

- Punkte im Dreieck: **Konvexe Hülle** von drei Punkten:

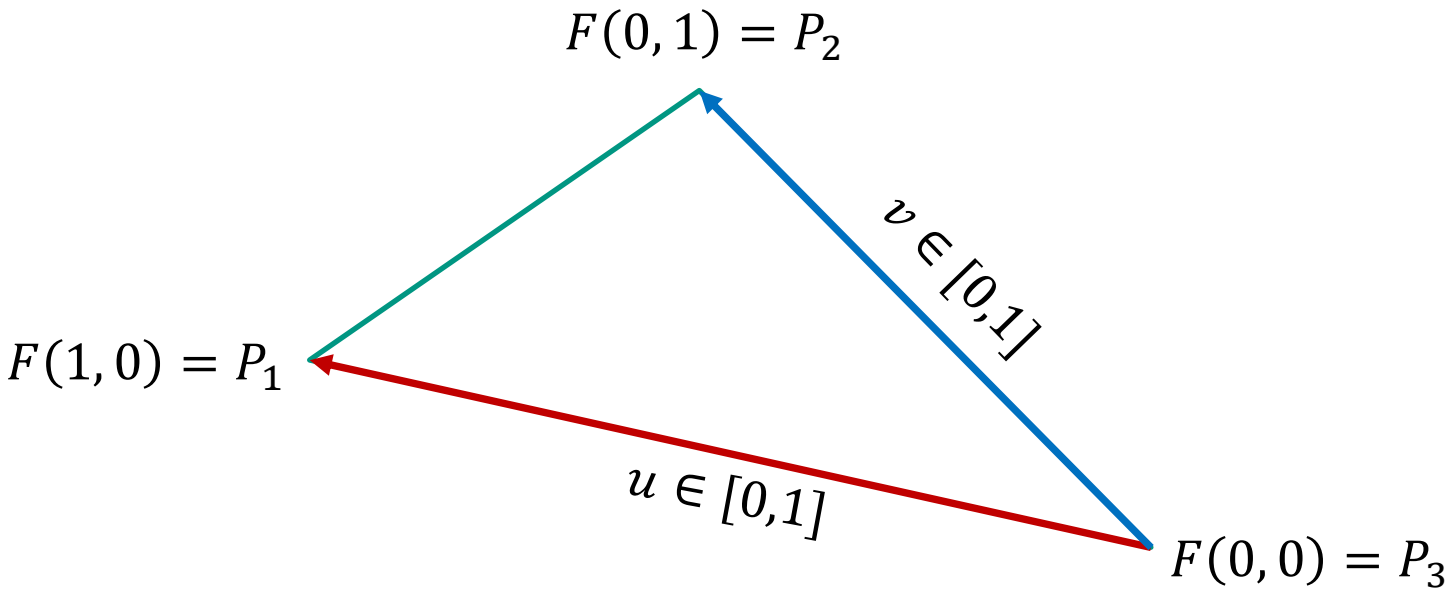
$$A_\Delta = \{ \boldsymbol{p} \in \mathbb{R}^3 \mid \boldsymbol{p} = u \cdot P_1 + v \cdot P_2 + w \cdot P_3, u + v + w = 1 \}$$

- Punkte im Dreieck als Funktion $F(u,v) \in A_\Delta$
  $F(u,v) = u \cdot P_1 + v \cdot P_2 + (1 - u - v) \cdot P_3$ mit $0 \leq u, v, u + v \leq 1$



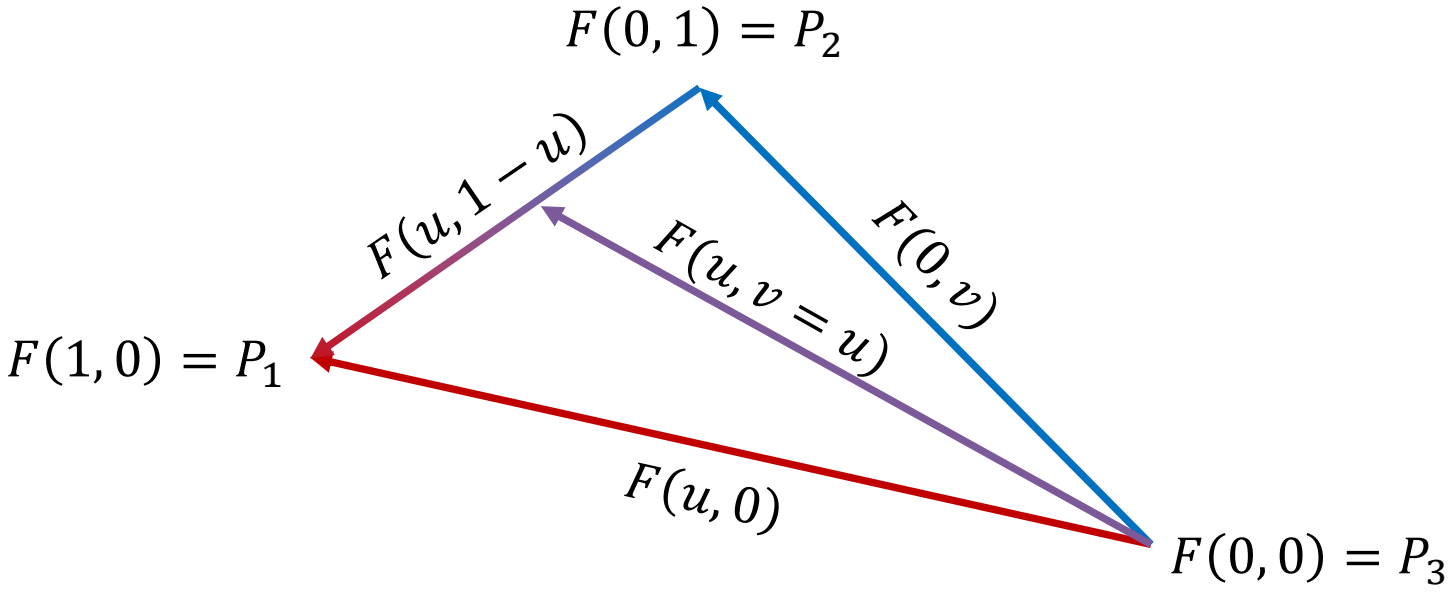
Dreiecksflächen: Darstellung

- Punkte im Dreieck als Funktion $F(u,v) \in A_\Delta$
  $F(u,v) = u \cdot P_1 + v \cdot P_2 + (1-u-v) \cdot P_3$ mit $0 \leq u, v, u+v \leq 1$

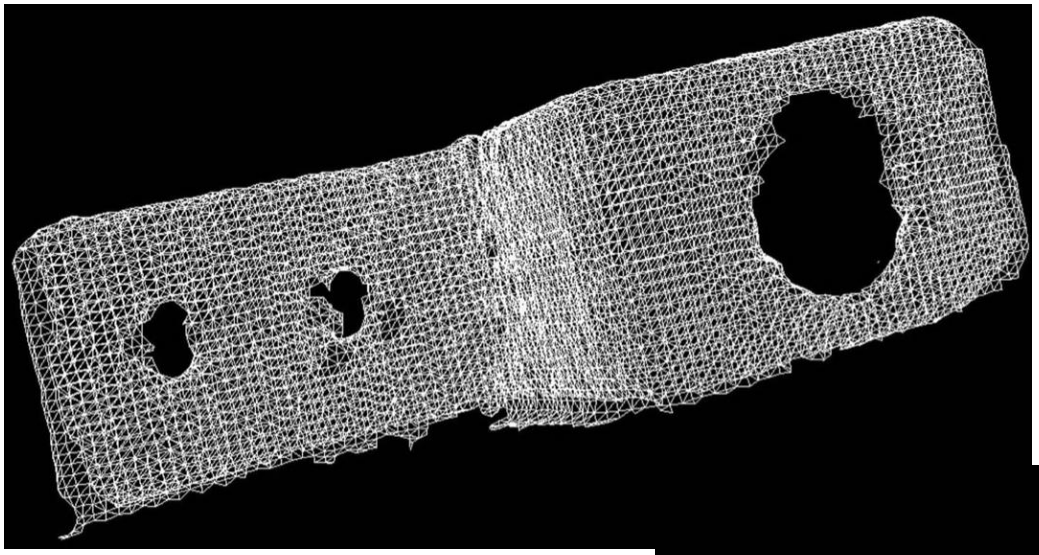


Dreiecksflächen: Darstellung

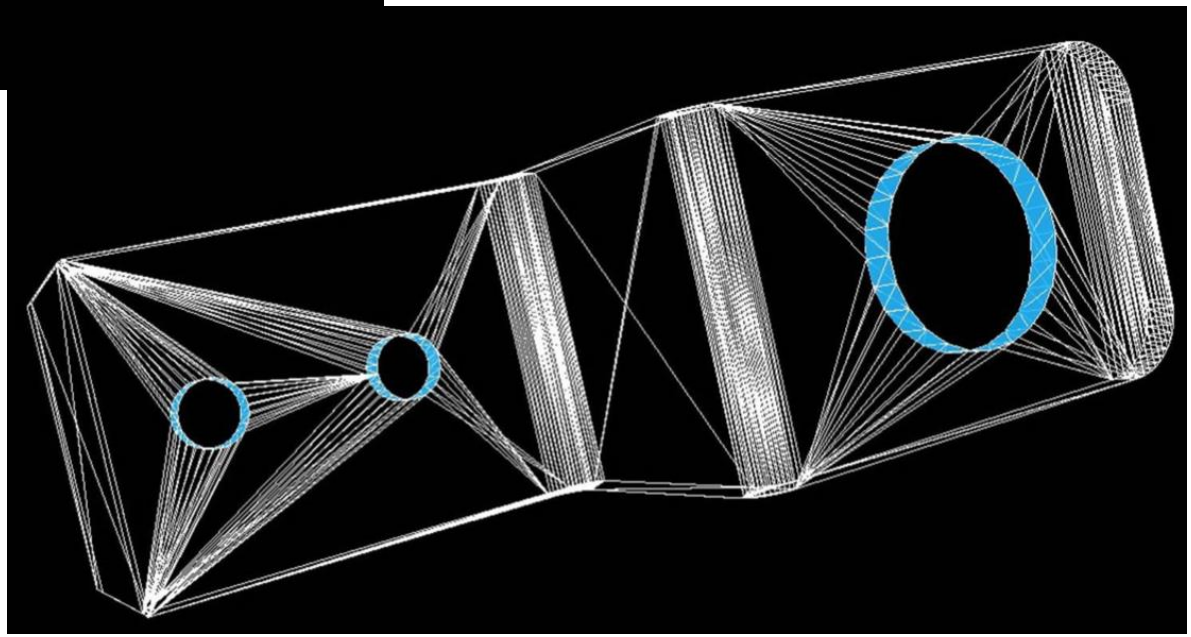
- Punkte im Dreieck als Funktion $F(u,v) \in A_\Delta$
  $F(u,v) = u \cdot P_1 + v \cdot P_2 + (1-u-v) \cdot P_3$ mit $0 \leq u, v, u+v \leq 1$



Erzeugung von Dreiecksflächen: Beispiel

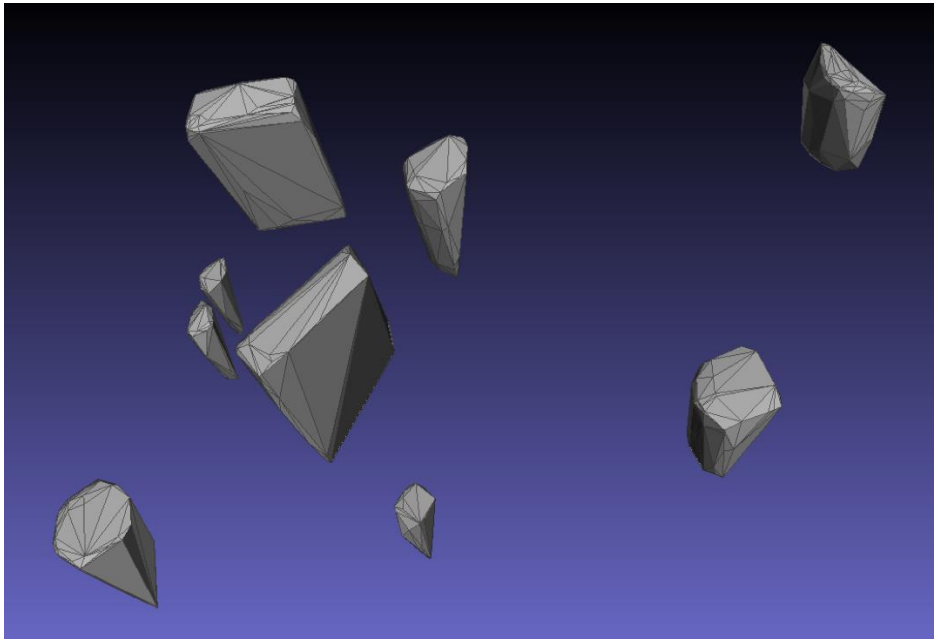


Mesh aus Laserdaten (Punktwolke) rekonstruiert



Mesh aus CAD-Modell (analytisches Modell) generiert

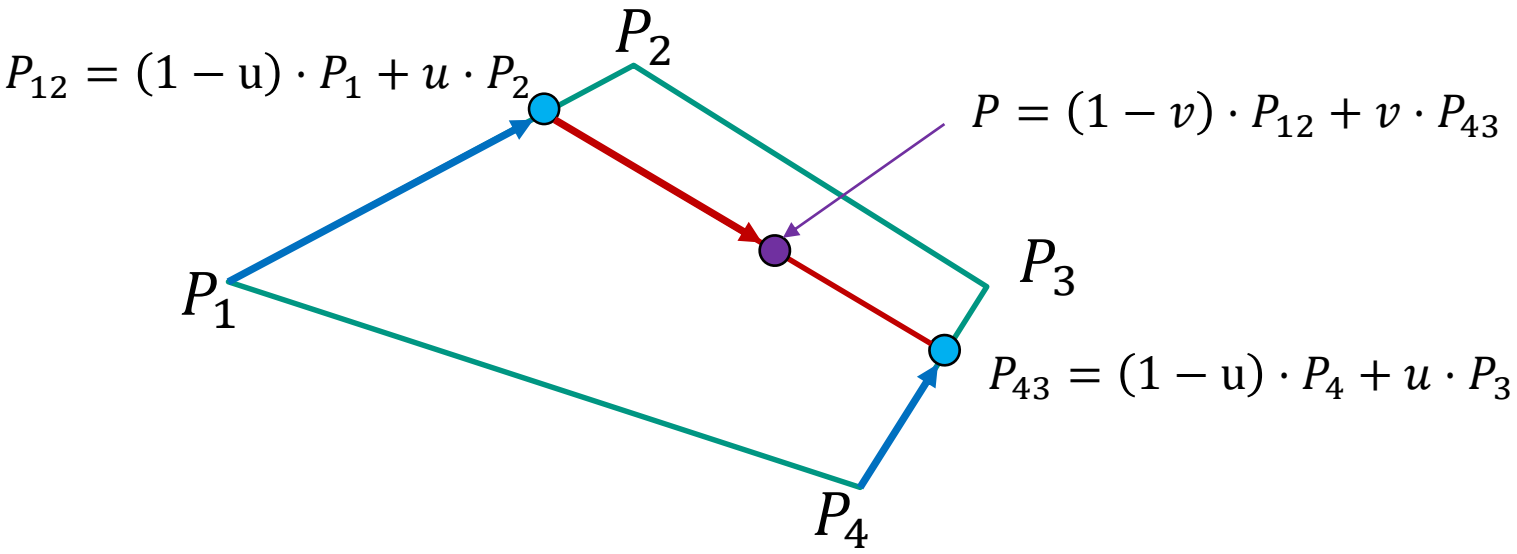
Erzeugung von Dreiecksflächen: Beispiel



Rekonstruktion aus Punktwolken

Viereckflächen

- Approximation von Freiformflächen im Raum durch **Vierecke**
- Gegeben sind 4 Punkte im Raum P_1, P_2, P_3, P_4.
- Definiere Fläche über **zweimalige lineare Interpolation** (bilinear)

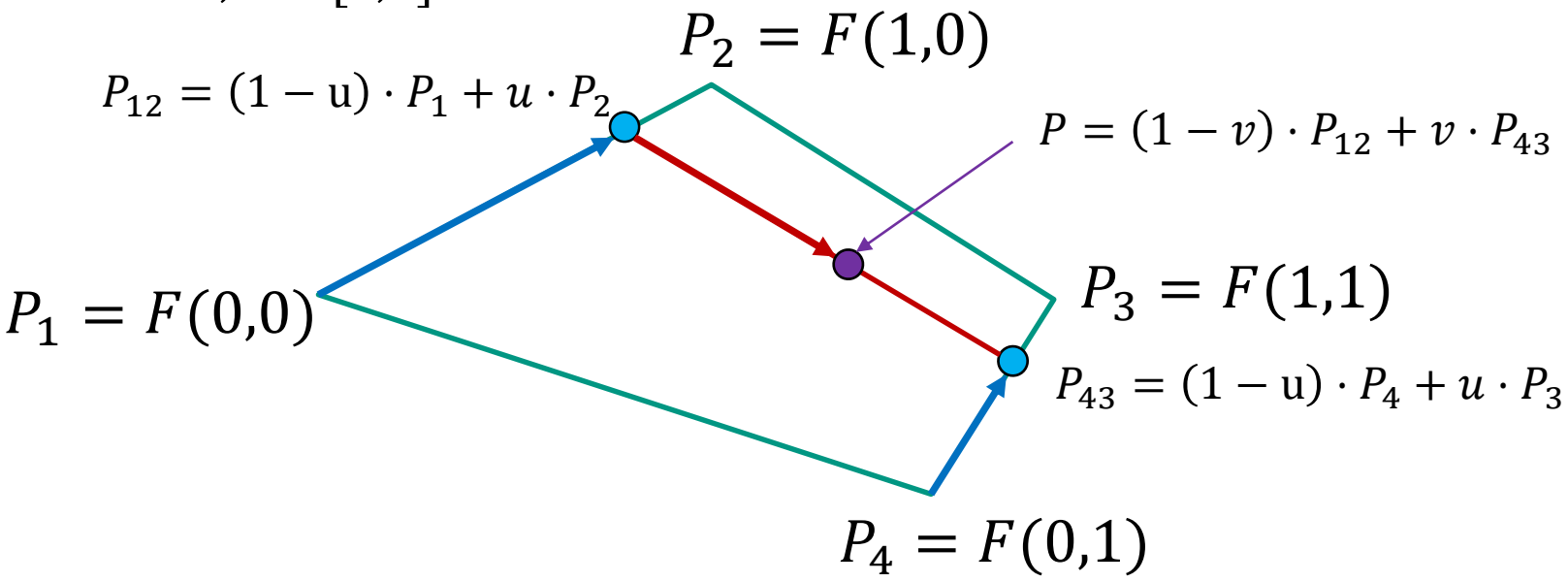


Viereckflächen: Bilineare Interpolation

- Definiere Fläche über **zweimalige lineare Interpolation** (bilinear)

- $F(u,v) = (1-v) \cdot P_{12} + v \cdot P_{43}$
  $= (1-v) \cdot \big((1-\mathrm{u}) \cdot P_1 + u \cdot P_2\big) + v \cdot \big((1-\mathrm{u}) \cdot P_4 + u \cdot P_3\big)$

  mit $u, v \in [0,1]$



Viereckflächen: Eigenschaften

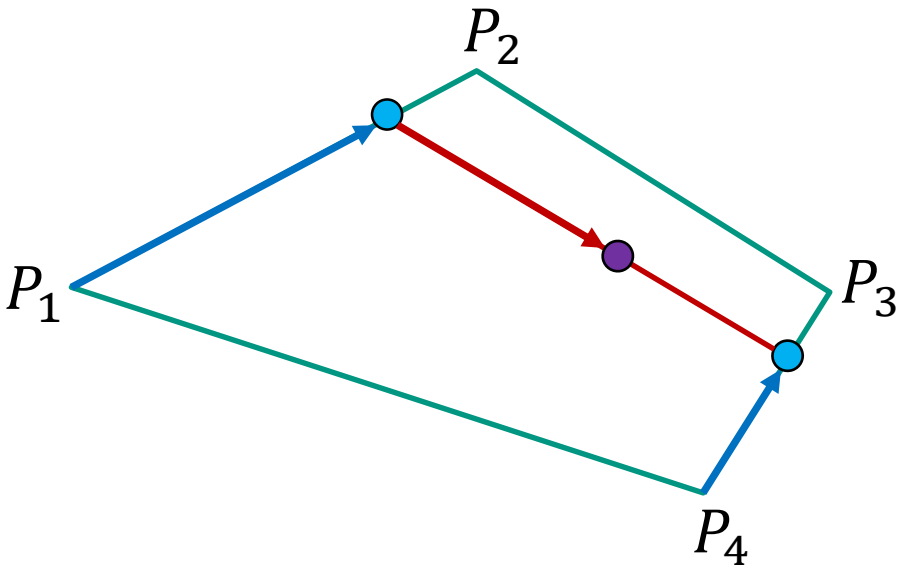
- **Vorteil**:

  + Flächenelemente können gekrümmt sein (3D)

  ⇒ weniger Gitterpunkte bei gleich guter Approximation

- **Nachteil**:

  - Rechnen mit gekrümmten Flächen ist aufwendig



Inhalt

- Motivation: Adaptive Roboteraufgaben
- **Objektmodelle**
  - **Geometrische Beschreibung**
    - **Kantenmodell**
    - **Flächenmodell**
    - **Volumenmodell**
  - Zusätzliche Eigenschaften
- Szenenmodelle

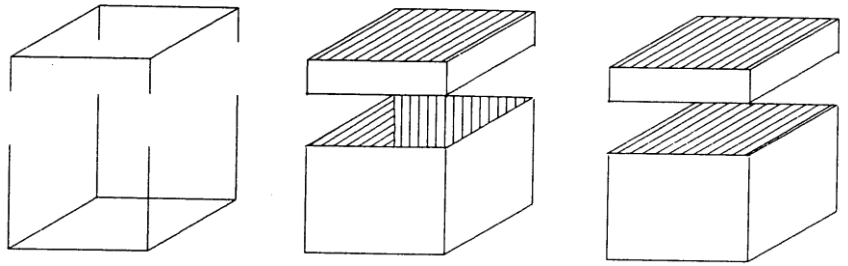
Kantenmodell

Reines Kantenmodell ohne weitere Informationen

- Nur die Kanten werden gespeichert, d.h. Punkte und
- Verbindungen (Gerade, Polygonzug, Bezierkurve, ...)
- **Vorteile**:

  + einfache Daten

  + wenige Daten



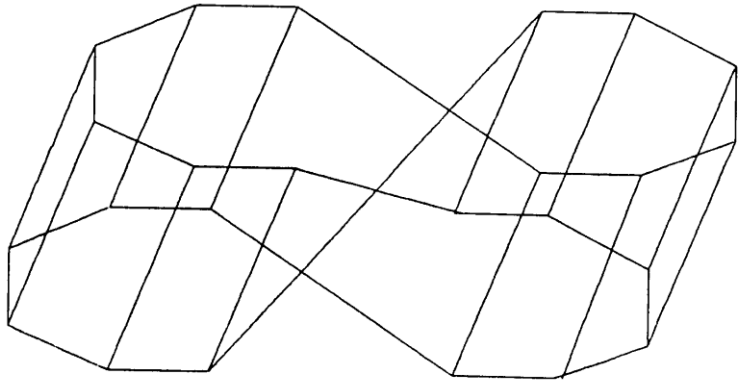
- **Nachteile**:

  - Mehrdeutigkeiten

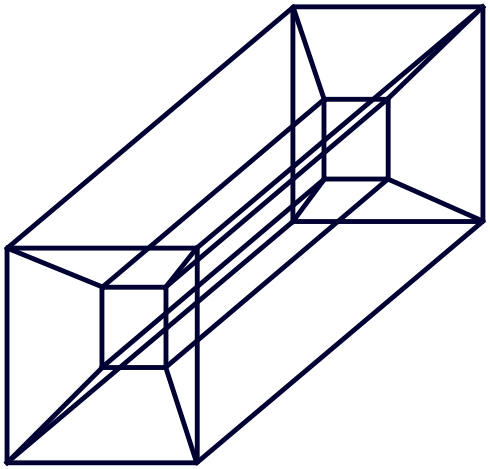
  - hoher Eingabeaufwand

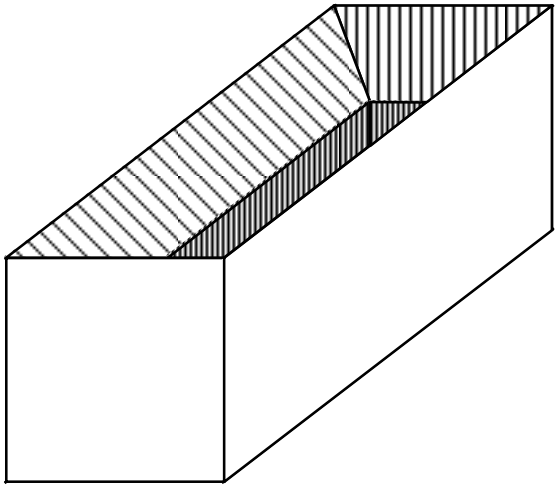
  - keine Kollisionsberechnung

  - kein Schnitt

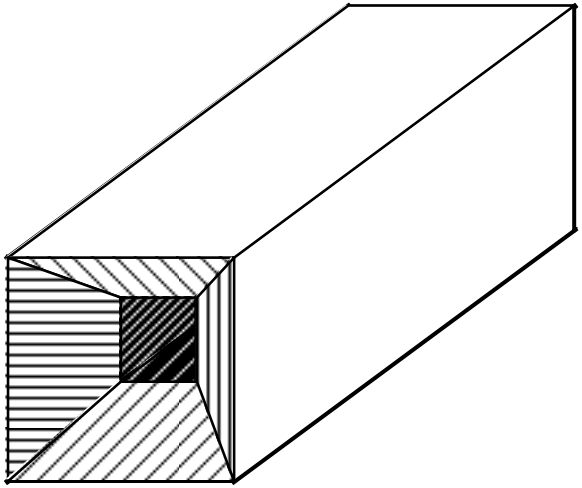


Kantenmodell

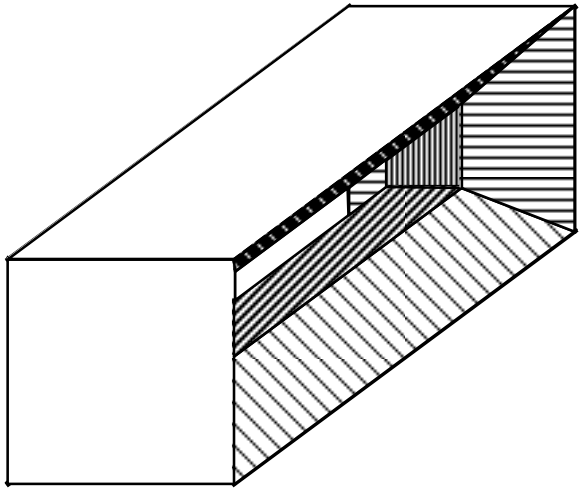




oben - unten



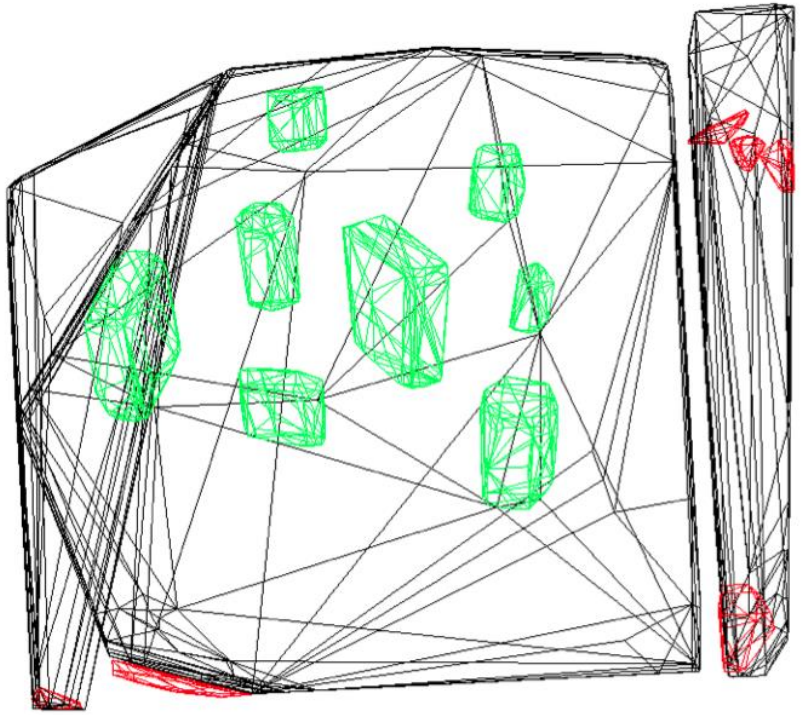
vorne - hinten



links - rechts

Kantenmodell: Beispiel

Kamerabild

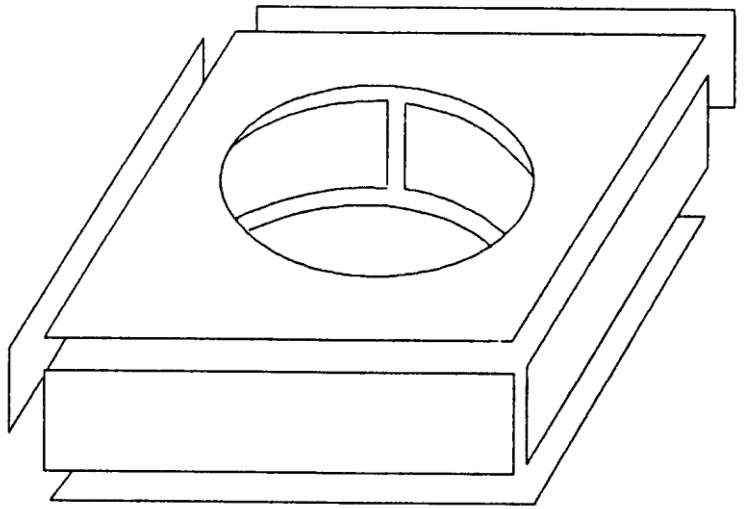


Rekonstruiertes Kantemodell

Flächenmodell

Speicherung von Kanten & Oberflächen

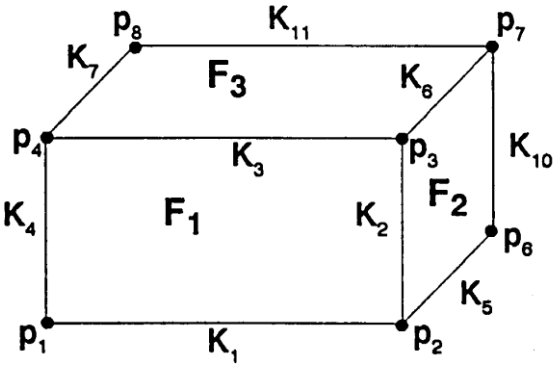
- **Vorteile**:
  - + effiziente Verfahren
  - + entspricht dem Vorgehen während der Modellierung
  - + schnelle Kollisions- und Abstandsberechnung
- **Nachteile**:
  - - hoher Eingabeaufwand
  - - Darstellung aufwendig
  - - Problem bei Schnittoperationen
  - - Inkonsistenzen möglich

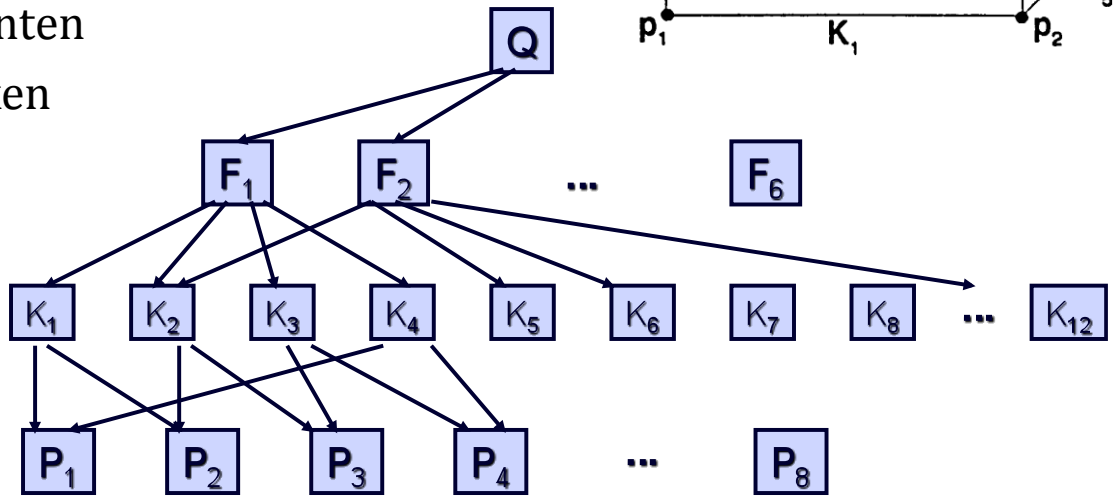


Boundary Representation

Organisationsform der geometrischen Flächenmodelle

- Hierarchische Darstellung eines Objektes durch begrenzende Elemente, i.d.R. Kanten oder Flächen.
- Elemente eines Quaders im Flächenmodell
- Elemente:
  - Q: Quader
  - F_i: $i \in \{1, \ldots, 6\}$: Flächen
  - K_i: $i \in \{1, \ldots, 12\}$: Kanten
  - P_i: $i \in \{1, \ldots, 8\}$: Ecken





Boundary Representation: Vorteile

Aus topologischer Struktur Informationen über

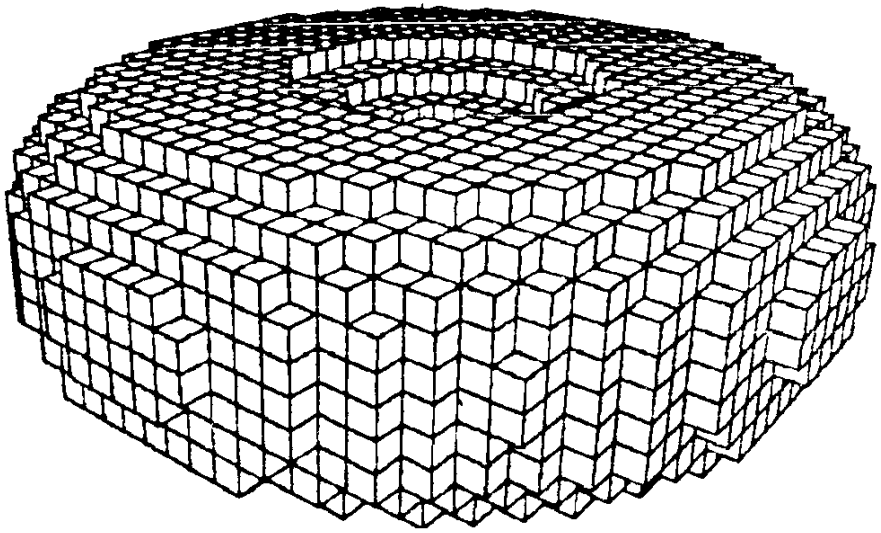
- Welche Flächen gehören zum Objekt?
- Welche Kanten gehören zur Fläche?
- Zu welchem Objekt gehört eine Fläche?
- Zu welchem Objekt gehört eine Kante?
- Welche Flächen stoßen aneinander?

  → *kantenbasierte Objekterkennung*

Volumenmodell: Approximative Zellenbelegung

- Objekte werden aus disjunkten Elementarzellen aufgebaut. Verwendung finden einfache geometrische Objekte z,B. Tetraeder, Quader, ...

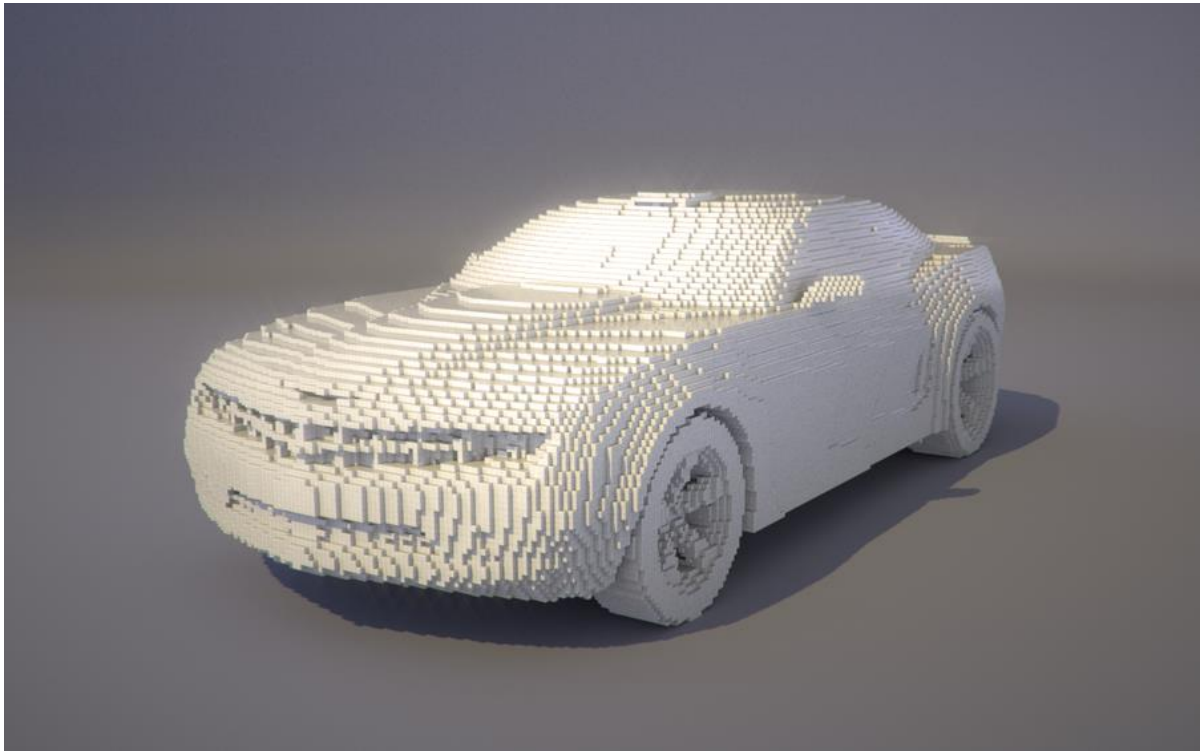
- Benutzt in der Strukturanalyse mit Finite-Elemente-Methoden (FEM).



Volumenmodell: Voxeldarstellung

Äquidistante Raumunterteilung in 3D

- Speicherungsmöglichkeiten
  - 0: nicht-belegt / 1: belegt / 2: weiß-nicht (unbekannt)
  - Wahrscheinlichkeit belegt 0.0 bis 1.0



Volumenmodell: Voxeldarstellung

Äquidistante Raumunterteilung in 3D

- **Vorteile**:
  - \+ Einfache Darstellung
  - \+ Berechnungen homogen, parallel auf GPU
  - \+ Raycasting u.ä. einfach parallelisierbar
- **Nachteile**:
  - Festes Gitter, gleicher Detaillierungsgrad sowohl bei großflächigen als auch bei feiner aufgelösten Strukturen
  - Auflösung/Präzision durch Speicher begrenzt

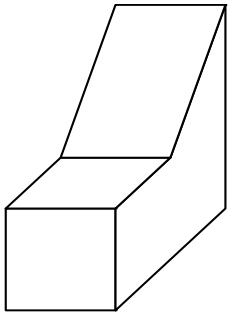
Octree: Effizientere Topologie als Voxel

- Der Raum wird in mehrere Zellen unterteilt
  (i.d.R. 8 Zellen: „oct-tree“).
- Zelle komplett vom Objekt belegt → als „belegt“ markieren
- Wenn die Zelle nur teilweise belegt ist, dann wird auf diese Zelle das Verfahren **rekursiv** angewendet. Ansonsten ist die Zelle leer
- Die Rekursion terminiert bei einer vorbestimmten minimalen Zellgröße
- Teilbelegte kleinste Zellen werden als belegt markiert.

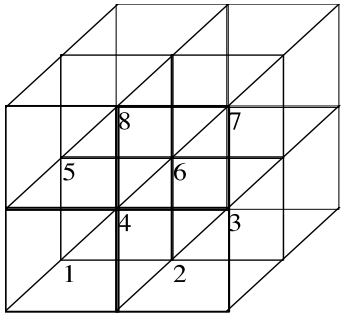
- 2D-Variante bereits behandelt: **Quadtree**

Octree: Beispiel

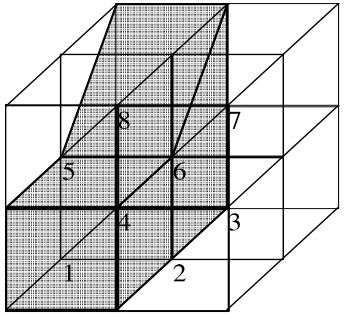
Körper



Zerlegung



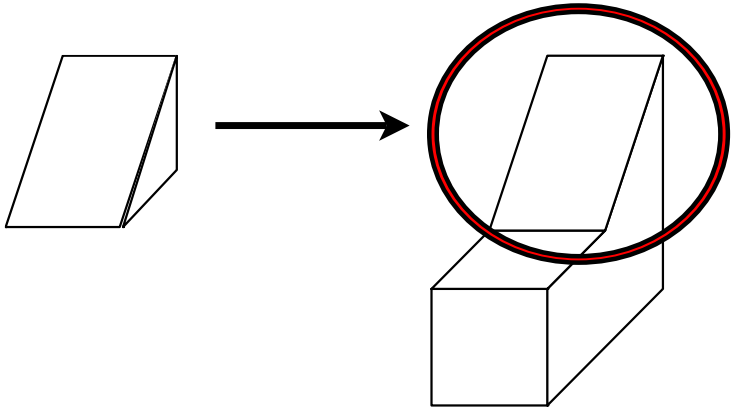
1. Zerlegung



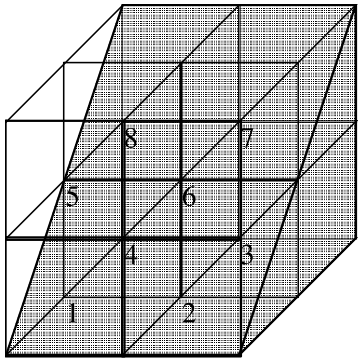
Zelle 1	belegt
Zelle 2	frei
Zelle 3	frei
Zelle 4	belegt
Zelle 5	frei
Zelle 6	frei
Zelle 7	frei
Zelle 8	teilbelegt

Octree: Beispiel

Restkörper

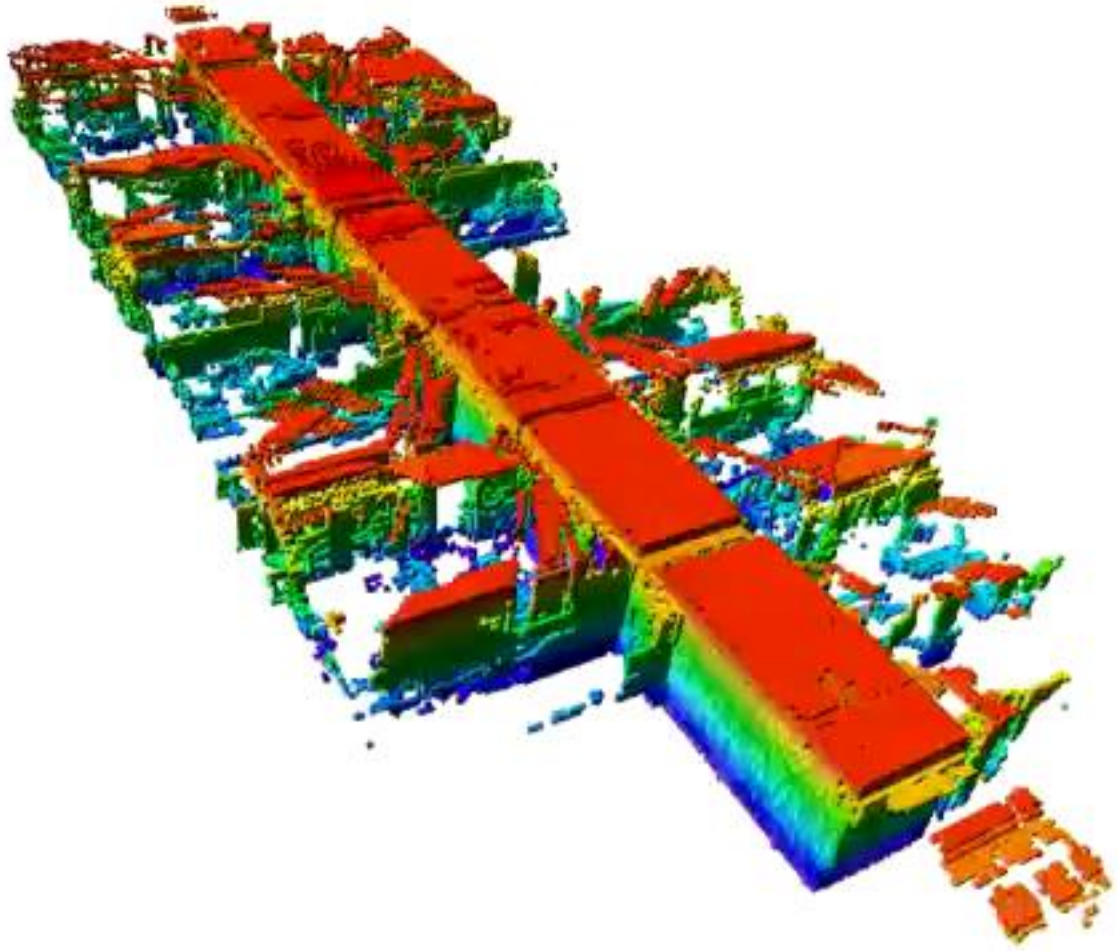


2. Zerlegung



Zelle 1	teilbelegt
Zelle 2	teilbelegt
Zelle 3	belegt
Zelle 4	belegt
Zelle 5	frei
Zelle 6	frei
Zelle 7	teilbelegt
Zelle 8	teilbelegt

Octree: Octomap



http://octomap.github.io

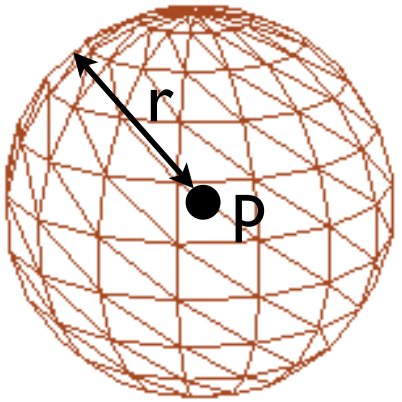
Ottree: Octomap

Octomap (BSD Lizenz) für Manipulation:

http://octomap.github.io

Analytisch-parametrische Modelle

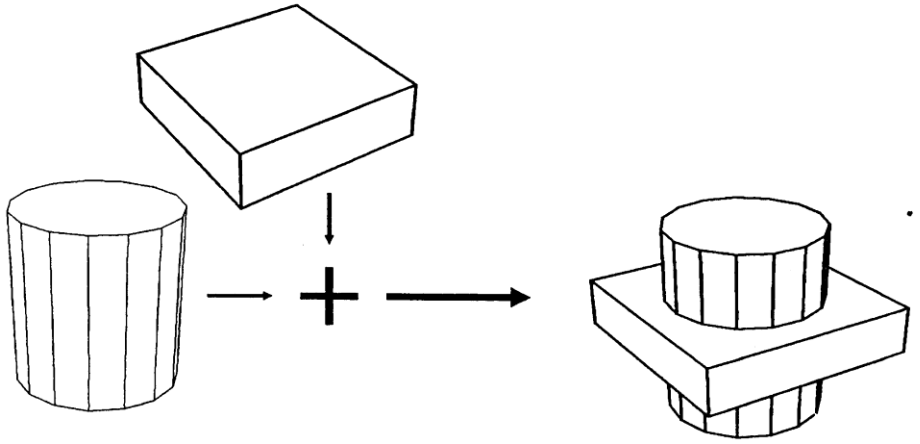
Beispiel analytisch gegebene Fläche



- Beispiel: 3D-Kugel
- $r = ||x - p||$
- Exaktes Modellierungsverfahren
- **Vorteile**:
  - + Geschlossene Darstellung (wenig Speicherbedarf)
  - + Analytische Darstellung erlaubt einfache Rechenverfahren (z.B. Schnitt von Ebenen / Kugeln→ schnelle Kollisionsberechnung)
- **Nachteil**:
  - - Wenige Flächen sind analytisch darstellbar

Parametrische Volumenmodelle (1)

- Grundkörper und topologische Operationen auf diesen (Schnitt, Vereinigung, ...) werden abgespeichert



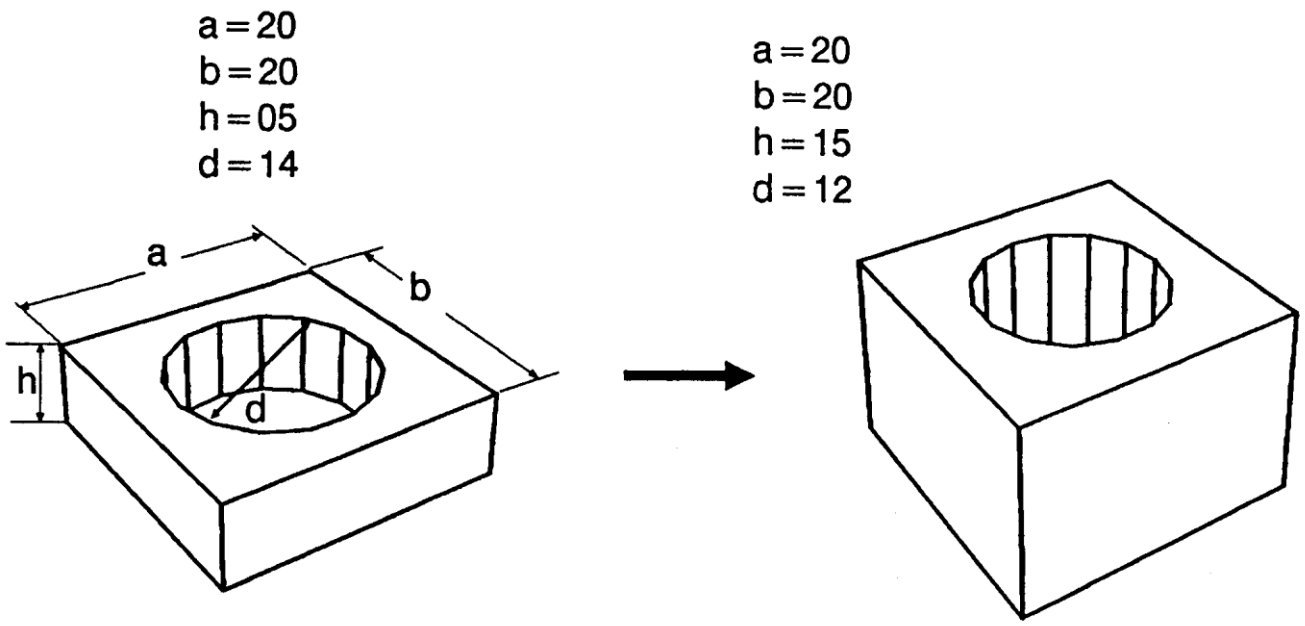
- **Vorteile**:
  - + Eindeutige Objektbeschreibung
  - + Geringer Eingabeaufwand
  - + Ergebnis von Operationen sind korrekte Objekte

- **Nachteile**:
  - - Hoher Implementierungsaufwand
  - - Einbindung von Freiformflächen schwierig

Parametrische Volumenmodelle (2)

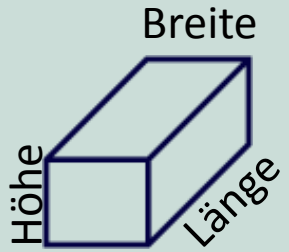
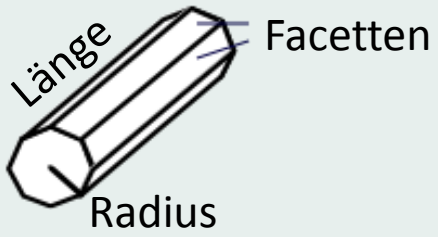
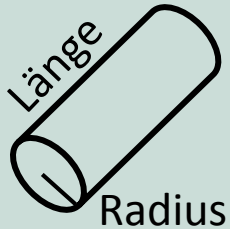
- Objekte sind bereits vorhanden und können durch Angabe von Parametern angepasst werden (Varianten).

- Konsistenzprüfungen sind notwendig ! ($d < a$)

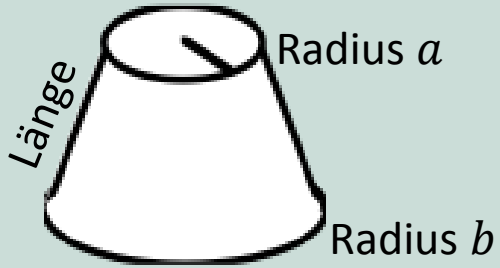
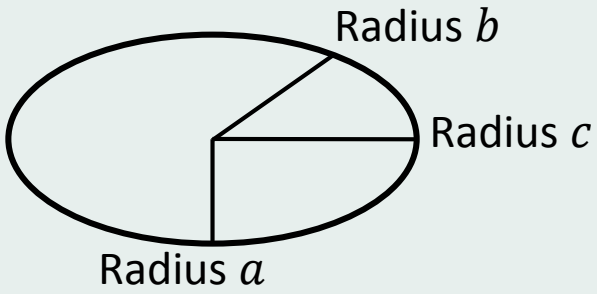
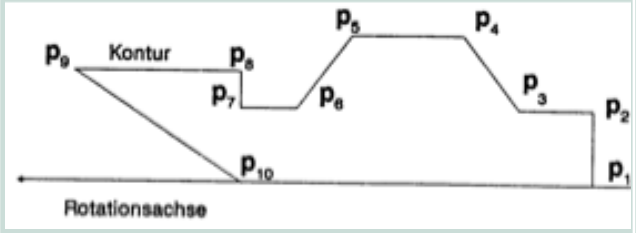
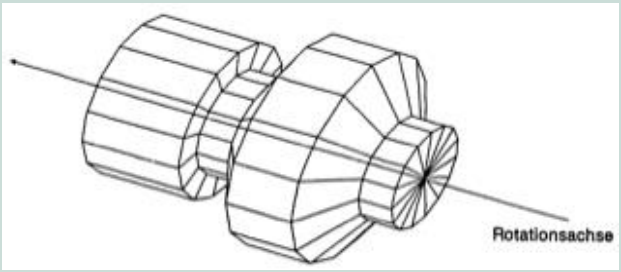


Constructive Solid Geometry (CSG)

- Es gibt eine Menge von einfachen Grundkörpern, die parametriert werden können und auf denen verschiedene Operationen definiert sind.

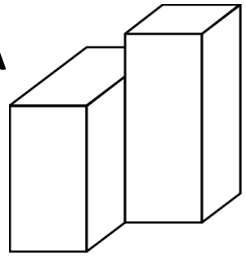
Grundkörper	Parameter	Skizze
Quader	Länge, Breite, Höhe	
Prisma	Länge, Radius, Facetten	
Zylinder	Länge, Radius	

Constructive Solid Geometry (CSG)

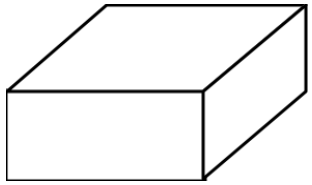
Grundkörper	Parameter	Skizze
Kegel	Länge, Radius a, Radius b	
Ellipsoid	Radius a, Radius b, Radius c	
Rotationskörper	Achse, Kontur	 

CSG: Operatoren

Objekt A

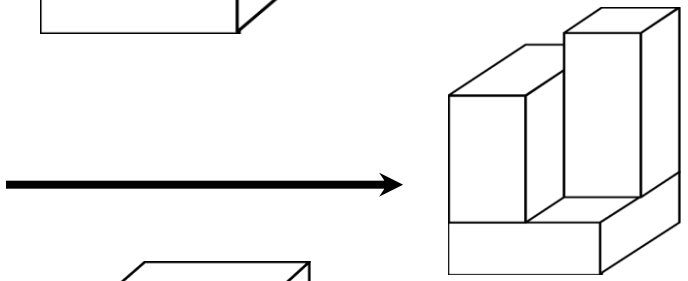


Objekt B

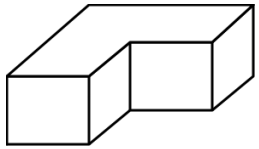


Operatoren

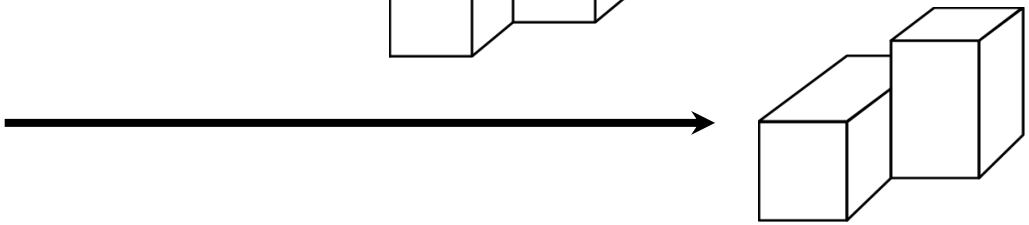
- *Vereinigung* (Summe) A ∪ B



- *Schnitt* A ∩ B



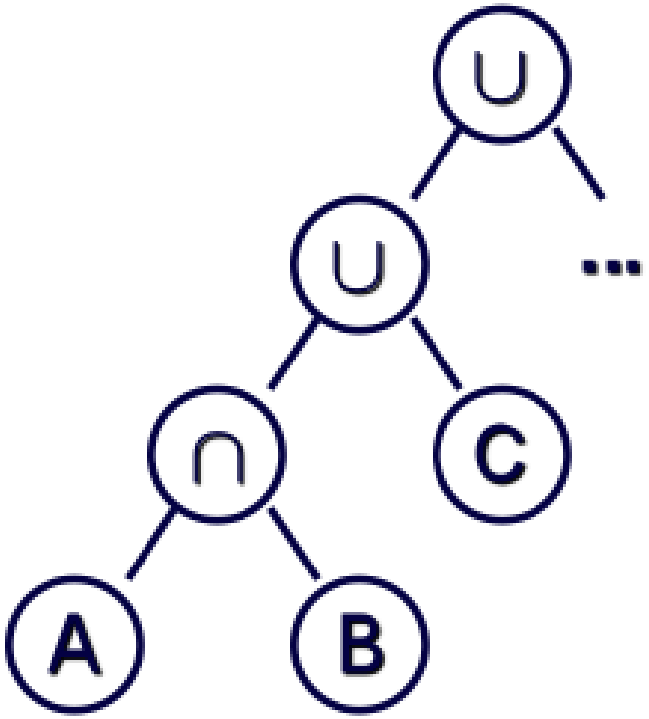
- *Differenz* A / B



- *Sweep*:
  Ein Grundelement (z.B. eine Fläche) wird entlang einer Raumkurve verschoben. Der durchdrungene Raum stellt das neue Objekt dar.

CSG: Operatoren

- Speicherung erfolgt als Binär-Baum mit
  - Knoten als Operation
  - linker bzw. rechter Teilbaum als Teil-CSG oder Grundkörper

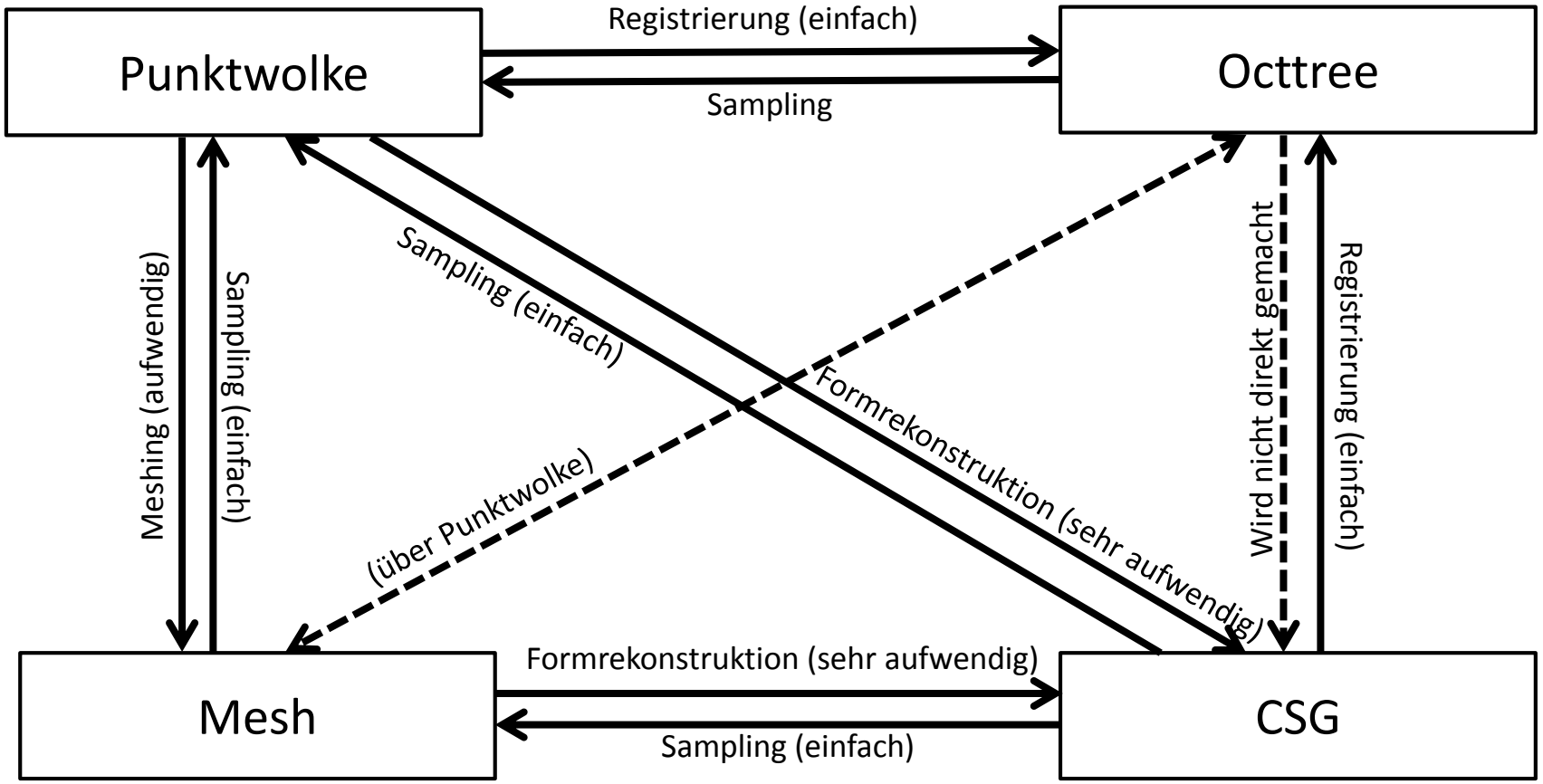


(A ∩ B) ∪ C

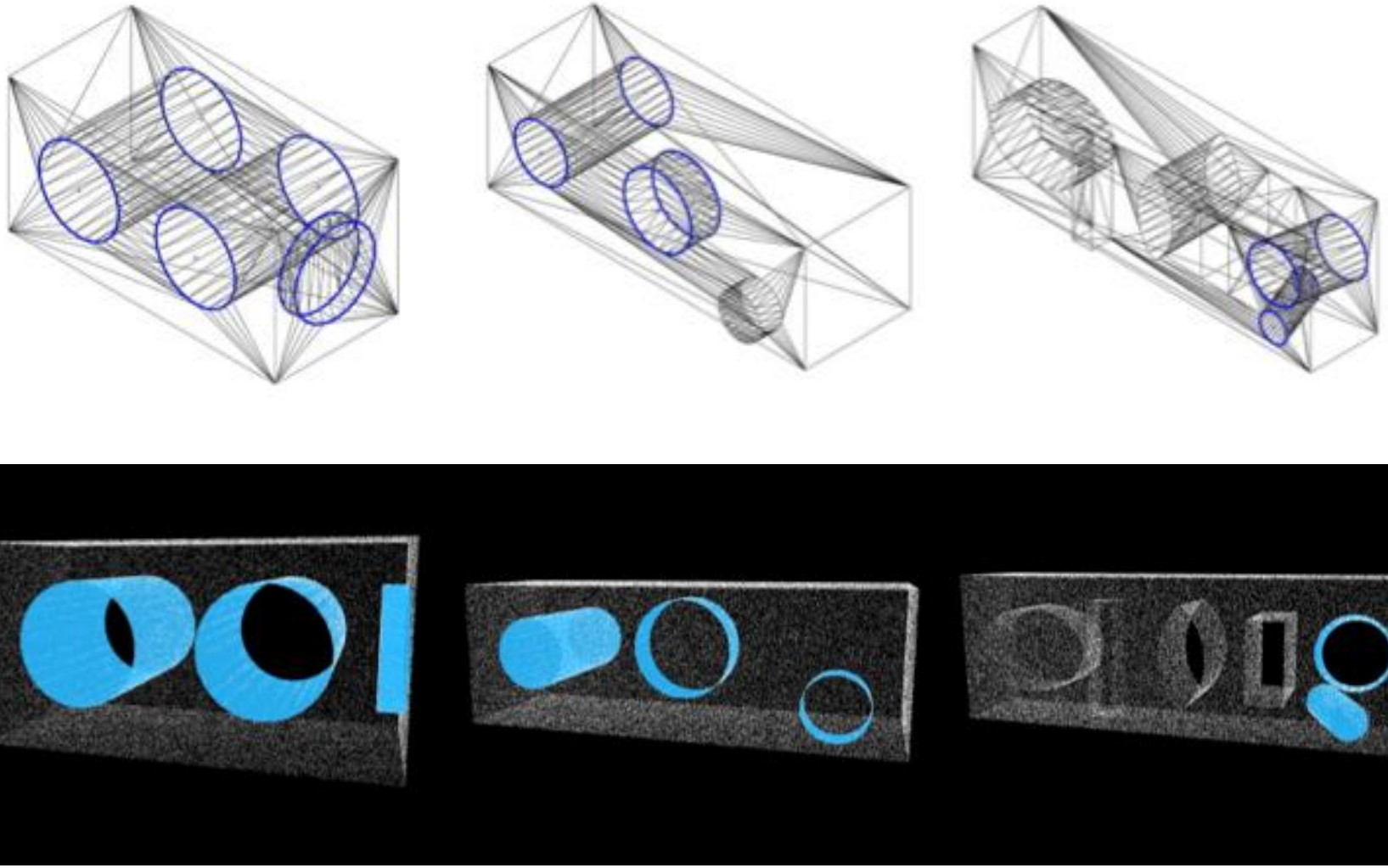
Anwendungsgebiete für geometrische Modelle

- Punktwolke:
  - Lokalisation, Klassifikation
  - Kartierung (mobile Systeme)
- Mesh:
  - Bewegungsplanung (v.a. Manipulatoren)
  - Dynamische Simulation (komplexe Starrkörper)
- Voxel/Octtree:
  - Bewegungsplanung (v.a. mobile Systeme)
  - Dynamische Simulation elastischer Materialien (FEM)
- CSG:
  - Computer-aided Design (CAD), Engineering (CAE), Manifacturing (CAM)
  - Dynamische Simulation (einfache Starrkörper)

Umwandlungen zwischen Modelltypen



Beispiel: Kantenmodell zu Punktwolke



Inhalt

- Motivation: Adaptive Roboteraufgaben
- **Objektmodelle**
  - Geometrische Beschreibung
  - **Zusätzliche Eigenschaften**
- Szenenmodelle

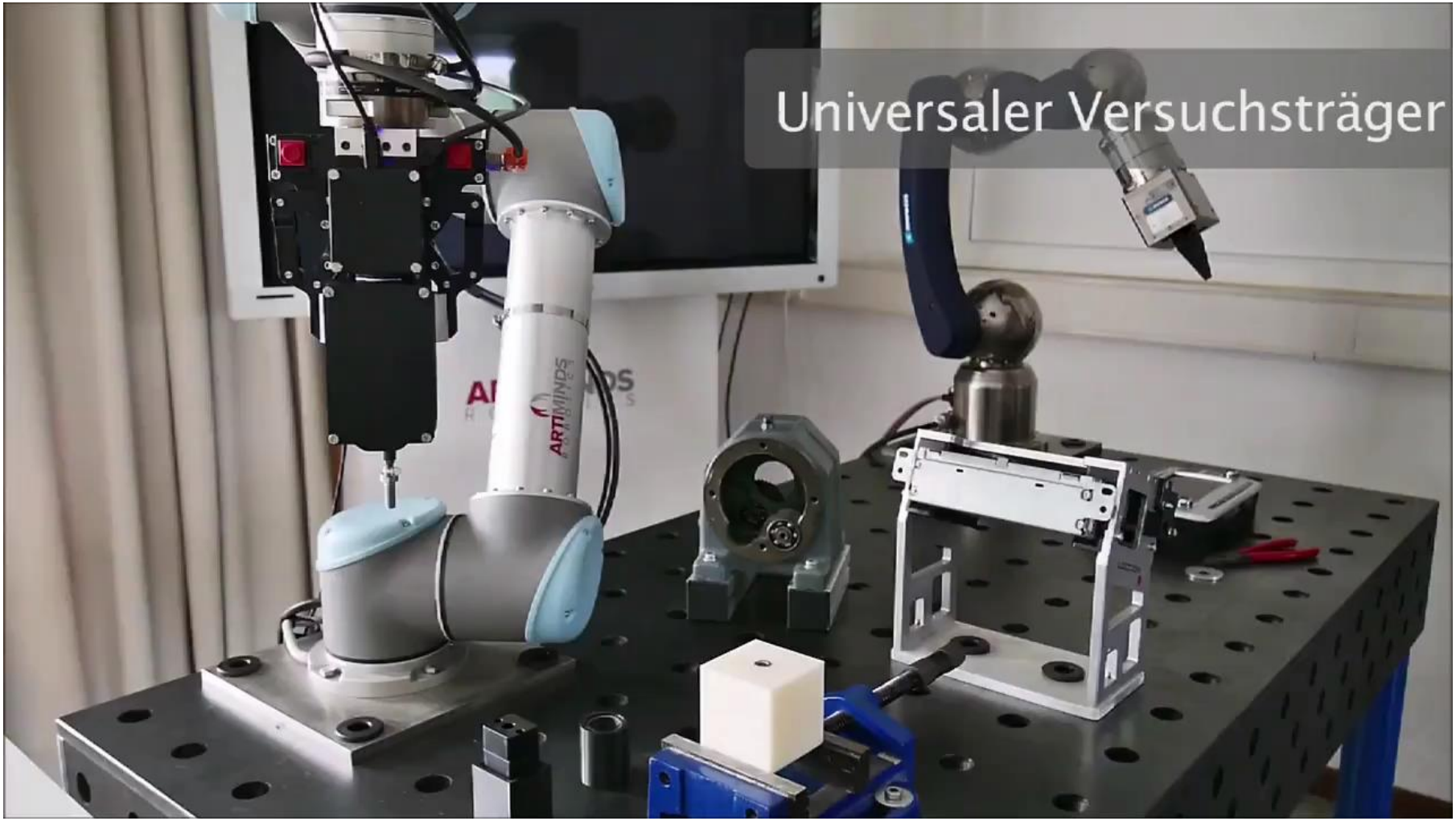
Eigenschaften des Objektes mit Geometriebezug

- Masse
- Oberflächeneigenschaften (z.B. Reibung)
- Temperatur
- Steifigkeit

- Greifpunkte
- Verbindungspunkte zur Montage
- Ablagepunkte bei Paletten
- Füllmenge bei Paletten
- Stellung bezüglich eines Referenzobjektes

Stark **aufgabenabhängig**!

Zusätzliche Eigenschaften: Beispiel



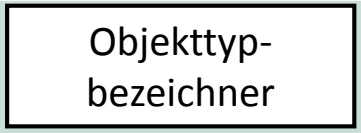
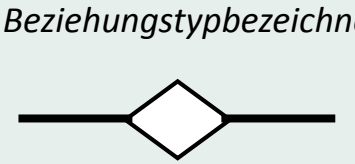
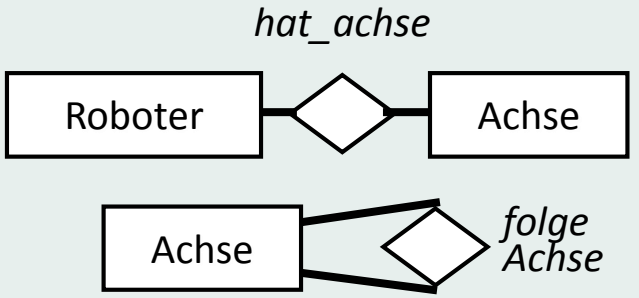
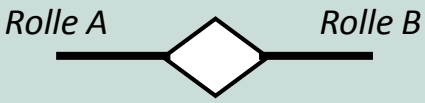
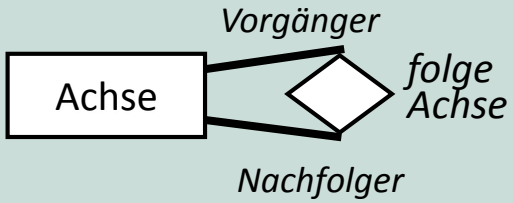
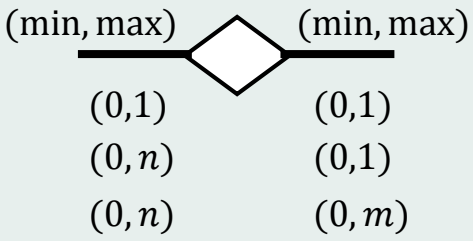
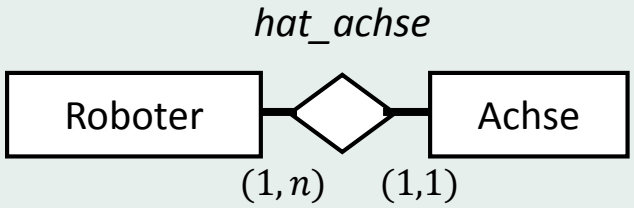
Inhalt

- Motivation: Adaptive Roboteraufgaben
- Objektmodelle
  - Geometrische Beschreibung
  - Zusätzliche Eigenschaften
- **Szenenmodelle**

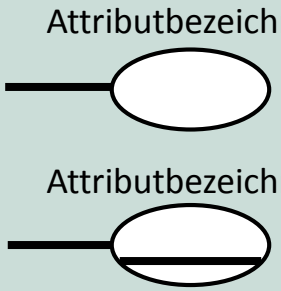
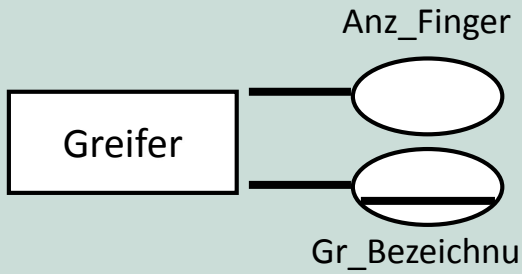
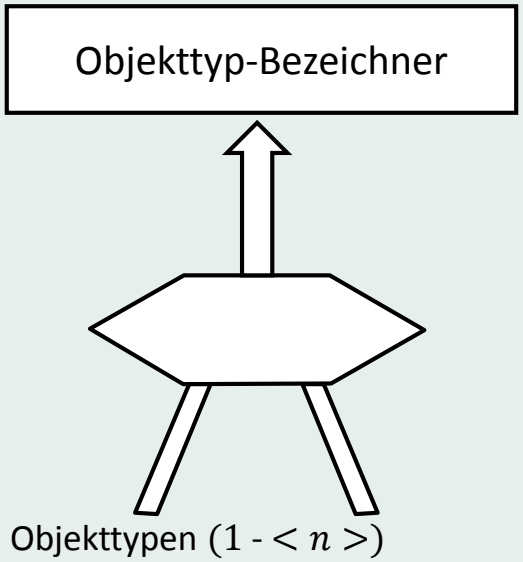
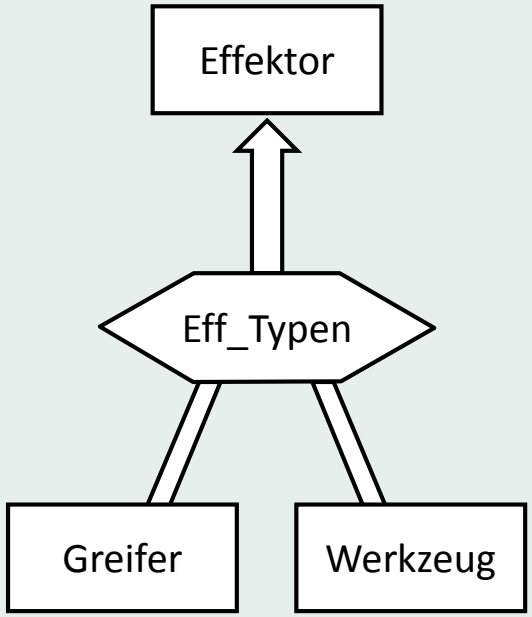
Übersicht Szenenmodelle

- Entity-Relationship-Modelle
- Semantische Netze
- Frame Modelle nach Minsky
- Implicit Shape Models (ISMs)
- Probablistische Object Constellation Models (OCMs)

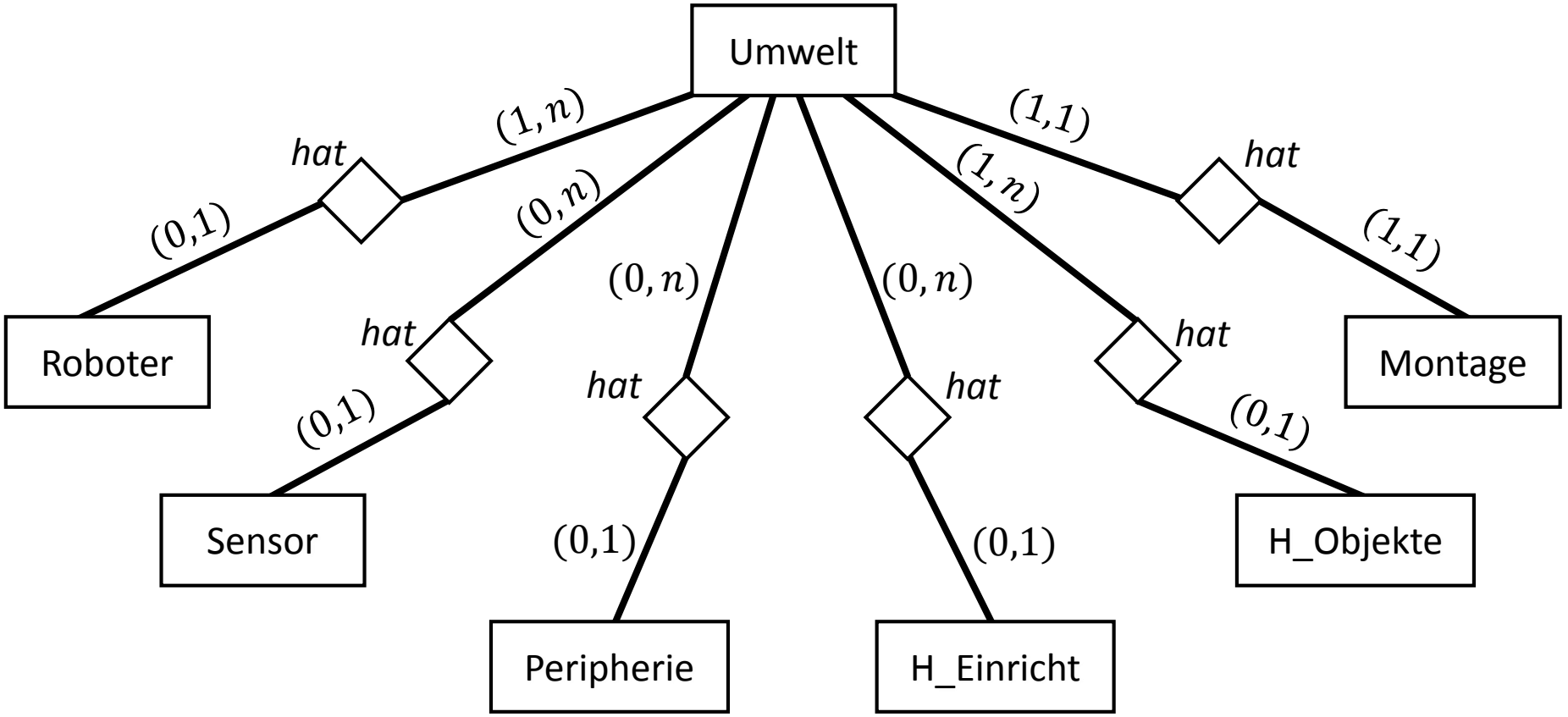
Entity-Relationship-Modelle (ER)

Modellierkonstrukt	Graphisches Symbol	Beispiel
Objekttyp (Entity set)		
Beziehungstyp (Relationship type) • zweiseitig • rekursiv		
Rollen		
Kardinalität		

Entity-Relationship-Modelle (ER)

Modellierkonstrukt	Graphisches Symbol	Beispiel
Attribute • beschreibende • identifizierende		
Generalisierungs-hierarchie		

Beispiel eines ER-Modells



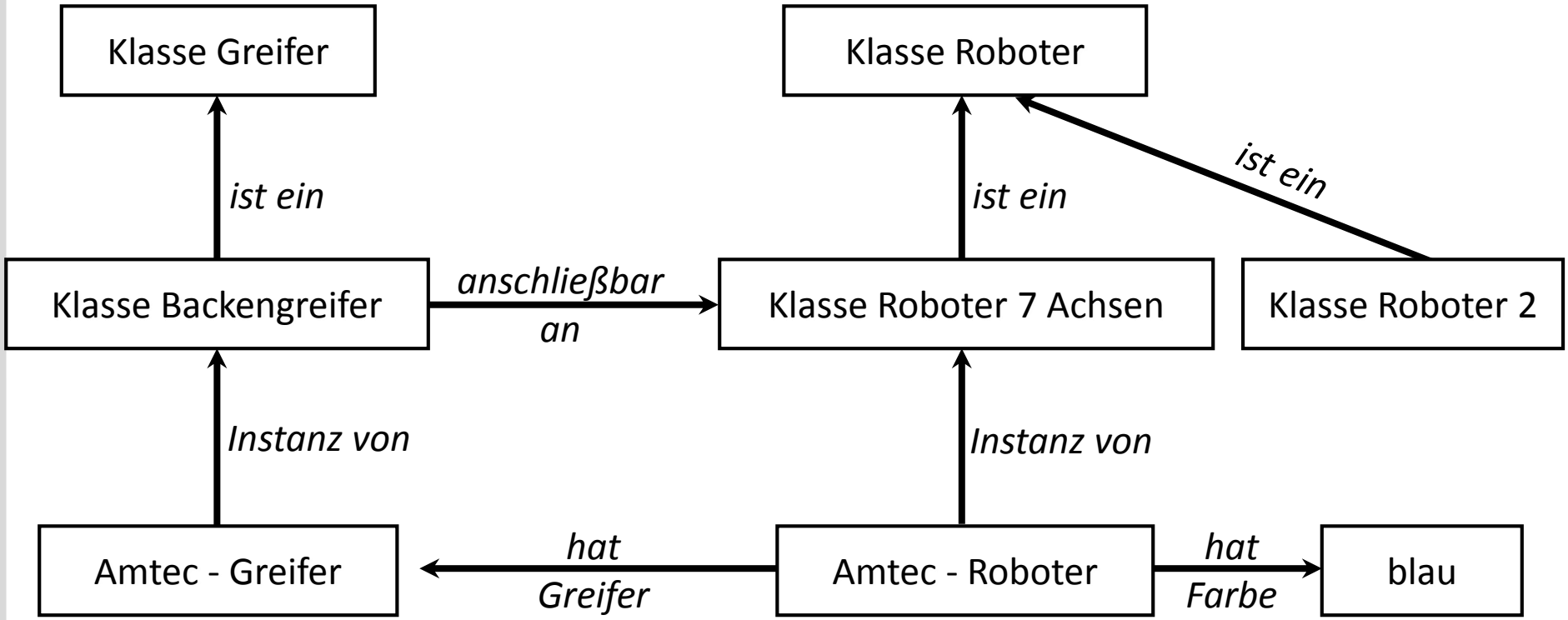
Semantische Netze

Konzept: Objekte und Beziehungen

- Semantisches Netz = gerichteter Graph
- Knoten = Objektklasse oder Einzelobjekt
- gerichtete und benannte Kanten = Beziehungen

- nur zweistellige Beziehungen
- mehrstellige Beziehungen => als eigenes Objekt
- keine Attribute => Attribute als „Wertobjekt“

Beispiel eines Semantischen Netzes

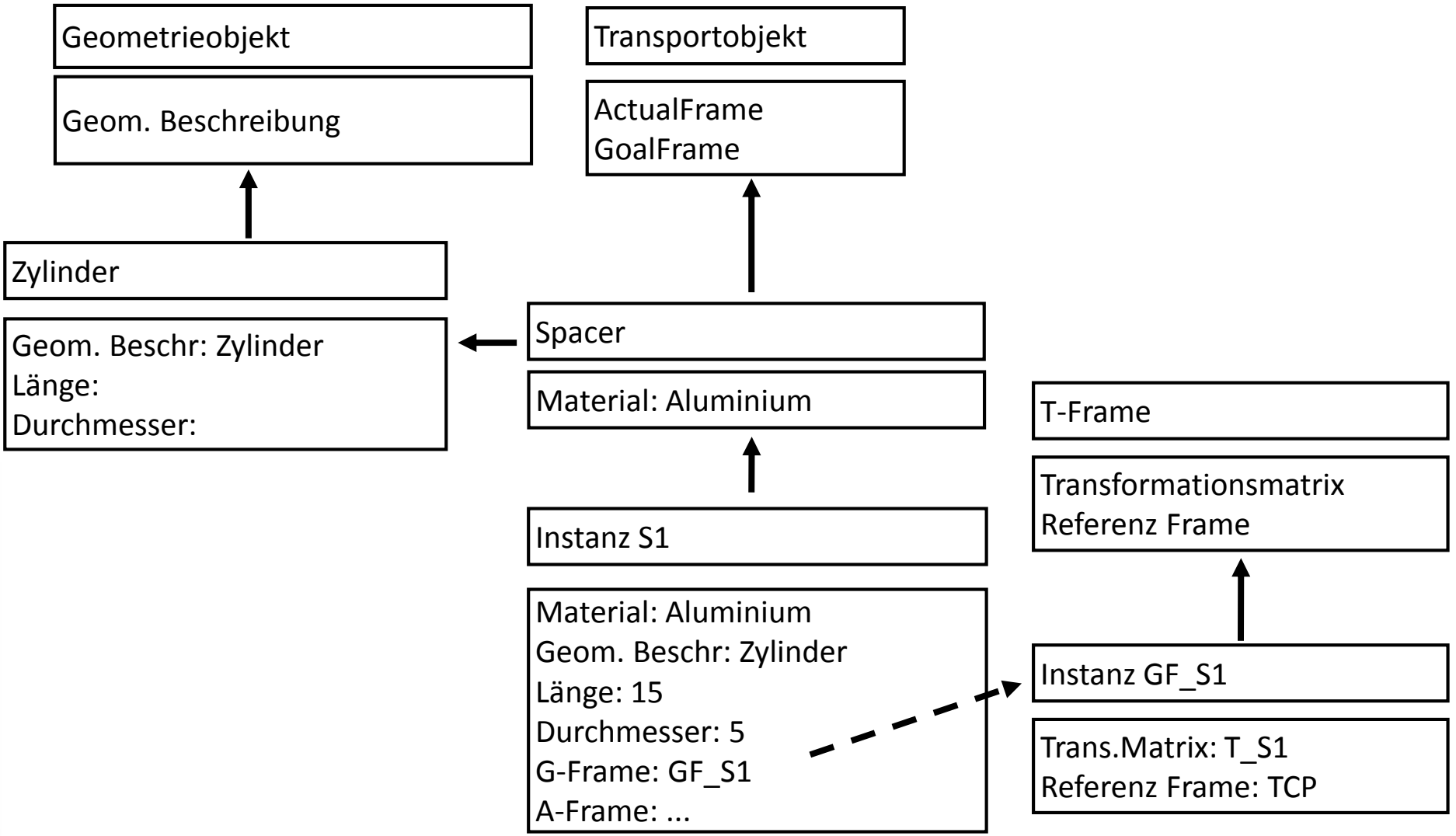


Frame-Modell nach Minsky

- Frame = Schablone (nicht mit Koordinaten-Frames verwechseln)
- Erfassung der:
  - Eigenschaften von Objekten
  - Einordnung in Hierarchie von Objektklassen
- Leichte Implementierung mit objektorientierten Sprachen
- Frame:
  - beschreibt ein Objekt / Objektklasse
  - enthält eine Menge von Slots
- Slots enthalten:
  - Attribute
  - Verweise auf andere Frames

➢ Instanzen, Vererbung

Beispiel eines Frame-Modells



Implicit Shape Modell (ISM)

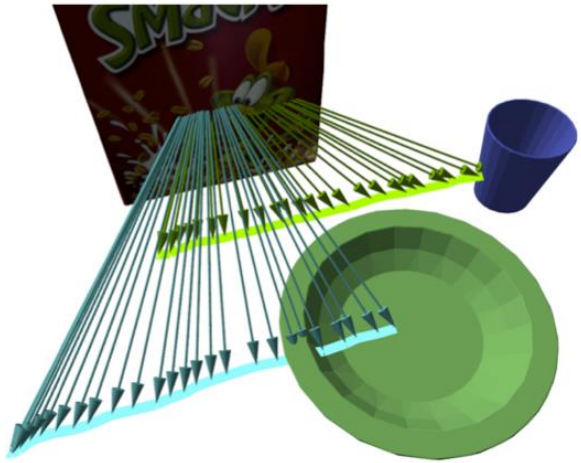
Moderne Szenenbeschreibung, robust bei Varianzen

- Szene besteht aus relativen Transformationsrelationen zwischen Objekten
- ISM ist Variation der Generalisierten Hough-Transformation (Bildverarbeitung)
- Daraus folgt, dass Instanzen der Relationen in *buckets* abstimmen (*voten*)
- Baumartige Topologie der relativen Relationen

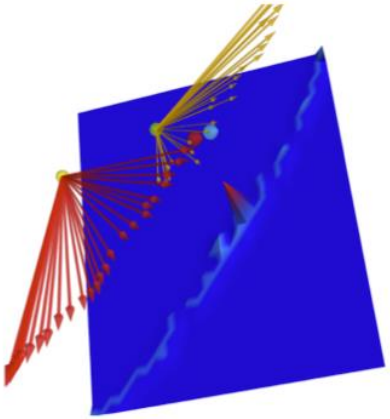
Implicit Shape Modell (ISM)

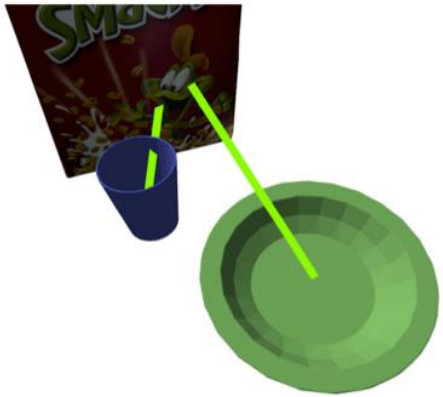
Hough-Voting findet im 3D-Raum statt

- Votes werden in einem Voxel-Gitter akkumuliert

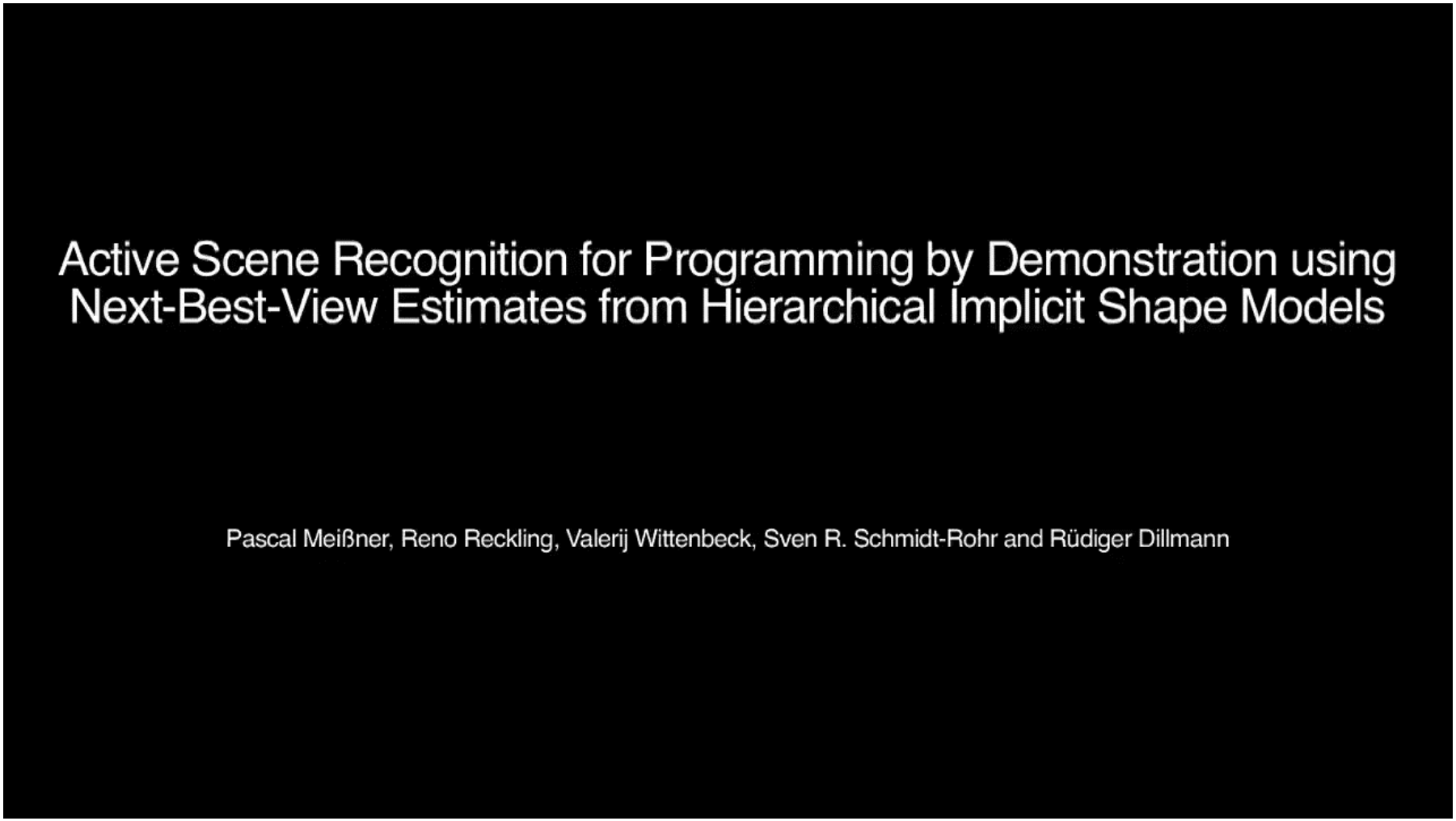


- Weitere Informationen: Meißner et al.





Implicit Shape Modell (ISM)



Object Constellation Modell (OCM)

Kurzvorstellung OCM: probabilistisches Modell

- Probabilistische Szenenrelationen
- Basiert auf Gauß-Mixturen (GMMs) und probabilistischer Graph-Inferenz

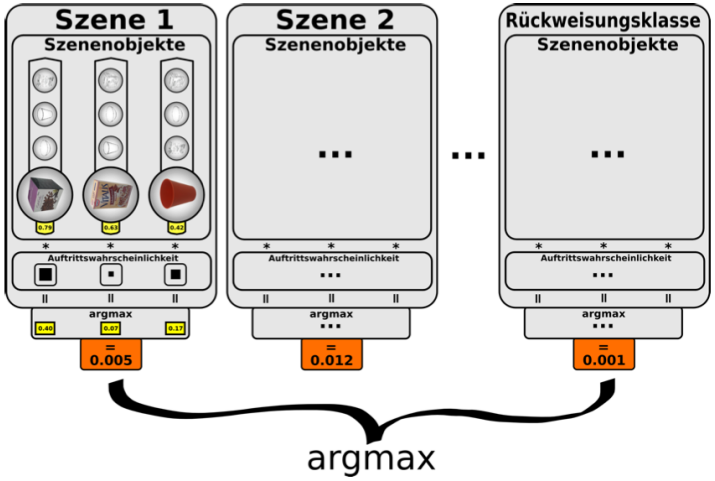
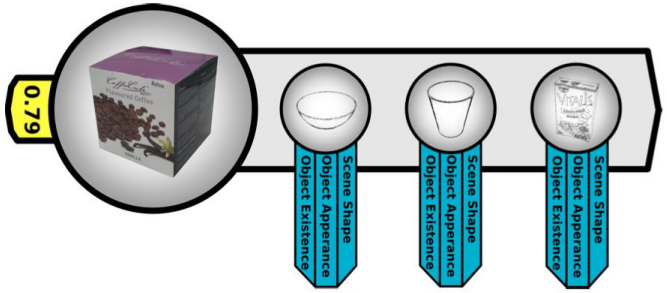


Abbildung 5.4: Eine grafische Darstellung des Szenenmodells. Es wird die Wahrscheinlichkeit einer Reihe von Szenen inklusive der Rückweisungsklasse berechnet. Jede Szene basiert auf einer Reihe von Szenenobjekten, wobei das mit der höchsten gewichteten Wahrscheinlichkeit die Wahrscheinlichkeit der Szene ergibt.



5.6: Das Object Constellation Model beschreibt ein einzelnes Szenenobjekt, also ein Objekt (hier die Kaffeebox) im Kontext einer Szene. Die Slots werden durch die kleinen Kreise rechts symbolisiert, jeder Slot beschreibt ein Objekt der Szene mit den in blau dargestellten Parametern. Der gelbe Kasten gibt an, mit welcher Wahrscheinlichkeit das Szenenobjekt vorliegt.

Weitere Informationen:
Gehrung, Meißner et al.