II
Aufgabenorientierte Programmierung

- Umweltmodellierung -
- Aufgabenmodellierung -

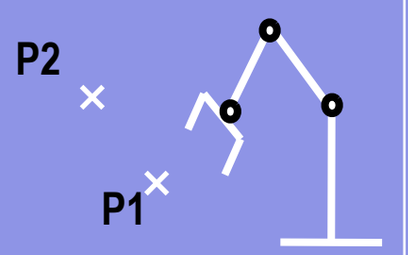
Prof. Dr. Rüdiger Dillmann
Robotik II

Roboterprogrammierverfahren

on-line

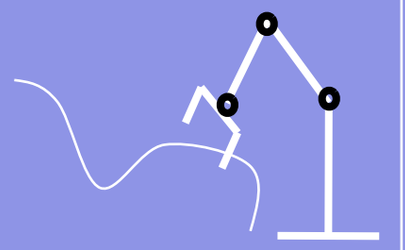
direkt

Teach-In



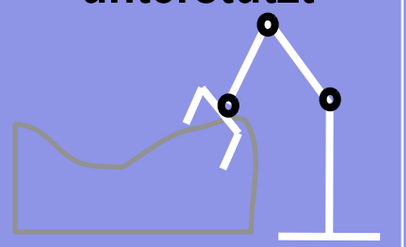
Punkte anfahren u. abspeichern

Play-Back



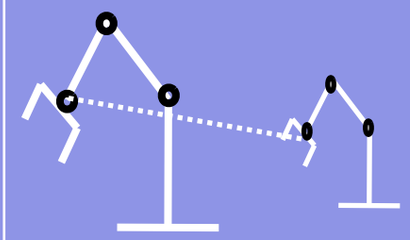
Bahn anfahren u. abspeichern

Sensor-unterstützt



Werkstückkontur erfassen

Master-Slave



Einsatz kinematischen Modells

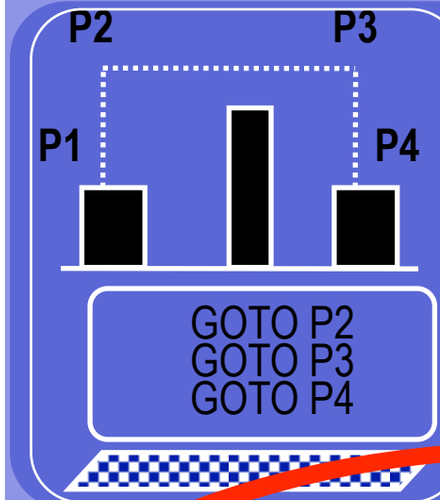
Hybride Verfahren

Interaktive Verfahren

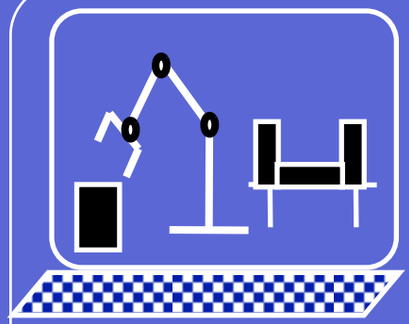
PdV

off-line

textuell

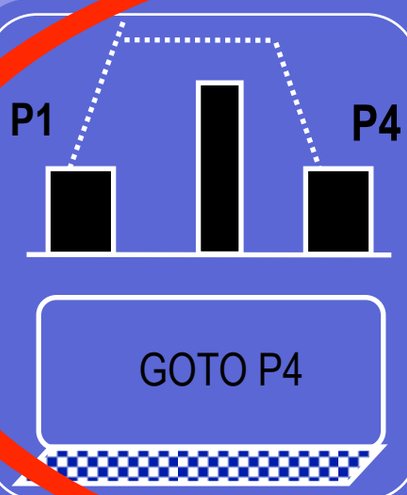


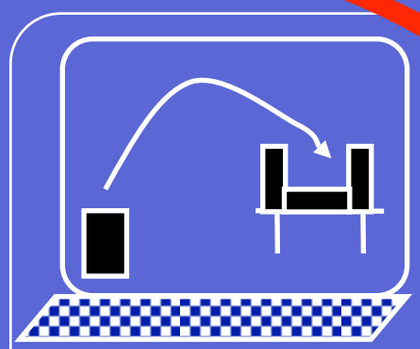
graphisch



Bewegungs-orientiert

explizit

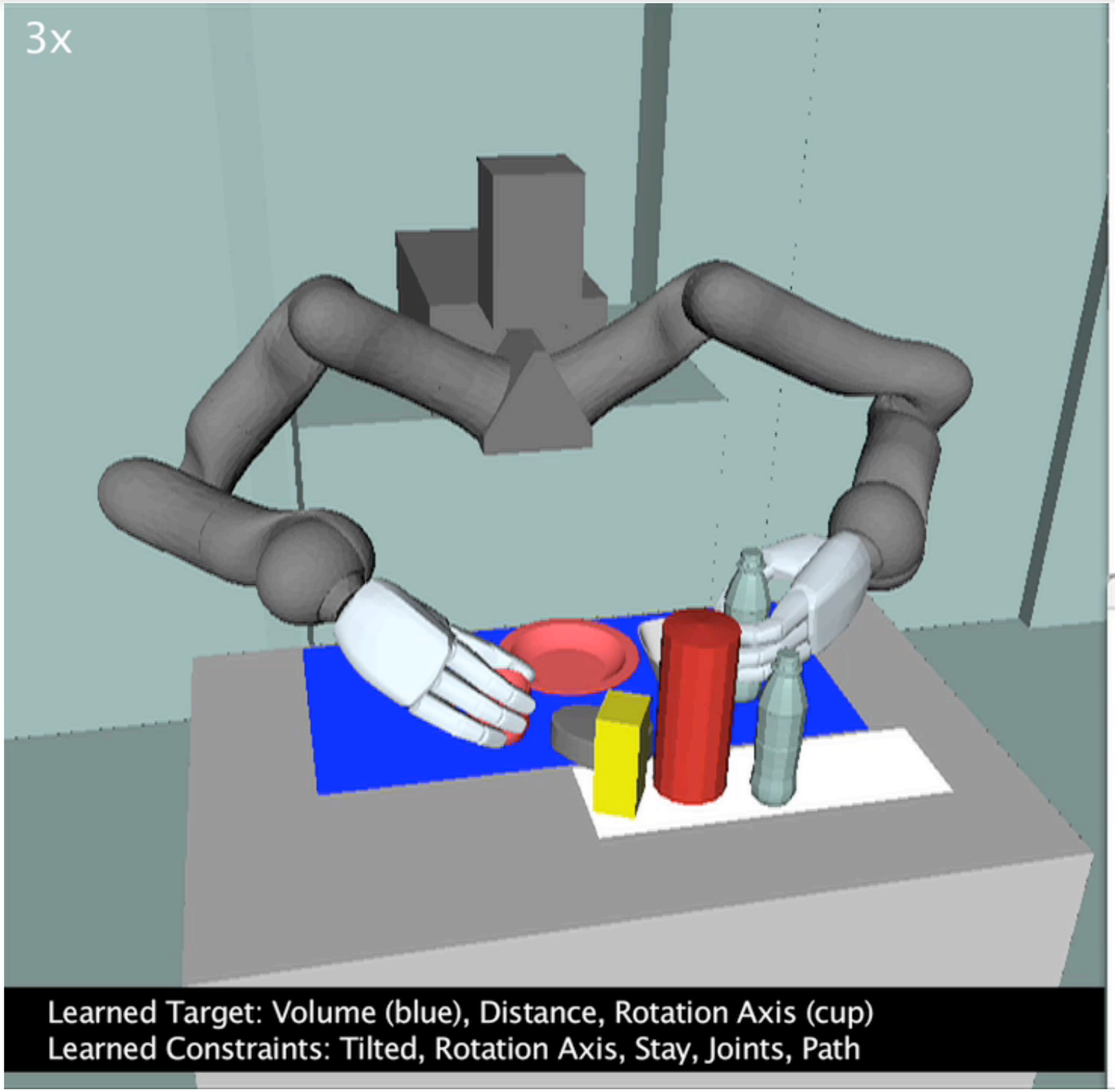


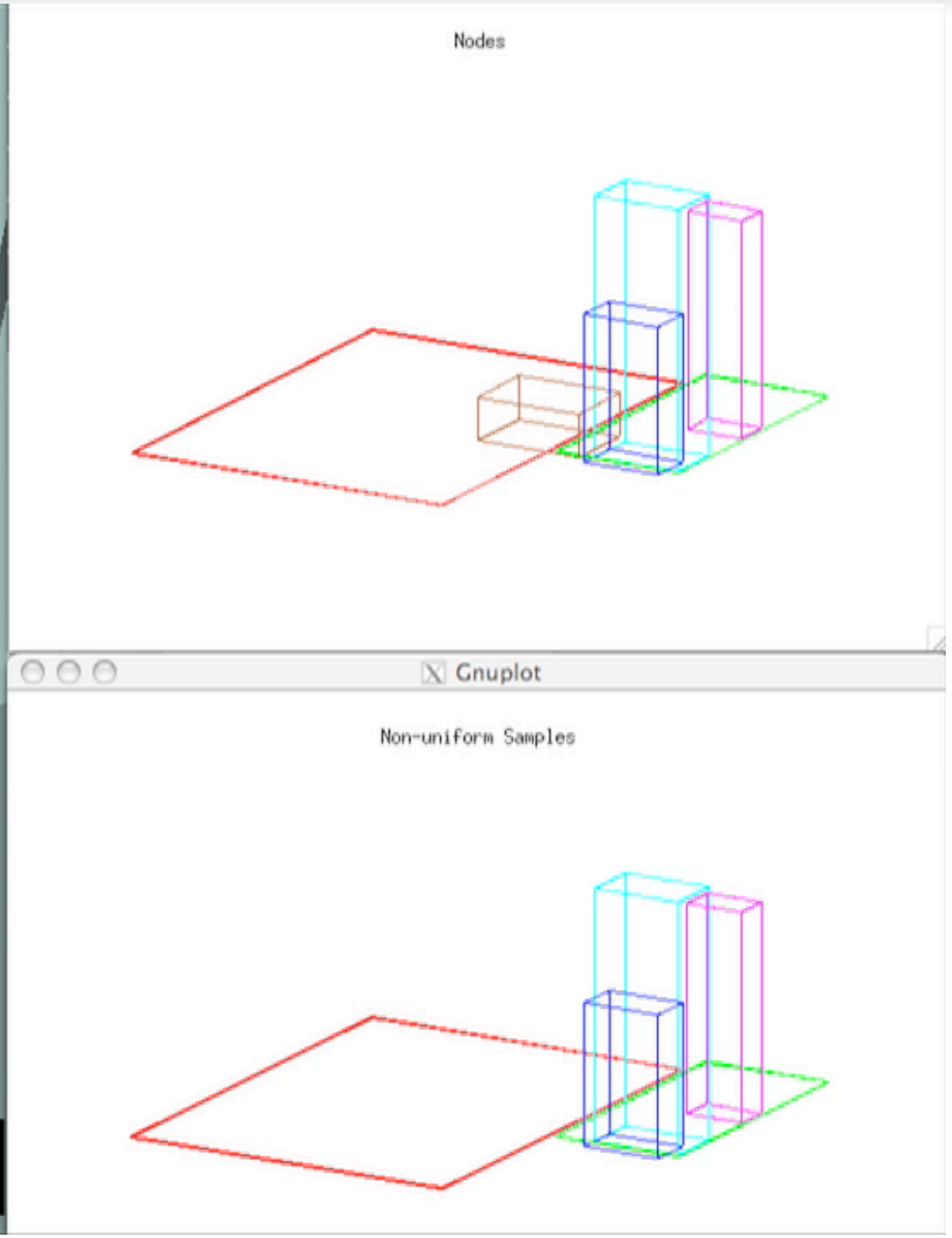


Aufgaben-orientiert

implizit

Aufgabenorientierte Programmierung: Beispiel





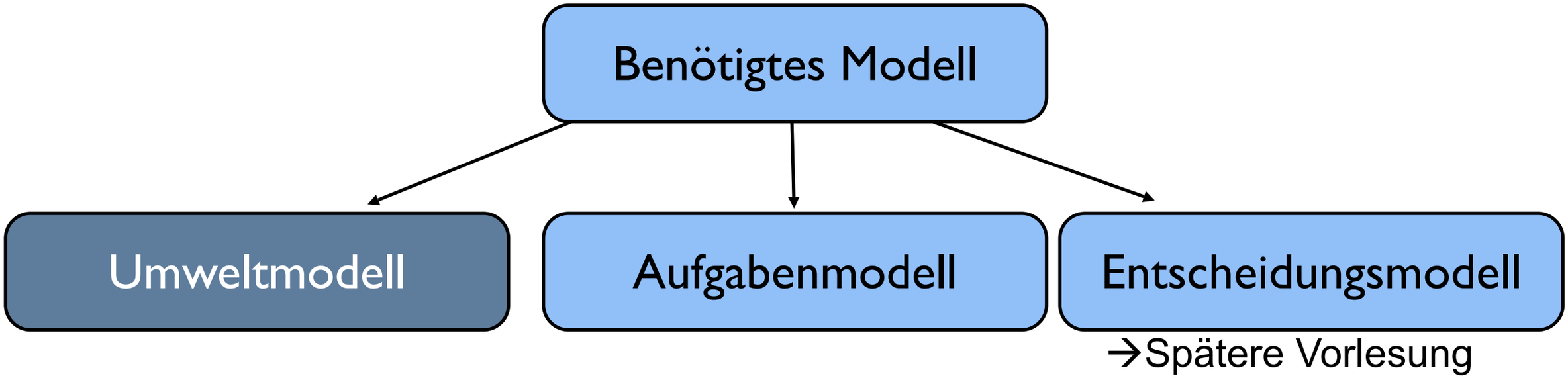
Modellierung

Verschiedene Modelle werden bei der Programmierung von Robotern benötigt:

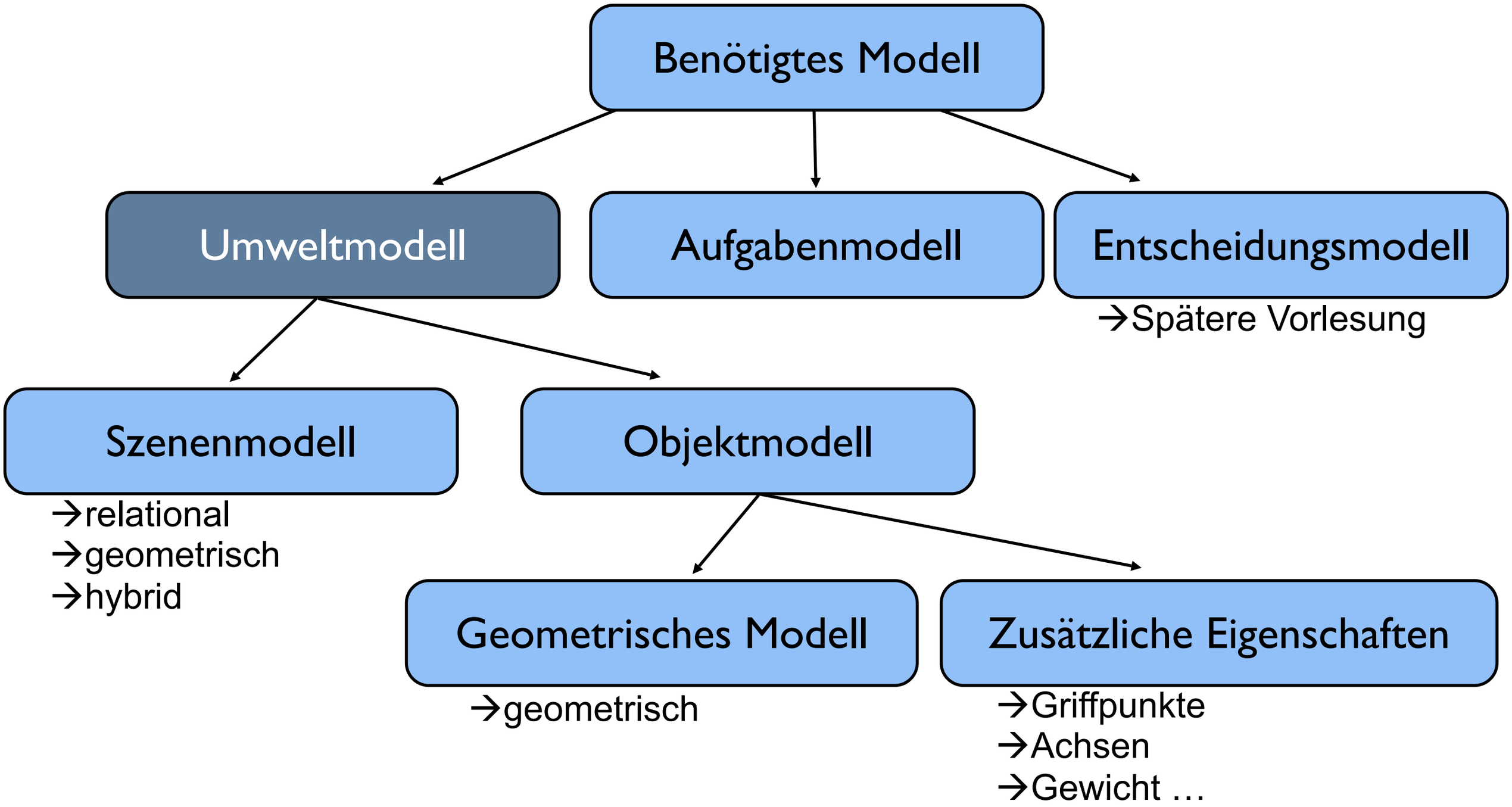
- Umweltmodell (Objekte und Szene)
- Aufgabenmodell
- Modell der Kinematik
- Modell der Dynamik
- Modelle der Sensoren

Die Bereiche „Kinematik“ und „Dynamik“ werden in der Vorlesung „Robotik I“, der Bereich „Sensoren“ in der Vorlesung „Robotik III“ behandelt.

Aufgabenorientierte Programmierung



Aufgabenorientierte Programmierung

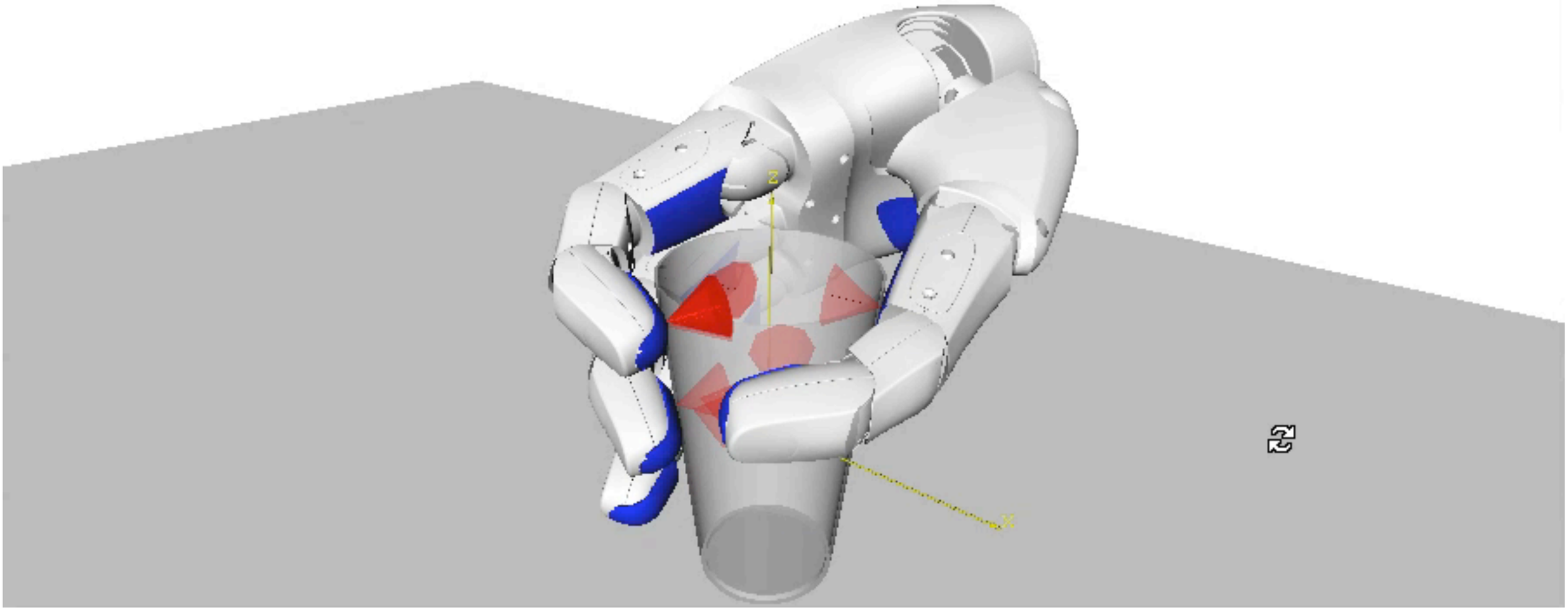


Umweltmodell

- Objektmodell
  - Geometrische Beschreibung
  - Modellierung von zusätzlichen Objekteigenschaften
- Szenenmodellierung

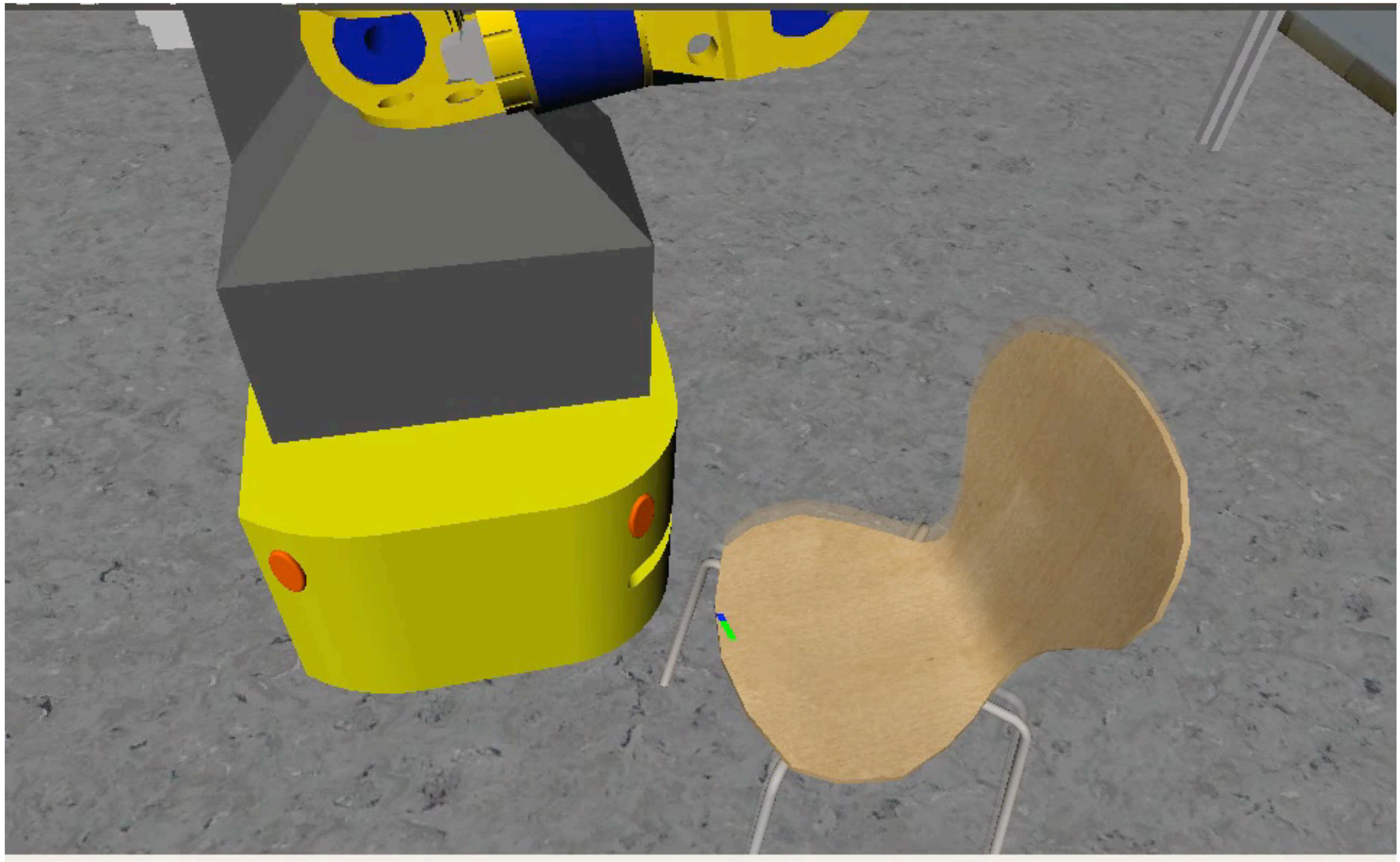
Geometrische Beschreibung: Motivation

Kollisionsberechnungen, Kontaktberechnung in Griffplanung, ...



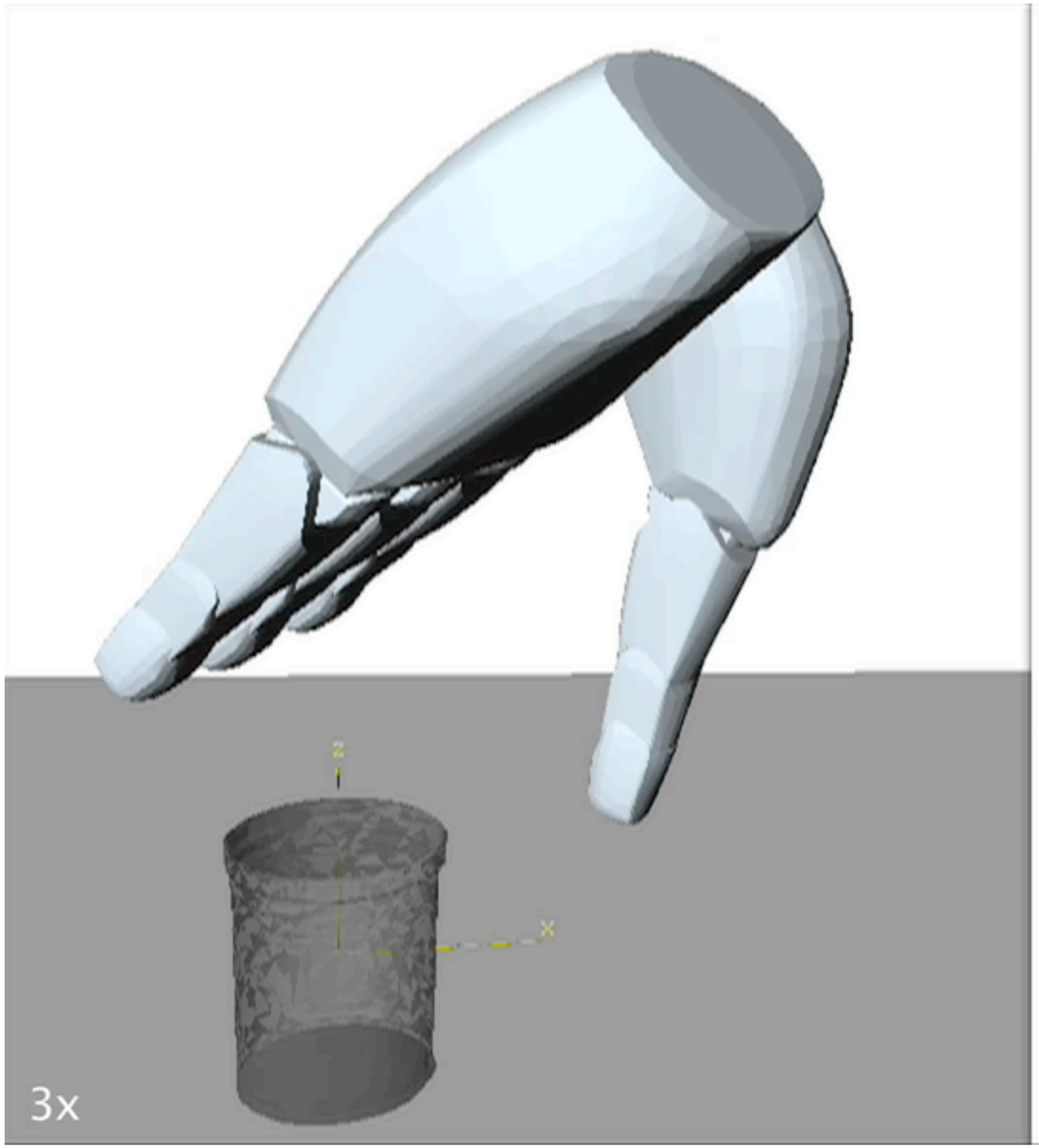
Geometrische Beschreibung: Motivation

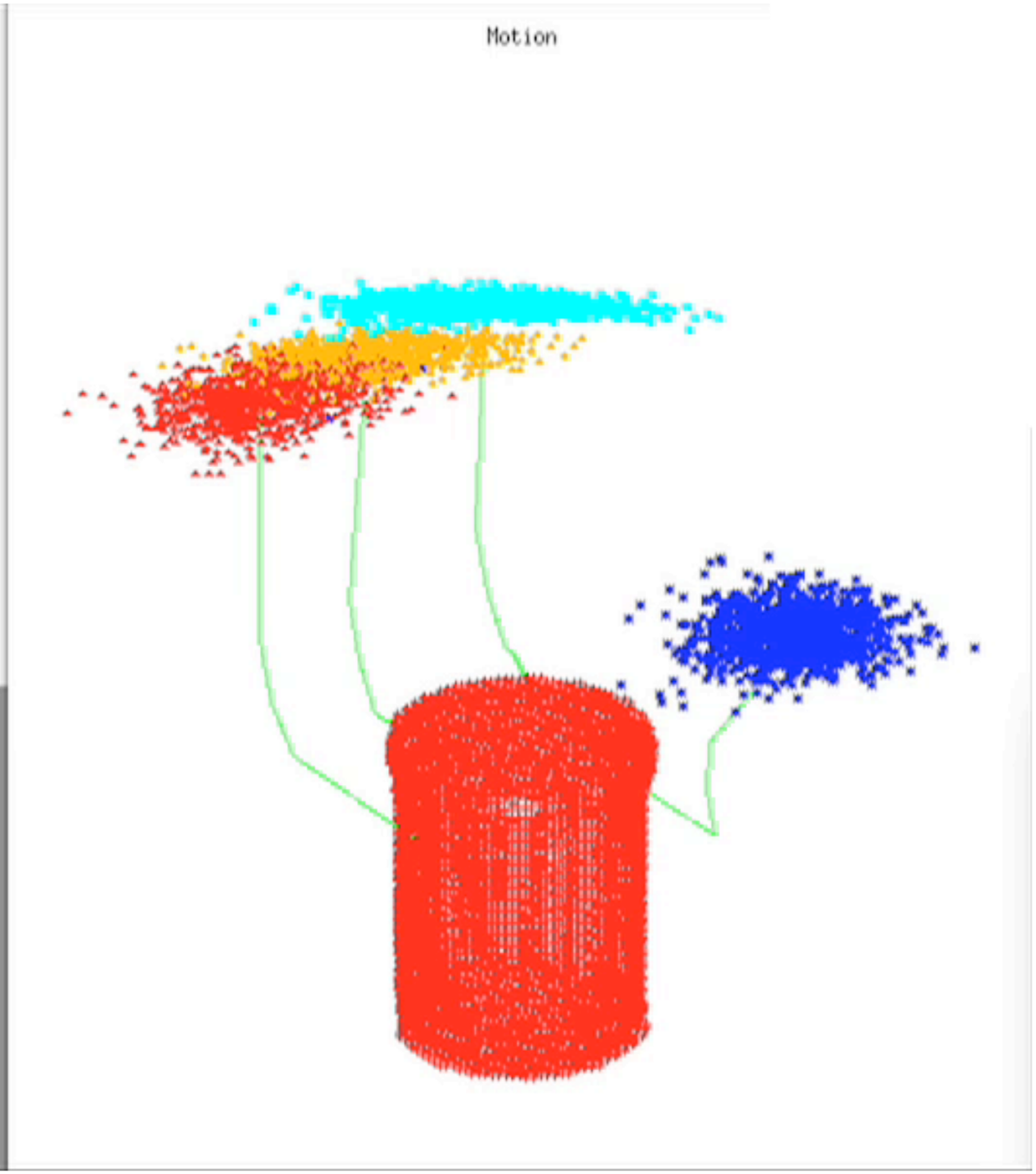
Simulation der Effekte von Handlungen auf die Umwelt



Geometrische Beschreibung: Motivation

Geometrische Beschreibung: Motivation





Geometrische Beschreibung

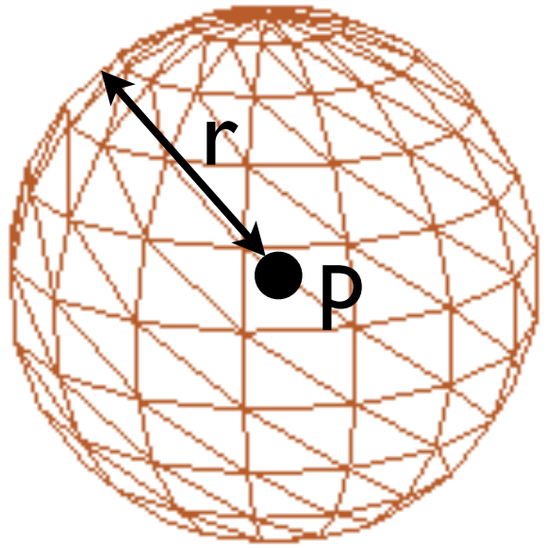
wichtig für:

- graphische Darstellung
- Berechnung von Kollisionen und Kontakten
- physikalisch-dynamische Simulation
- geometriebezogene Bewegungsplanung

Flächen: analytisch

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



Flächen: approximativ

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

→ Approximation

Vorteile:

+ Definition sehr einfach
+ einfache Algorithmen

Nachteile:

- hoher Speicherbedarf
- hoher Rechenaufwand

Dreiecksflächen

Approximation von Freiformflächen:

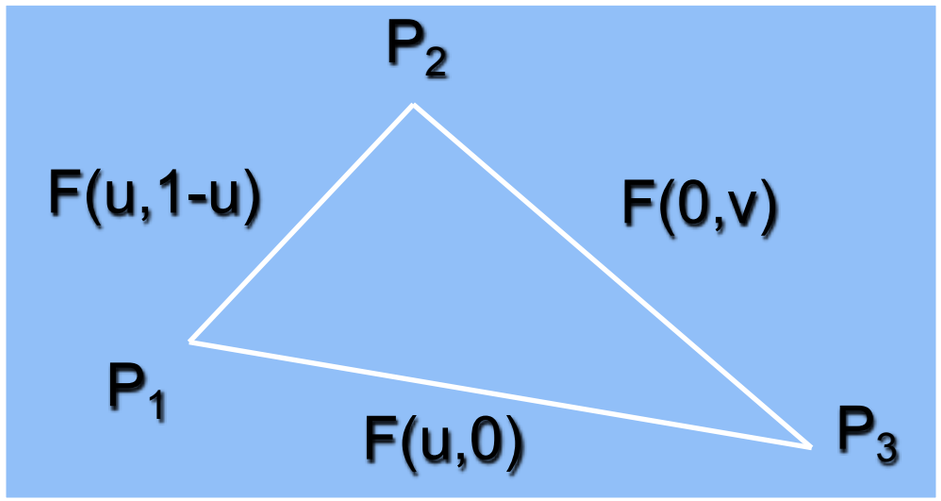
Die einfachste Fläche ist die **Dreiecksfläche**

Definition:

Gegeben seien 3 Punkte im Raum P_1, P_2, P_3

Damit hat die Fläche folgende Gleichung:

$$F(u,v)=u \bullet P_1+v \bullet P_2+(1-u-v) \bullet P_3 \quad \text{mit} \quad 0 \le u,v,\ u+v \le 1.$$



Bilineare Viereckselemente / Pflaster

Definition:

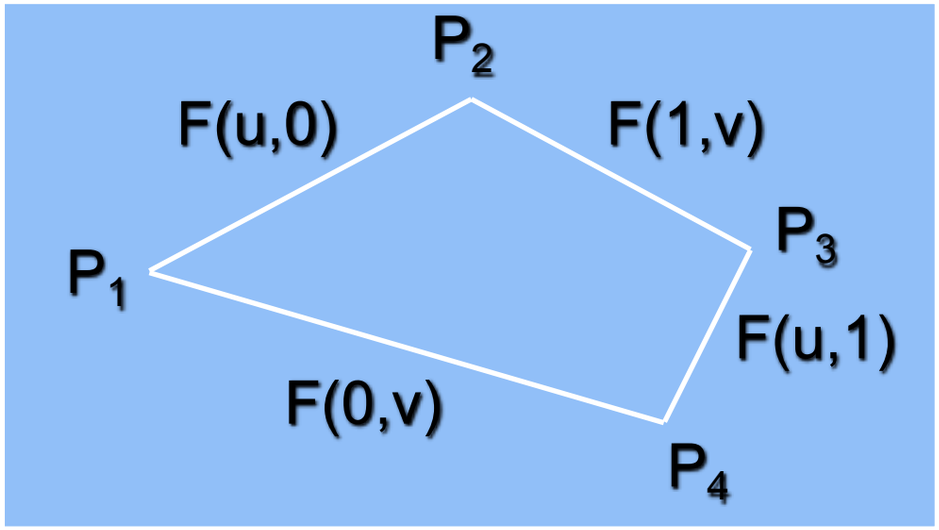
Gegeben sind 4 Punkte im Raum P_1, P_2, P_3, P_4

Damit wird die Fläche definiert durch

$$F(u,v)=(1-u)(1-v) \bullet P_1 + (1-u)v \bullet P_2 + u(1-v) \bullet P_3 + uv \bullet P_4$$

mit $0 \le u \le 1$,

$0 \le v \le 1$.



Vorteil:

+ Flächenelemente können gekrümmt sein

  ⇒ weniger Gitterpunkte bei gleich guter Approximation

Nachteil:

- Rechnen mit gekrümmten Flächen ist aufwendig

Bilineare Viereckselemente / Pflaster

Definition:

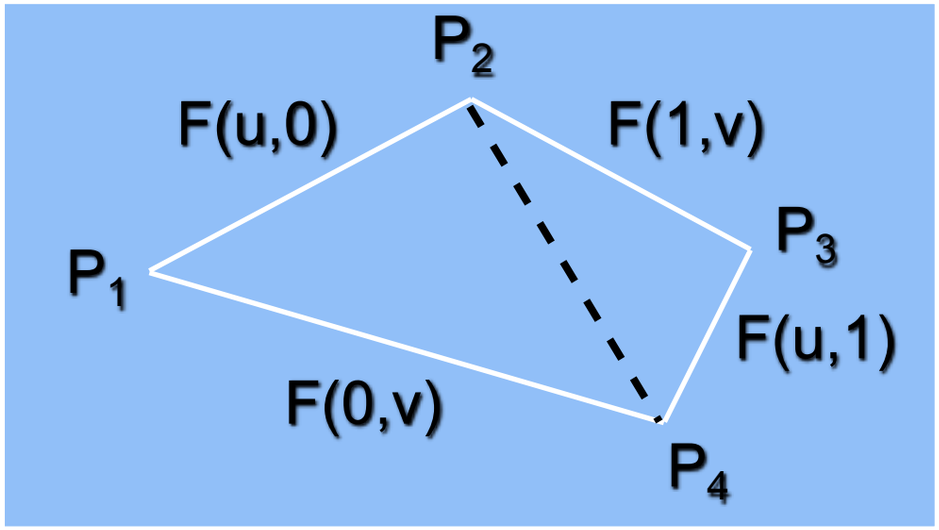
Gegeben sind 4 Punkte im Raum P_1, P_2, P_3, P_4

Damit wird die Fläche definiert durch

$$F(u,v)=(1-u)(1-v) \cdot P_1 + (1-u)v \cdot P_2 + u(1-v) \cdot P_3 + uv \cdot P_4$$

mit $0 \le u \le 1$,

$0 \le v \le 1$.



Vorteil:

+ Flächenelemente können gekrümmt sein

  ⇒ weniger Gitterpunkte bei gleich guter Approximation

Nachteil:

- Rechnen mit gekrümmten Flächen ist aufwendig

Bilineare Viereckselemente / Pflaster

Definition:

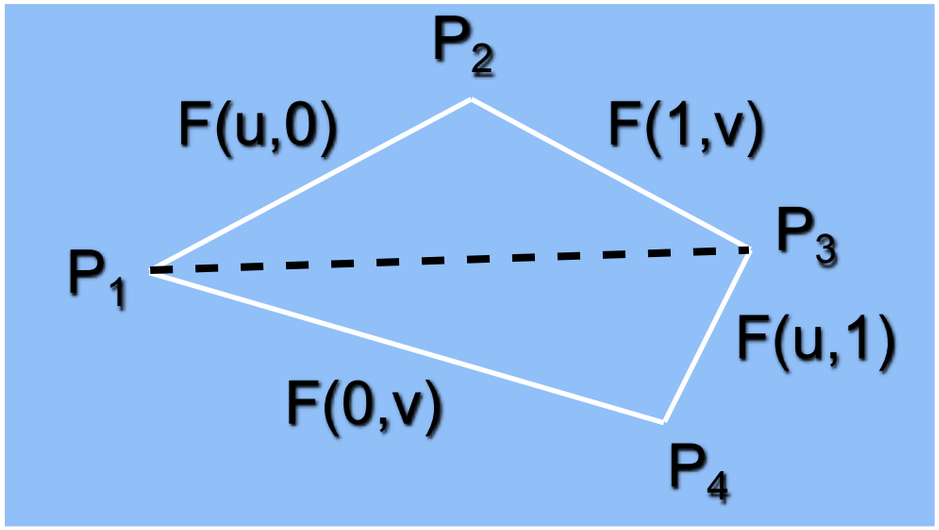
Gegeben sind 4 Punkte im Raum P_1, P_2, P_3, P_4

Damit wird die Fläche definiert durch

$$F(u,v)=(1-u)(1-v) \cdot P_1 + (1-u)v \cdot P_2 + u(1-v) \cdot P_3 + uv \cdot P_4$$

mit $0 \le u \le 1$,

$0 \le v \le 1$.



Vorteil:

+ Flächenelemente können gekrümmt sein

  ⇒ weniger Gitterpunkte bei gleich guter Approximation

Nachteil:

- Rechnen mit gekrümmten Flächen ist aufwendig

Bezierflächen

Erweiterung des Ansatzes der Bezierkurven.

Definition: Gegeben ist ein Gitter von Führungspunkten P_{ij}

$0 \le i \le N$ und $0 \le j \le M$

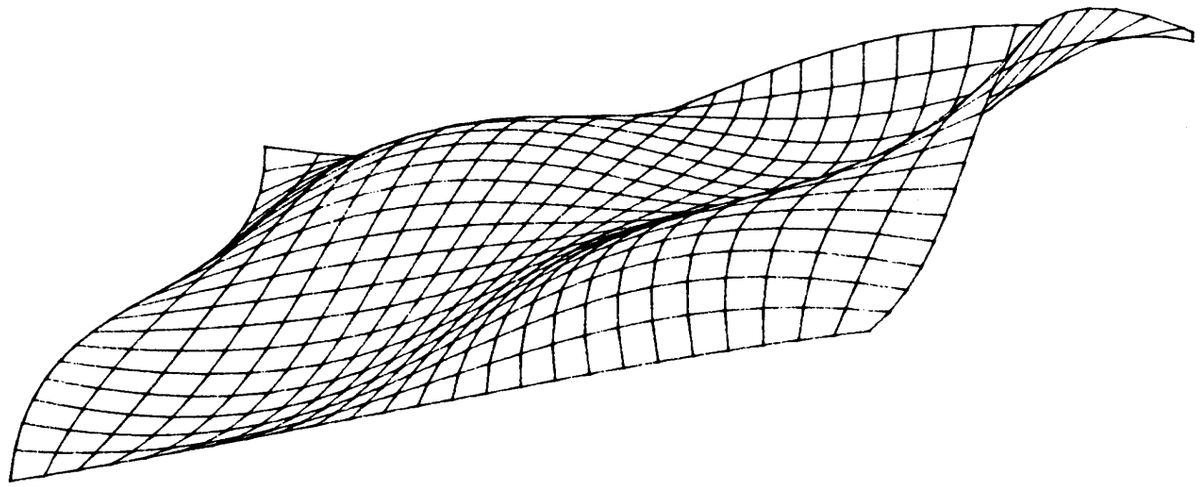
Damit ist die Fläche beschrieben durch

$$F(u,v) = \sum_{i=0}^{N} \sum_{j=0}^{M} P_{ij} \cdot B_{i,N}(u) \cdot B_{j,M}(v)$$

mit $B_{i,N}(u) = (1-u)B_{i,N-1}(u)+uB_{i-1,N-1}(u)$

$B_{j,M}(v) = (1-v)B_{j,M-1}(v)+vB_{j-1,M-1}(v)$

Die $B_{i,N}$ bzw. $B_{j,M}$ heißen auch Bernsteinpolynome.



3D Objektmodell

- Geometrische Modellierung von Objekten

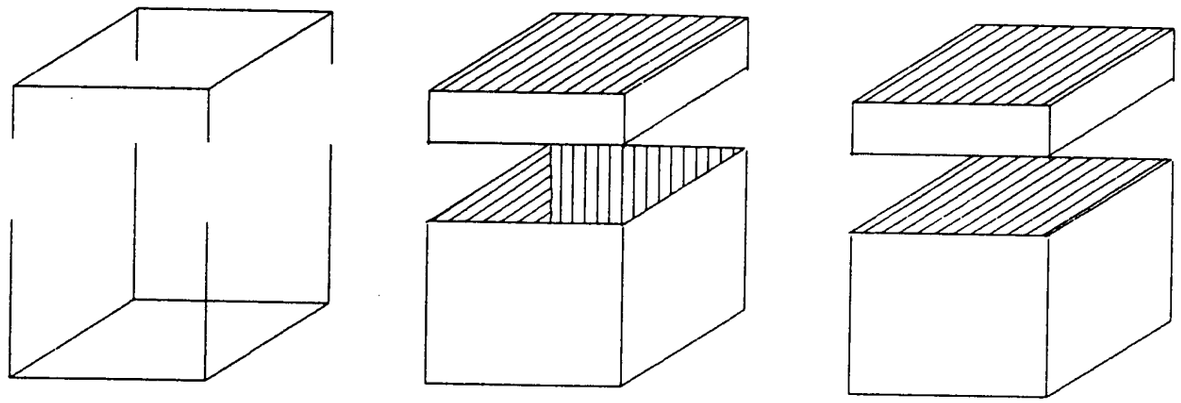
1. Kantenmodell
2. Flächenmodell
3. Volumenmodell

Kantenmodell

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

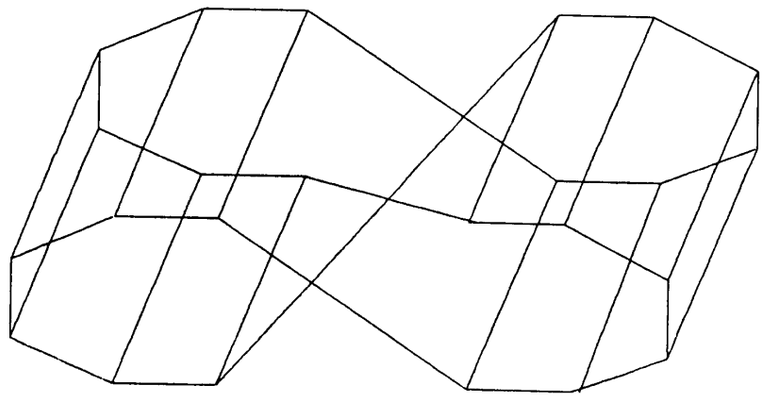
Vorteile:

+ einfache Daten
+ wenige Daten



Nachteile:

- Mehrdeutigkeiten
- hoher Eingabeaufwand
- keine Kollisionsberechnung
- kein Schnitt



Flächenmodell

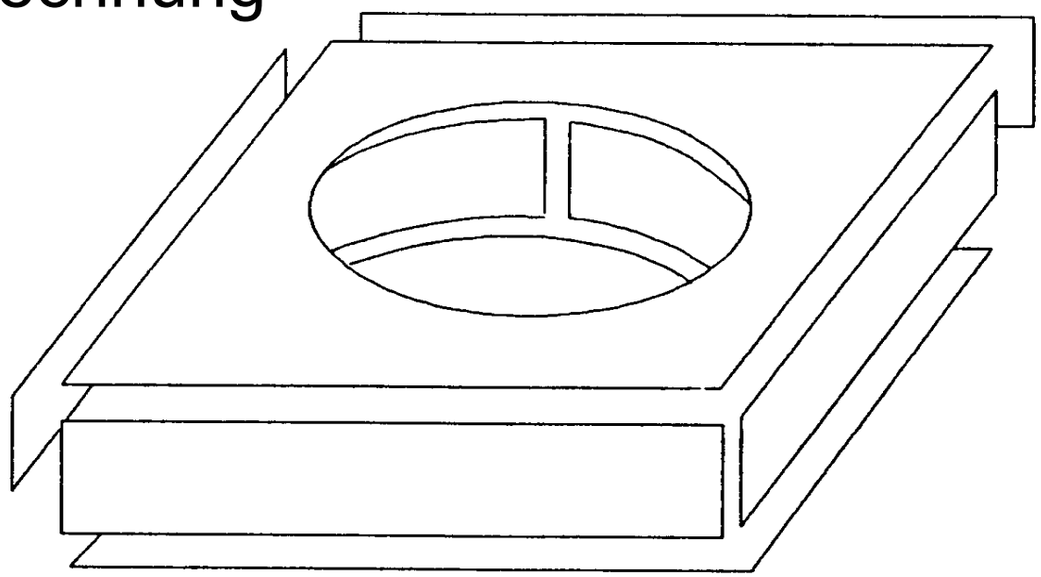
Speicherung von Kanten und Oberflächen (Dreiecke, Bezier, ...)

Vorteile:

+ effiziente Verfahren
+ entspricht dem Vorgehen während der Modellierung
+ schnelle Kollisions- und Abstandsberechnung

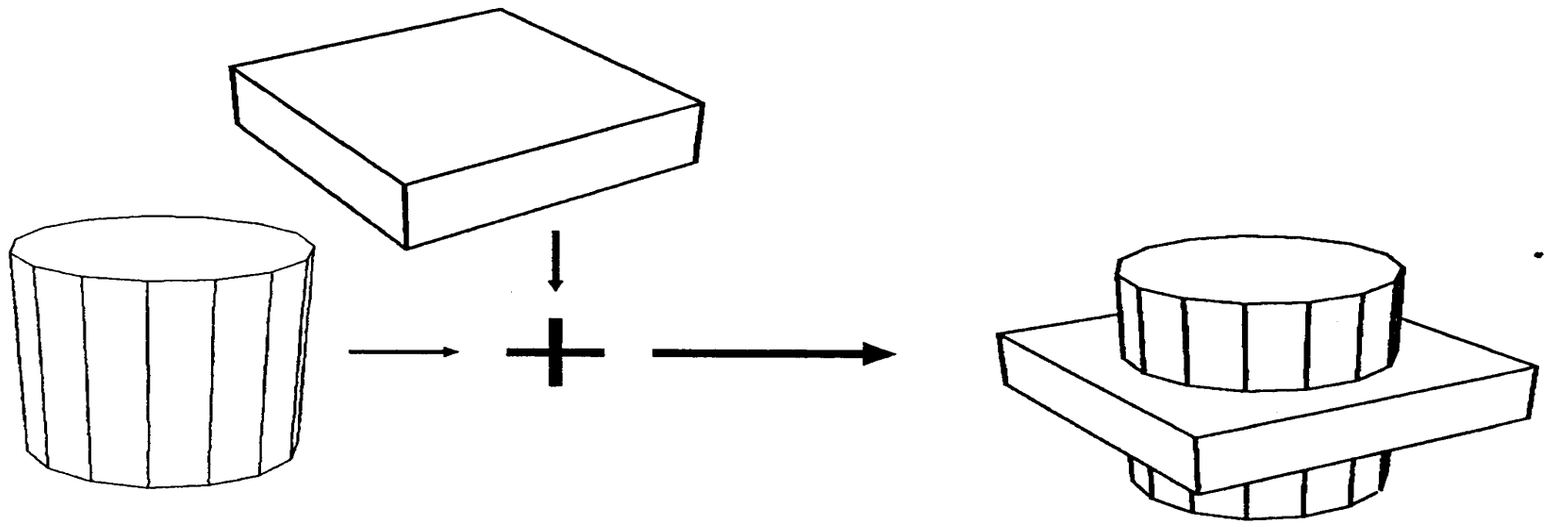
Nachteile:

- hoher Eingabeaufwand
- Darstellung aufwendig
- Problem bei Schnittoperationen
- Inkonsistenzen möglich



Volumenmodelle: Parametrische Modelle

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



Vorteile:

+ eindeutige Objektbeschreibung
+ geringer Eingabeaufwand
+ Ergebnis von Operationen sind korrekte Objekte

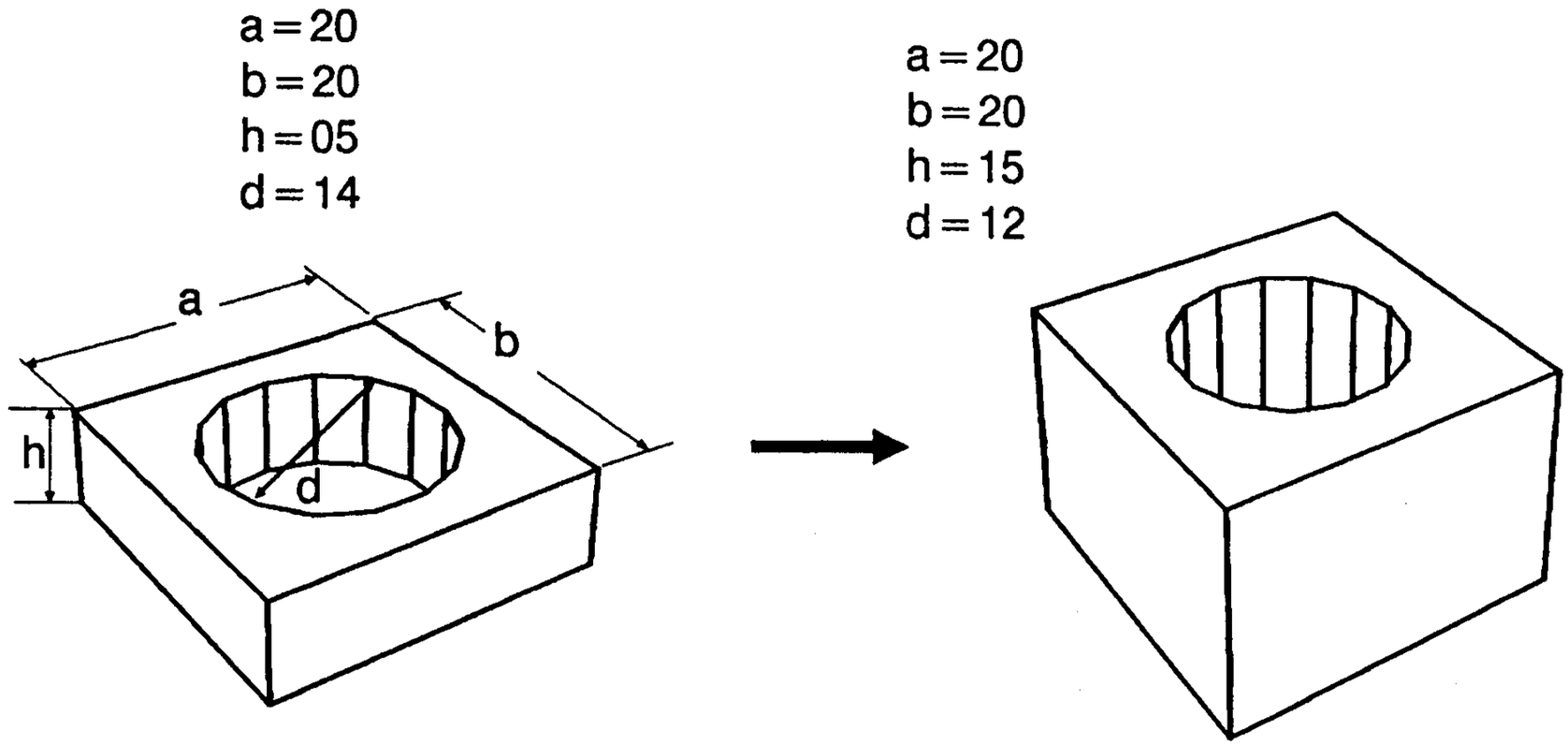
Nachteile:

- hoher Implementierungsaufwand
- Einbindung von Freiformflächen schwierig

Volumenmodell: Parametrische Modelle

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

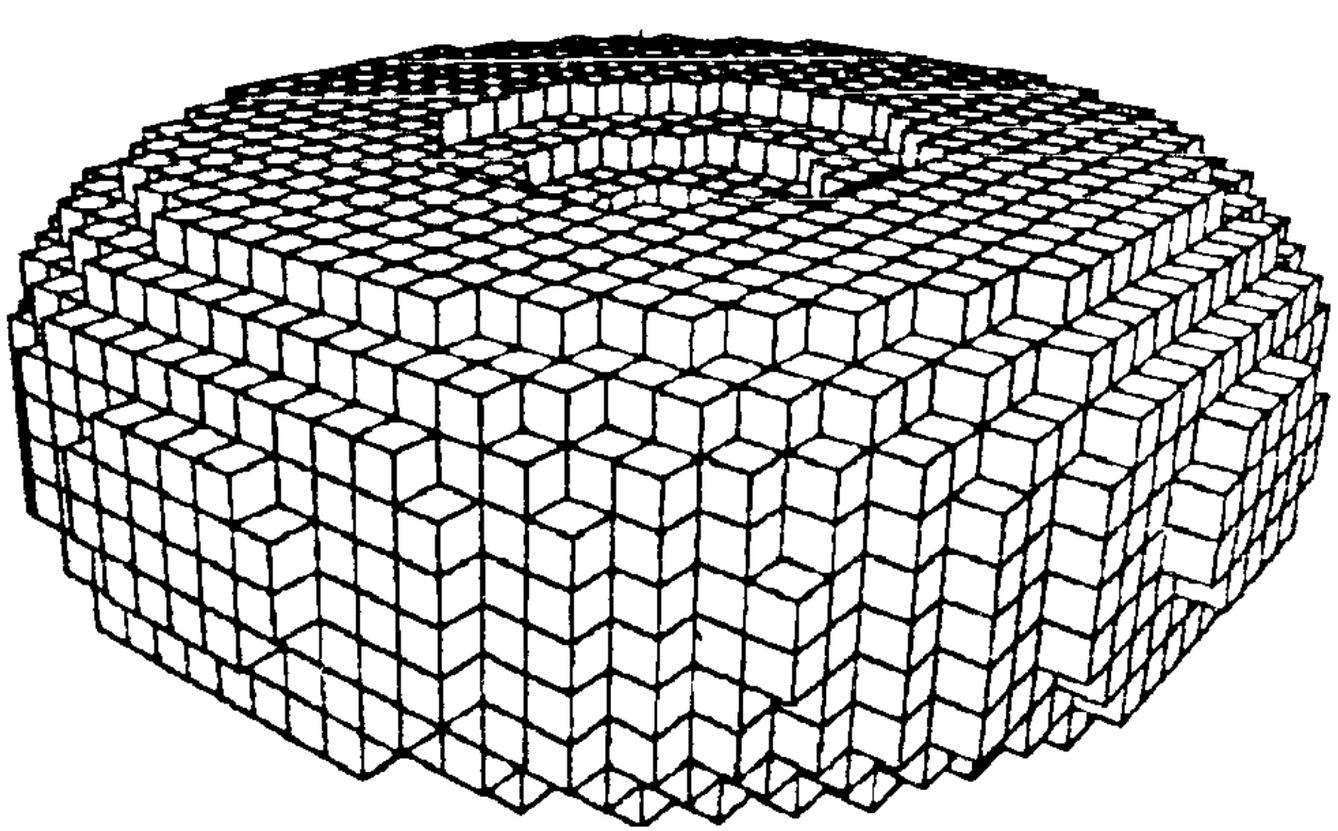
Konsistenzprüfungen sind notwendig !!! (d < a)



Volumenmodell: Zellenzerlegung

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

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



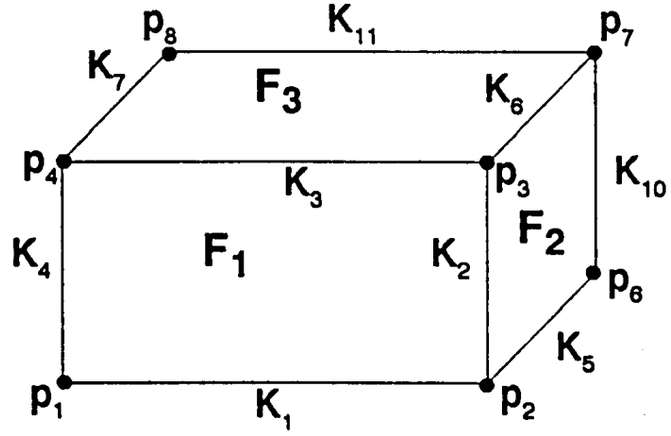
Umsetzung der Modelle

- Boundary Repräsentation
- Constructive Solid Geometry (CSG)
- Zellenbelegung

Boundary Repräsentation

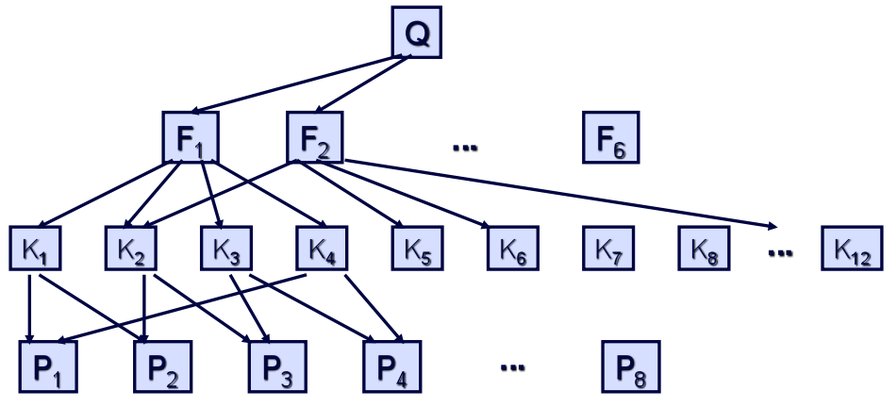
Boundary - Repräsentation (B-Rep)

Hierarchische Darstellung eines Objektes durch begrenzende Elemente, i.d.R. Kanten oder Flächen.



Elemente eines Quaders im Flächenmodell
Elemente:

- Q : Quader
- F_i : $i \in \{1,...,6\}$: Flächen
- K_i : $i \in \{1,...,12\}$: Kanten
- P_i : $i \in \{1,...,8\}$: Ecken



Boundary Repräsentation: Vorteile

Aus topologischer Struktur Information über z.B.:

- 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?

siehe Dillmann: „Informationsverarbeitung in der Robotik“

Boundary Repräsentation: Vorteile

Aus topologischer Struktur Information über z.B.:

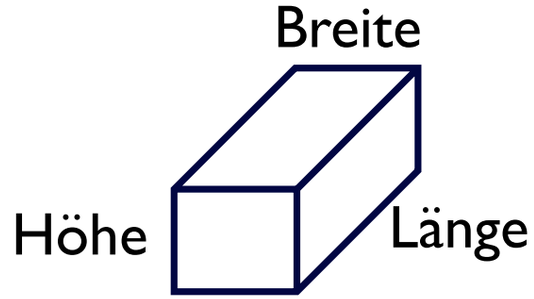
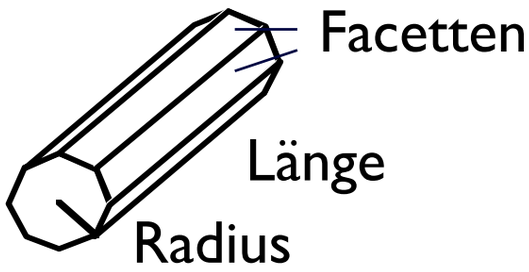
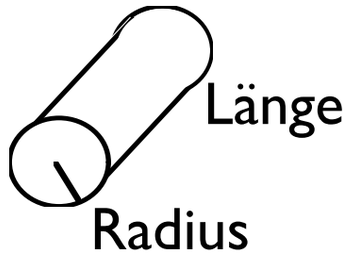
- Welche Flächen gehören zum Objekt?
- Welche Kanten gehören zur Fläche?
- Zu welchem Objekt gehört eine Fläche?
- Zu welchem Objekt gehört eine Kante?
- Welche Flächen stoßen aneinander?

→ kantenbasierte Objekterkennung

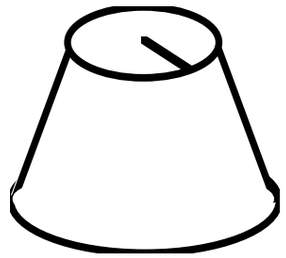
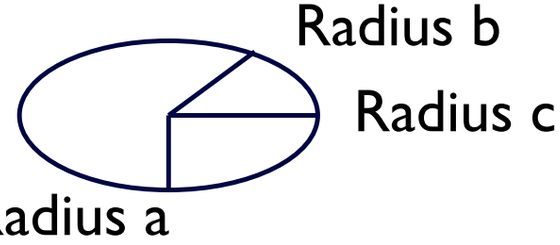
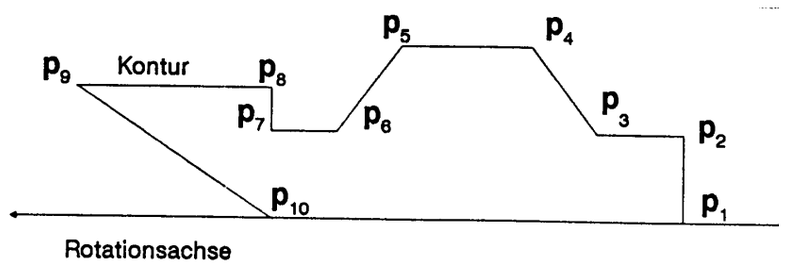
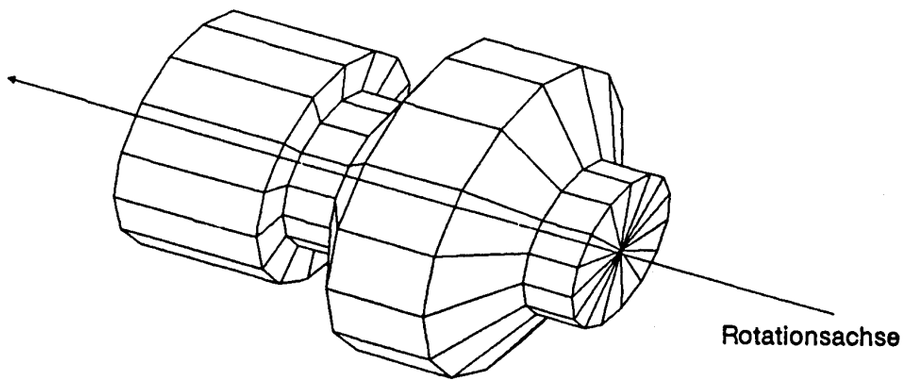
siehe Dillmann: „Informationsverarbeitung in der Robotik“

Constructive Solid Geometry (CSG)

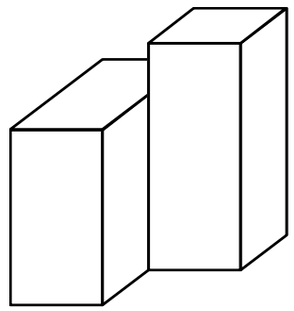
Es gibt eine Menge von einfachen Grundkörpern, die parametriert werden können und auf denen verschiedene Operationen definiert sind.

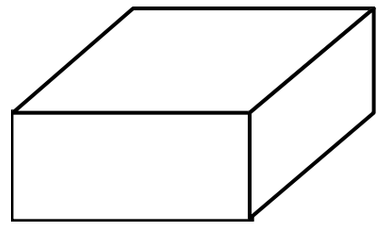
Grundkörper	Parameter	Skizze
Quader	Länge, Breite, Höhe	
Prisma	Länge, Radius, Facetten	
Zylinder	Länge, Radius	

Constructive Solid Geometry (CSG)

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

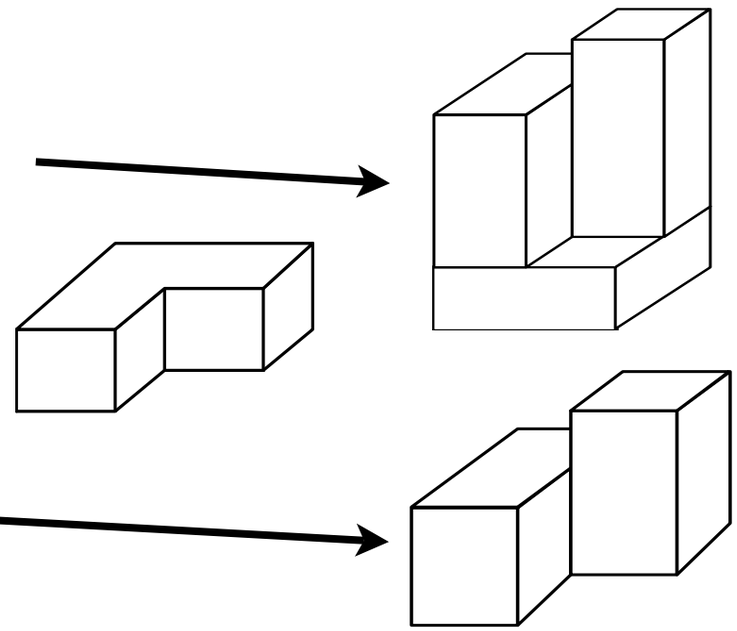
Operatoren (CSG)

Objekt A 

Objekt B 

Operatoren:

- Vereinigung A ∪ B (Summe)
- Schnitt A ∩ B
- Differenz A / B



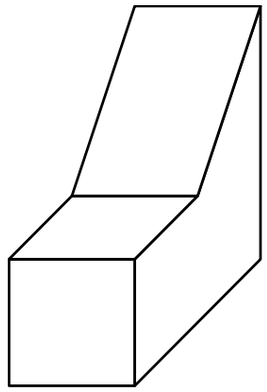
- Sweep:
  Ein Grundelement (u.U. eine Fläche) wird entlang einer Raumkurve verschoben. Der durchdrungene Raum stellt das neue Objekt dar.

Zellenbelegung

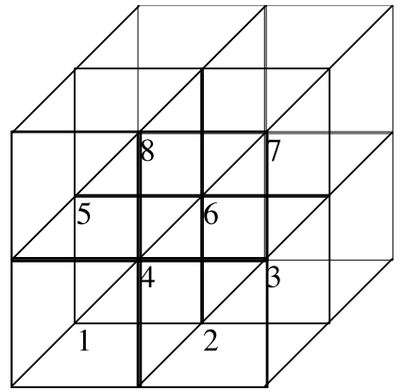
- Der Raum wird in mehrere Zellen unterteilt (i.d.R. 8 Zellen: „Octree“).
- Zelle komplett vom Objekt belegt → als„belegt“ markieren
- Wenn die Zelle nur teilweise belegt ist, dann wird auf diese Zelle das Verfahren rekursiv angewendet. Ansonsten ist die Zelle leer.
- Die Rekursion terminiert bei einer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

Zellenbelegung (Beispiel)

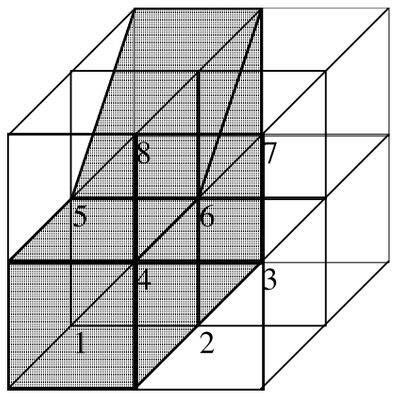
Körper



Zerlegung



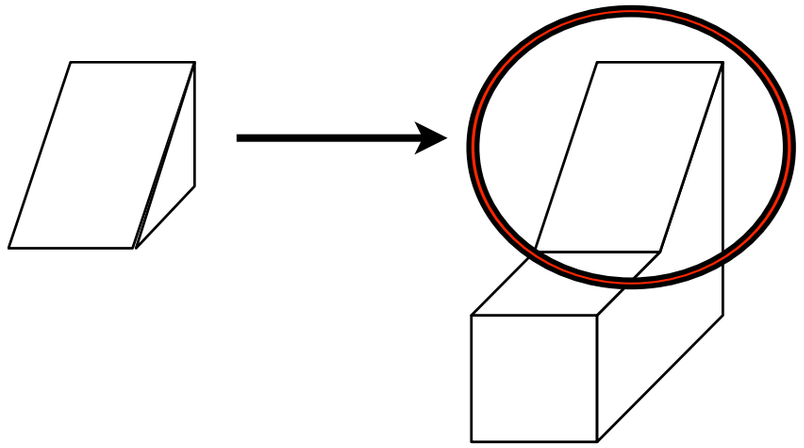
1. Zerlegung



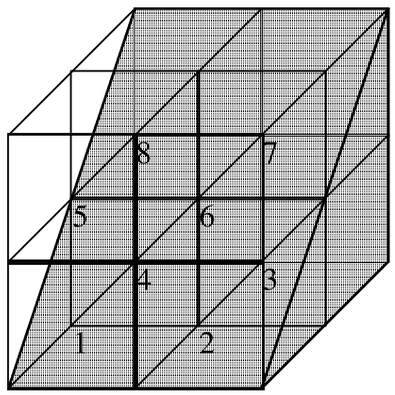
Zelle 1	belegt
Zelle 2	frei
Zelle 3	frei
Zelle 4	belegt
Zelle 5	frei
Zelle 6	frei
Zelle 7	frei
Zelle 8	teilbelegt

Zellenbelegung (Beispiel)

Restkörper

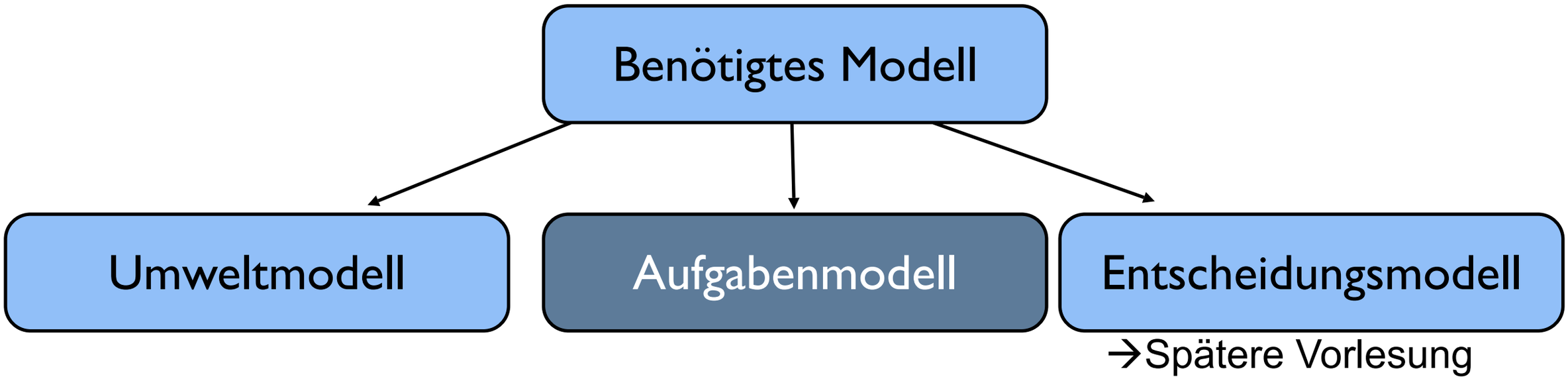


2. Zerlegung

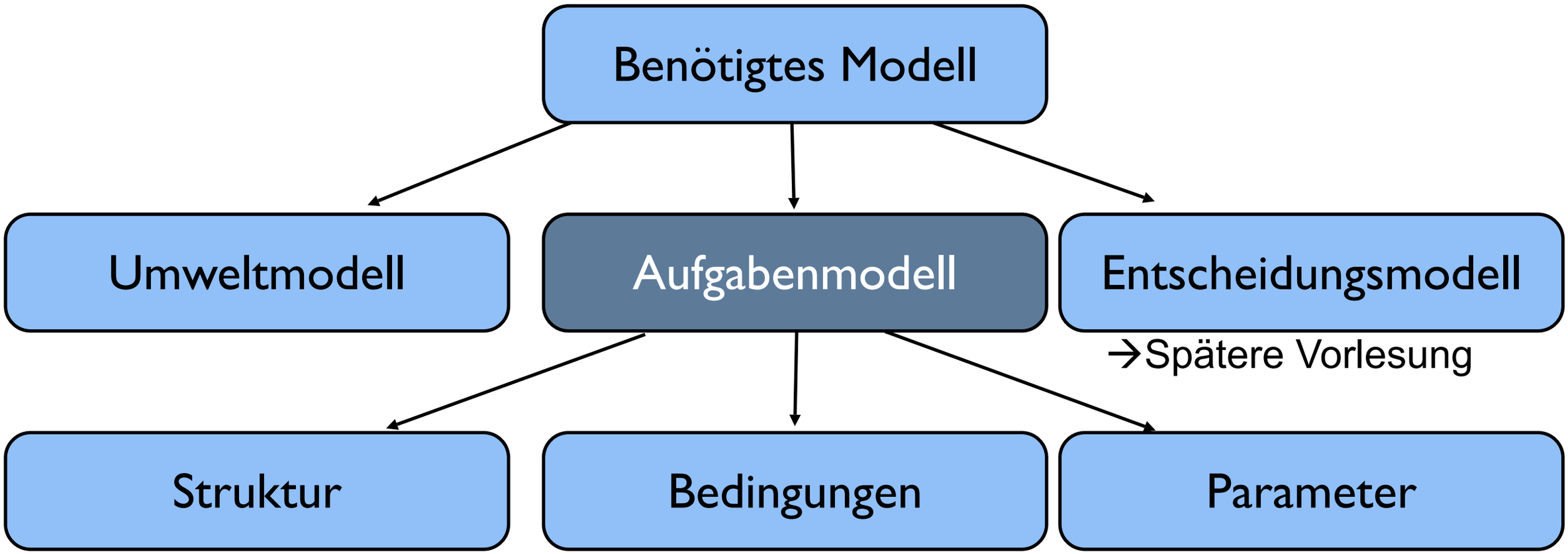


Zelle 1	teilbelegt
Zelle 2	teilbelegt
Zelle 3	belegt
Zelle 4	belegt
Zelle 5	frei
Zelle 6	frei
Zelle 7	teilbelegt
Zelle 8	teilbelegt

Aufgabenorientierte Programmierung



Aufgabenorientierte Programmierung



→Reihenfolge
→hierarchisch

→Vorbedingungen
→Während-Bedingungen
→Nachbedingungen

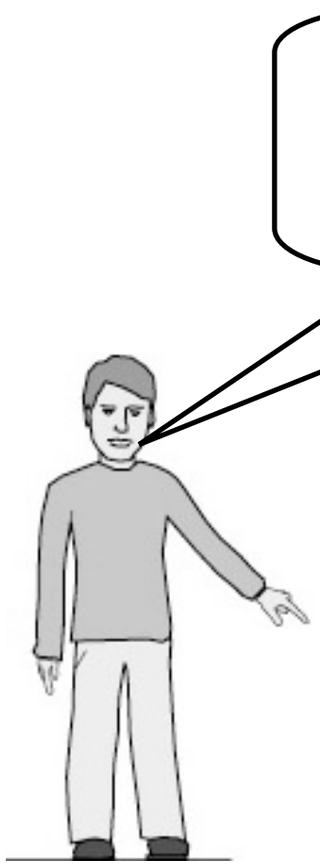
→Bewegung
→Dialog

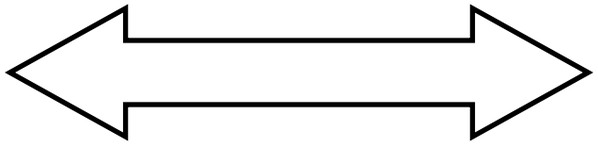
Gliederung

- Symbolische Abstraktion
- Modellierung der Reihenfolge von Operatoren
- Makro-Operatoren
- Abstraktionsebenen im Aufgabenmodell
- Validierung der Modelle
  - Simulation
  - Graphische Animation

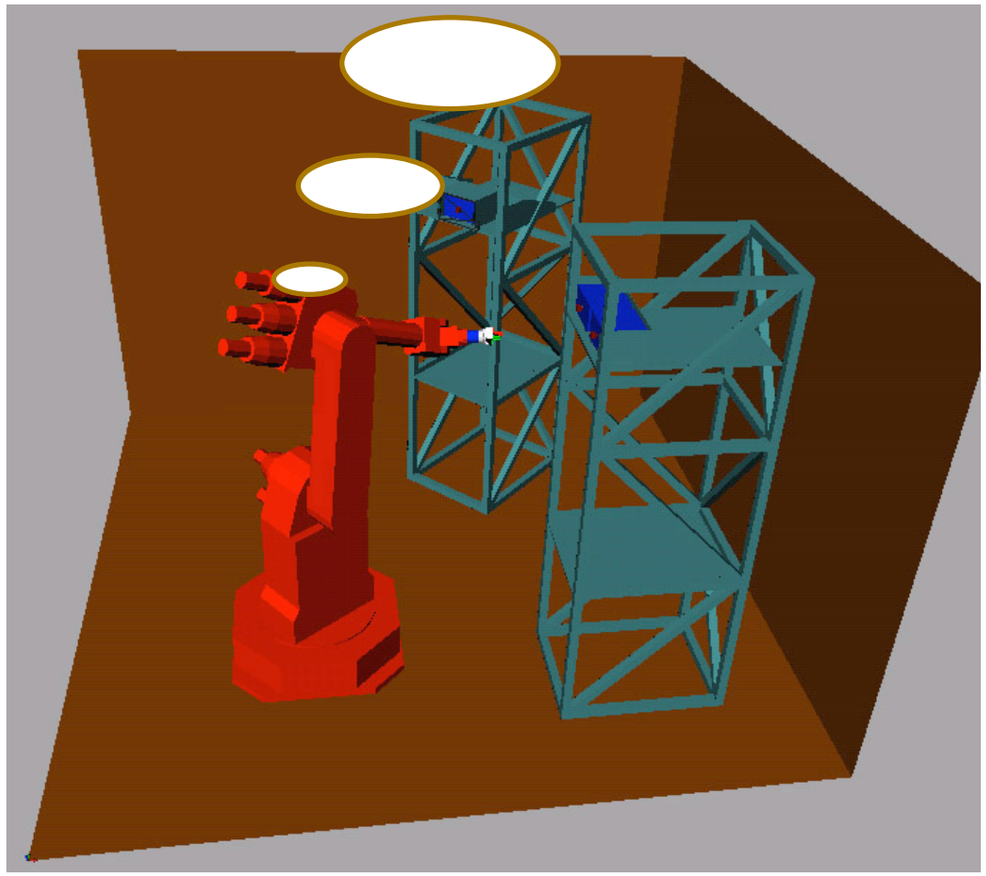
Die kognitive Lücke

Reiche mir die blaue Box





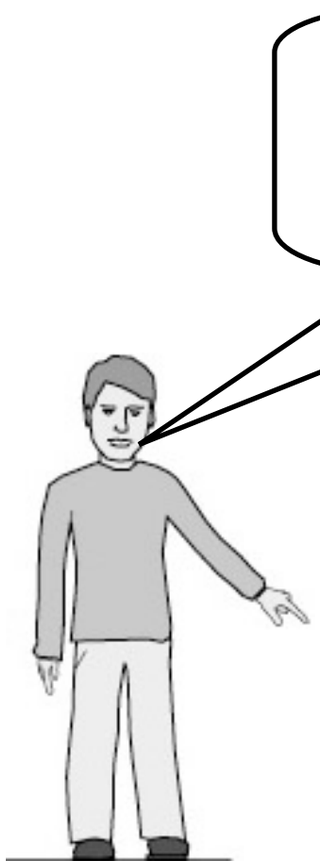
Häh?
MoveJoint (...) ?
MovePos (x,y,z,...)?
Box1 {Farbe: „blau“,Geometrie: Würfel(30x50x20), ...}

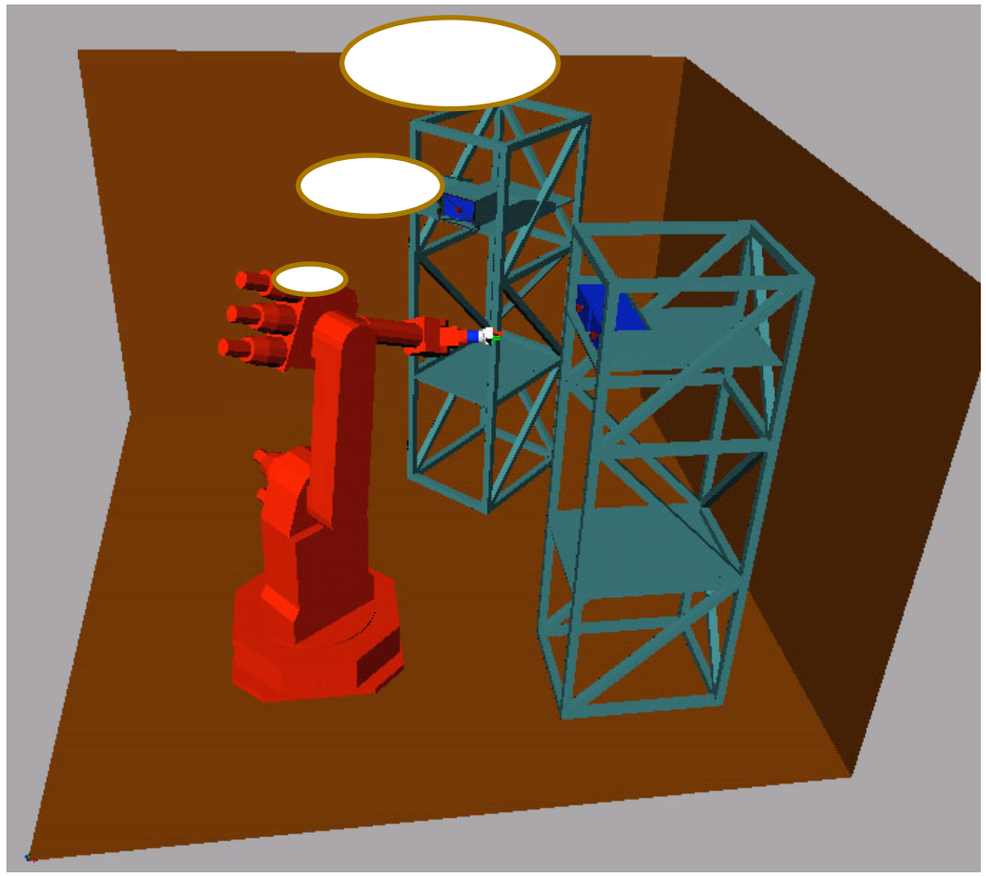


Geometrisches Objektmodell

Häh?
MoveJoint (...) ?
MovePos (x,y,z,...)?
Box1 {Farbe: „blau“,Geometrie: Würfel(30x50x20), ...}

Reiche mir die blaue Box



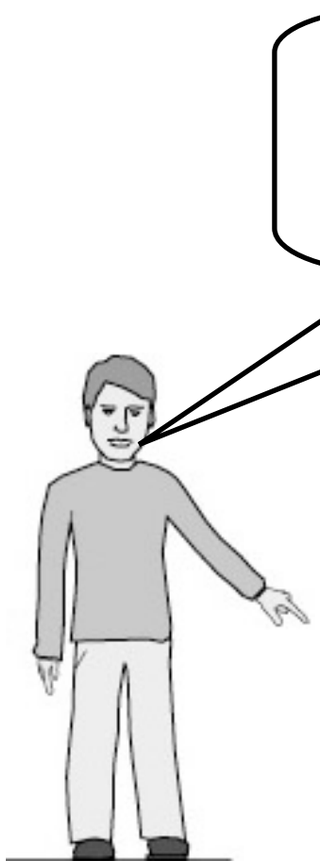


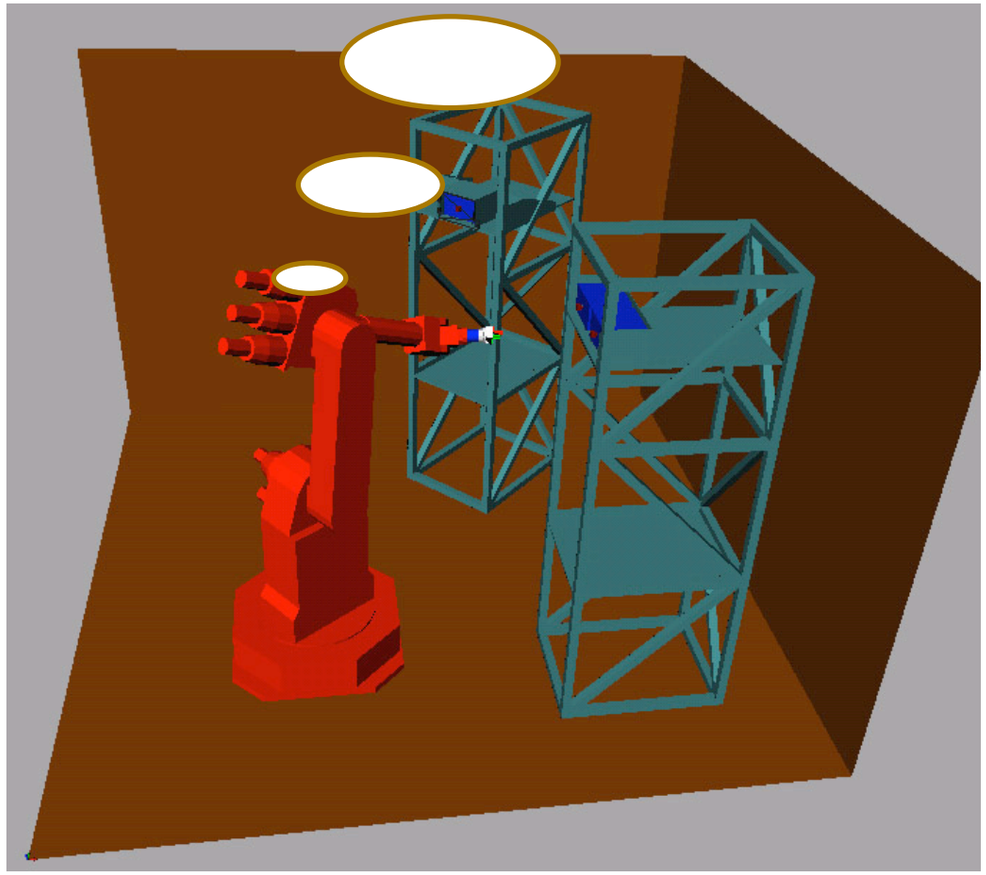
Teil des Szenenmodells

Geometrisches Objektmodell

Häh?
MoveJoint (...) ?
MovePos (x,y,z,...)?
Box1 {Farbe: „blau“,Geometrie: Würfel(30x50x20), ...}

Reiche mir die blaue Box





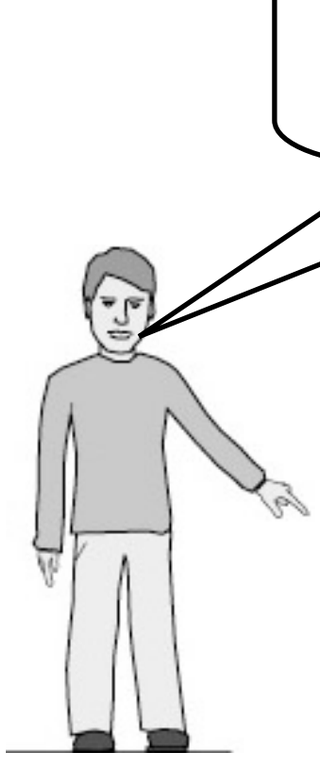
Teil des Aufgabenmodells

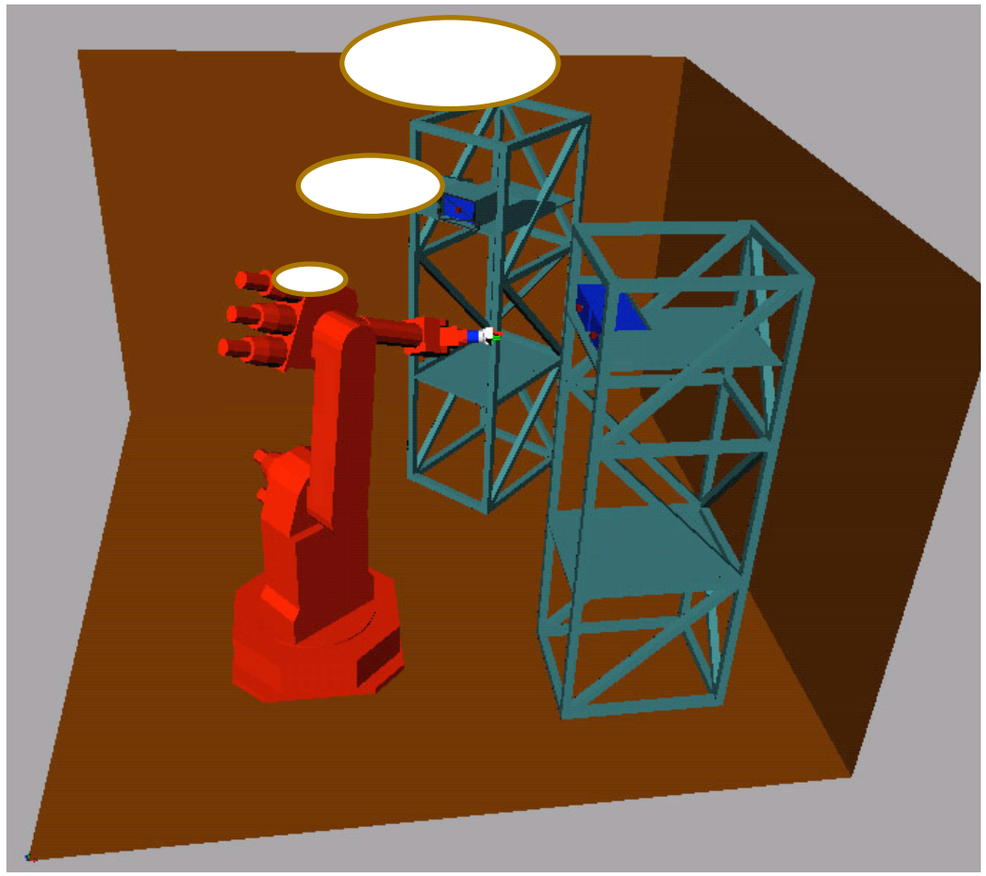
Teil des Szenenmodells

Geometrisches Objektmodell

Häh?
MoveJoint (...) ?
MovePos (x,y,z,...)?
Box1 {Farbe: „blau“,Geometrie: Würfel(30x50x20), ...}

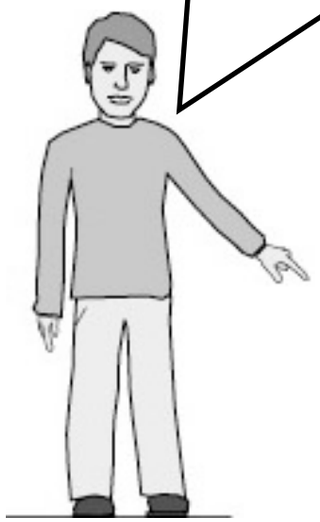
Reiche mir die blaue Box



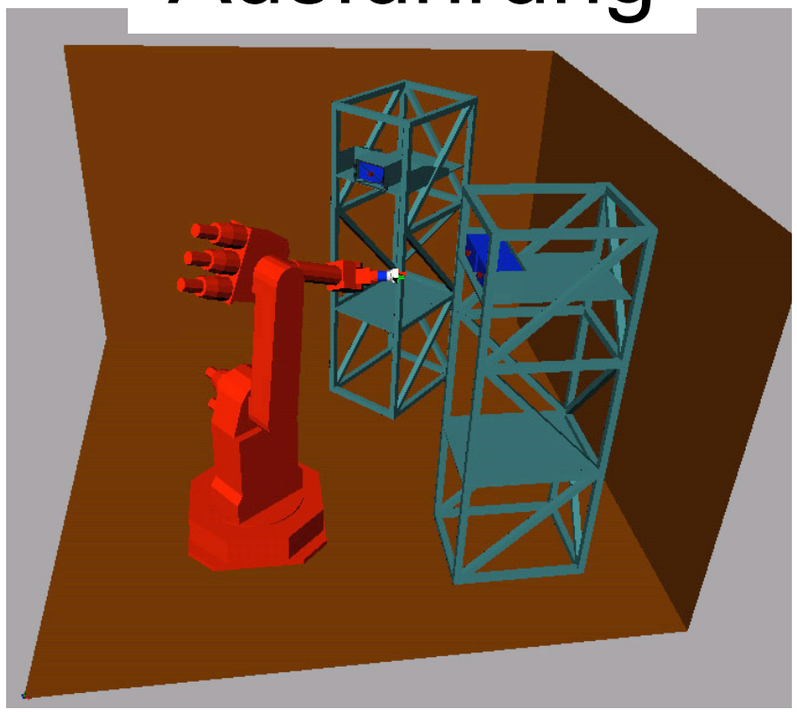


Einordnung des Aufgabenmodells

Eingabe der Aufgabe
„Reiche mir die blaue Box“



Ausführung



Aufgabenmodell

Symbolische Eingabe

Planer

Aktorsteuerung:
Roboterbefehle
Sensorabfragen
...

Umweltmodell

Symbolische Abstraktion

- Benutzer: beschreibt die Aufgabe mit seinen Worten
- Abbildung: Erzeugung der Roboterbefehle über mehrere Stufen.

Bsp. :

- „Bring mir Tee!“

Symbolische Abstraktion

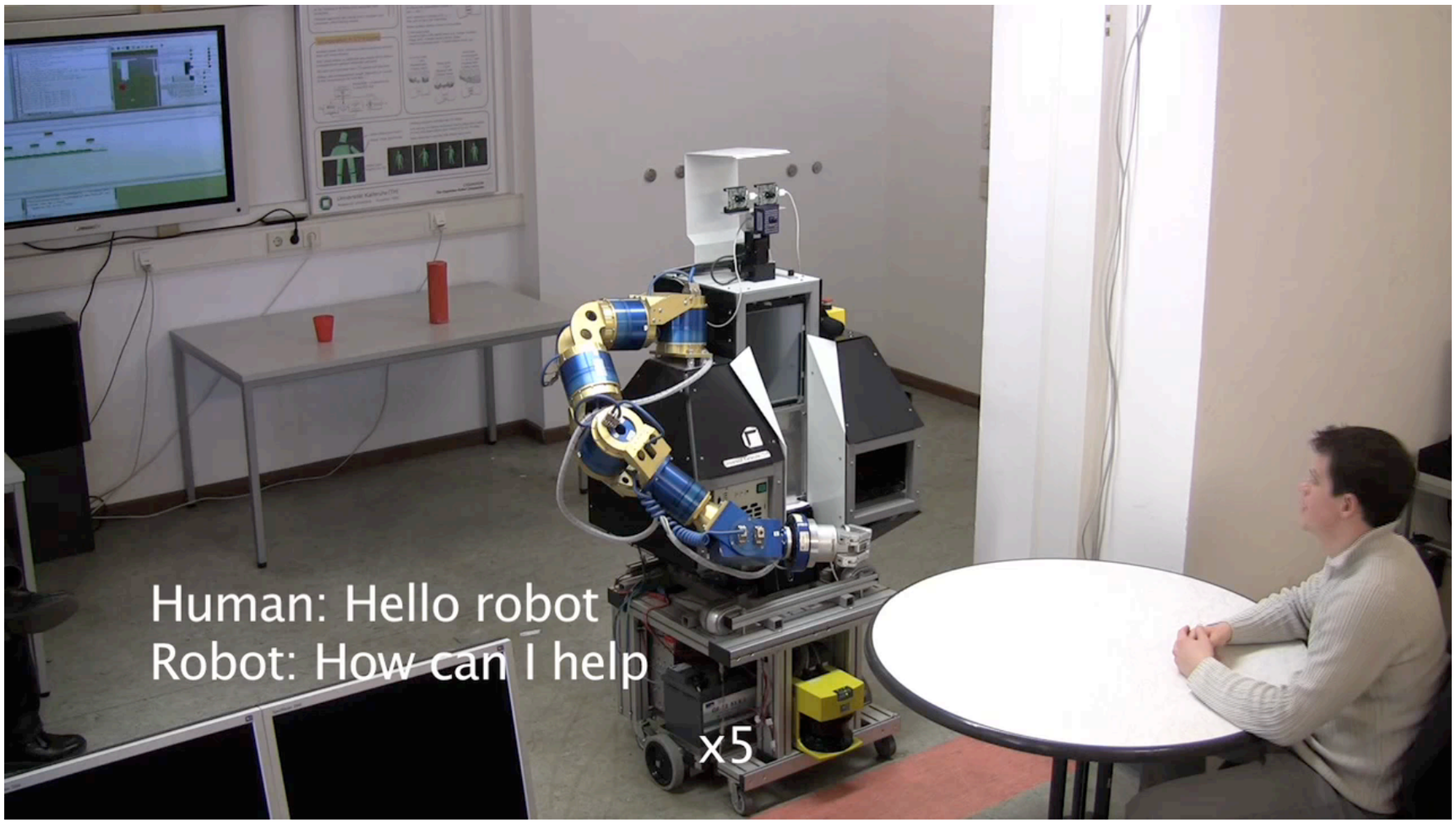
Bsp. : „Bring mir Tee“

➡„Fahre in die ‚Küche‘, greife ein Glas, fahre zurück und reiche mir den Becher.“

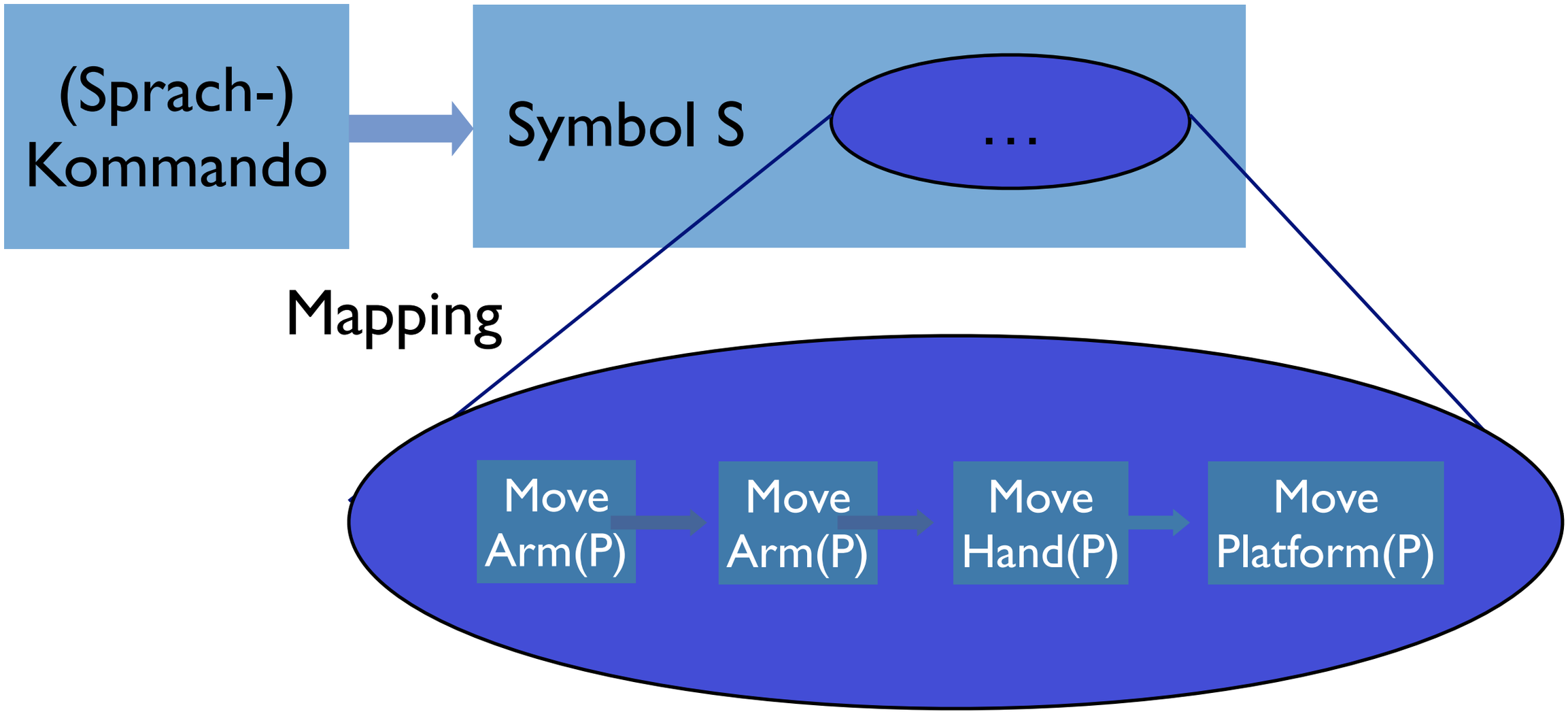
➡„DriveTo(...), SearchObject(...), MoveArm(...), Grasp(Becher), MoveArm(...),

DriveTo(...),

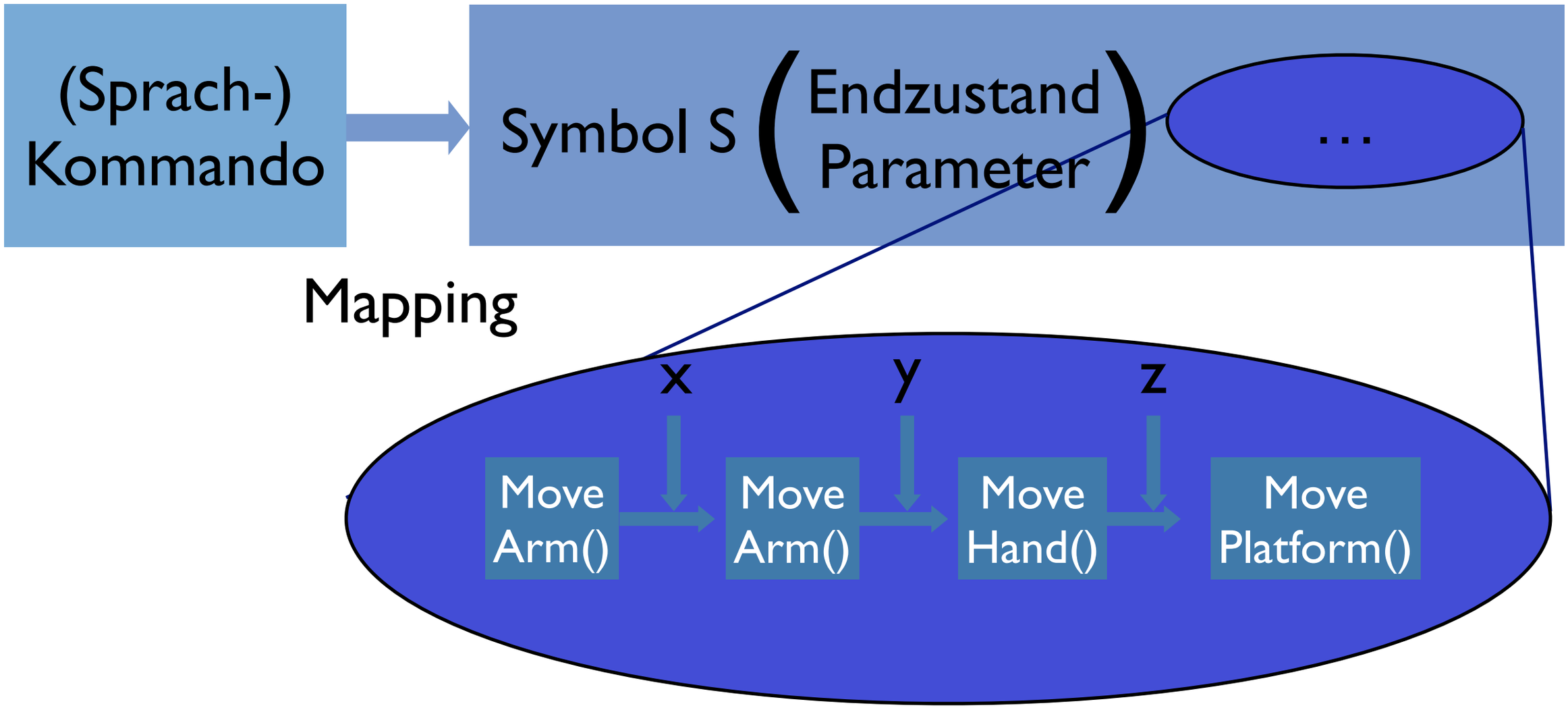
MoveArm(...) ...!“



Symbolische Abstraktion



Symbolische Abstraktion



Symbolische Handlungsatome

- In den gegebenen Beispielen wurden bereits implizit Handlungsatome angenommen

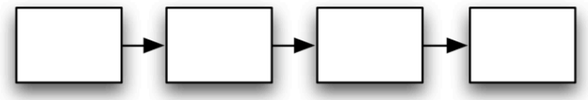
- Die kleinsten auszuführenden Handlungseinheiten werden als *Elementaroperationen* oder *atomare Handlungen* bezeichnet.

- Komplexe Handlungen werden aus Elementaroperationen zusammengesetzt. Diese können dazu üblicherweise parametriert werden.

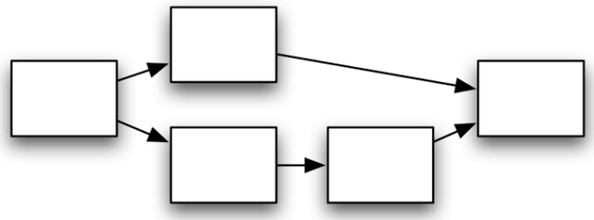
- Jeder Roboter besitzt eine endliche Menge an Elementaroperationen.

Ansätze für das Aufgabenmodell

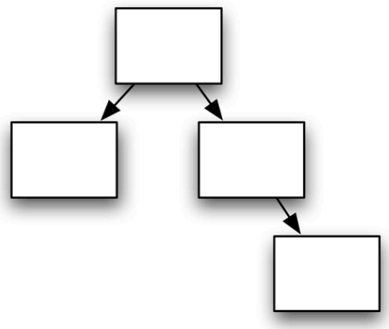
- Sequentiell
  - Festgelegte Folge von elementaren Aktionen



- Vorranggraph
  - Darstellung der Abhängigkeiten



- Hierarchisch
  - Abstraktion von Teilhandlungen



Anforderungen an das Aufgabenmodell

Erweiterbarkeit

Wiederverwendbarkeit

Erklärbarkeit

Integration des Wissens in ein Planungssystem

Sequentielle Handlungsbeschreibung

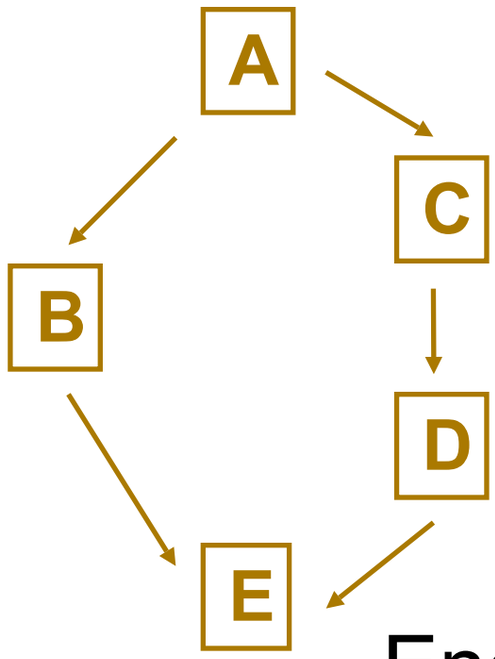
- Folge von (parametrierten) atomaren Handlungen
- Eindeutig festgelegte Reihenfolge
- Reihenfolge und Anordnung nicht unbedingt erklärbar (lesbar)
- Bedingungen, Alternativen, etc. schlecht darstellbar
- Sinnvoll für einfache Aufgaben oder sehr strukturierte Umgebungen (zB Leittechnik/Industrie)
- Komplexe Handlungen: Rein sequentielle Beschreibungen nicht mächtig genug!

Sequentielle Handlungsbeschreibung: Ausführung

- Ausführung sequentiell beschriebener Handlungen trivial
- Linearer Handlungsfluss
- Ausführung besteht aus:
  - Anstoßen einer Elementarhandlung
  - Evtl. Handlungsüberwachung
  - Nach (erfolgreicher) Beendigung: Übergang zur nächsten Elementarhandlung

- Einfache Mechanismen zur Ausführung nötig!

Modellierung der Reihenfolge: Vorranggraph



Symbol S $\begin{pmatrix} \text{Endzustand} \\ \text{Parameter} \end{pmatrix}$

Endzustand

vor „B“ muß „A“ ausgeführt sein

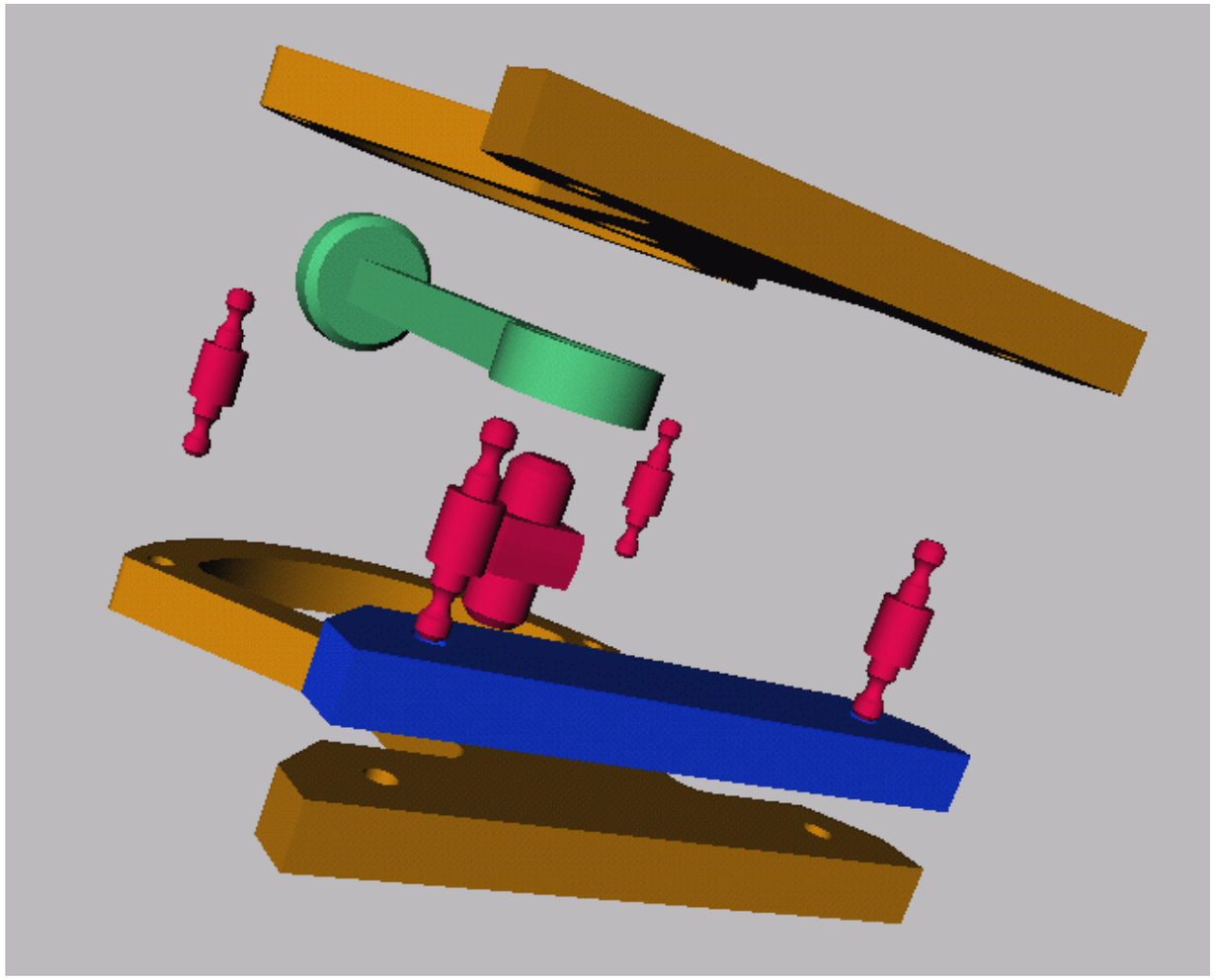
vor „C“ muß „A“ ausgeführt sein

vor „E“ müssen „B“ und „D“ ausgeführt sein

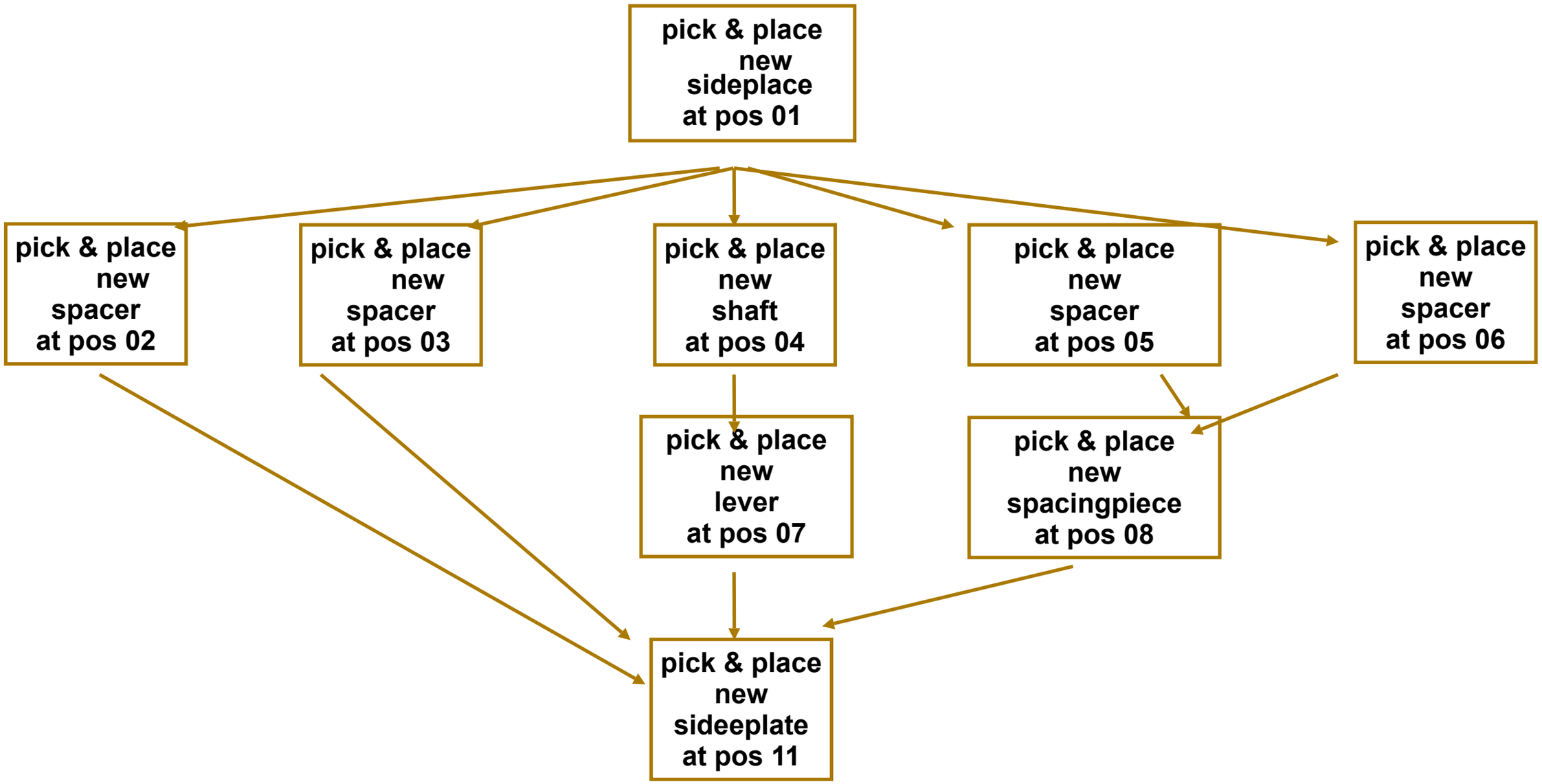
keine Aussage zwischen „B“ und „C“ bzw. „B“ und „D“

Vorranggraph: Beispiel

- Cranfield Benchmark:



Vorranggraph des Cranfield-Benchmarks



Vorranggraph: Ausführung

- Darstellung der Handlung in mehreren unabhängigen Teilzweigen
- Üblicherweise: Serialisierung notwendig!
  - Berechnung optimaler Operatorreihenfolgen
  - Freiheitsgrad zum Ausführungszeitpunkt
  - Serialisierung aufwendig (-> Planungsverfahren, siehe spätere Vorlesungen)
- Ausführung beinhaltet also:
  - Serialisierung der Handlung nach gegebenen Kriterien
  - Ausführung der sequentiell beschriebenen Handlung
- Mächtigere Handlungsbeschreibung benötigt auch mächtigere Verfahren bei der Ausführung!

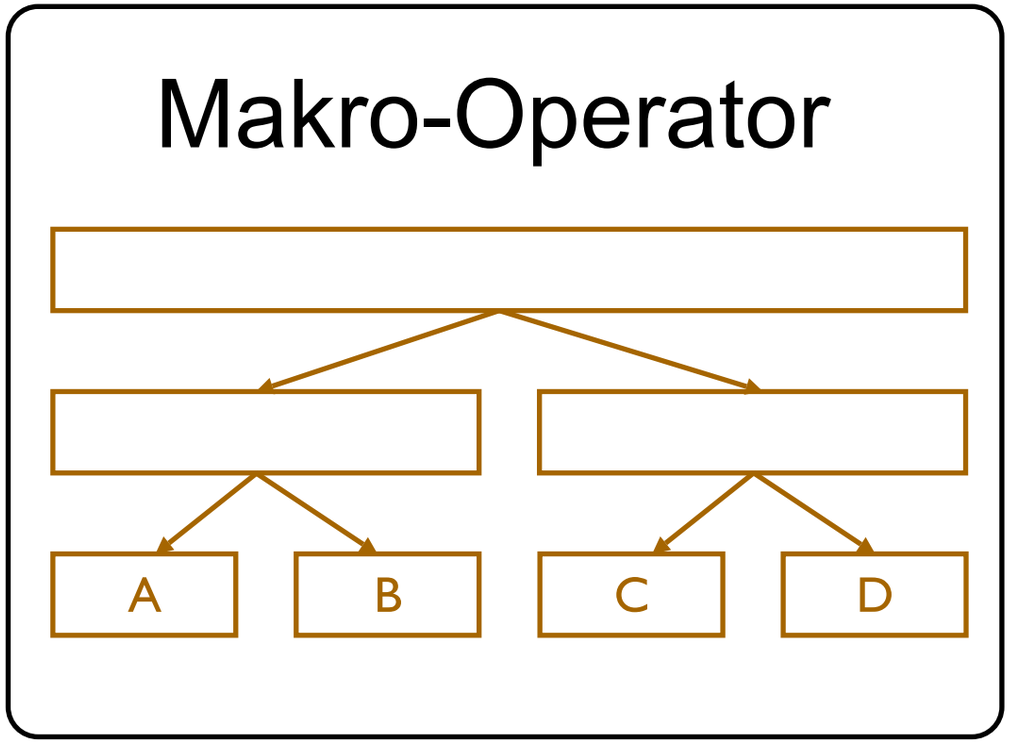
Hierarchisches Aufgabenmodell

Aufgabenspezifikation:

- Hierarchisch

Aufgabenzerlegung/-teilung:

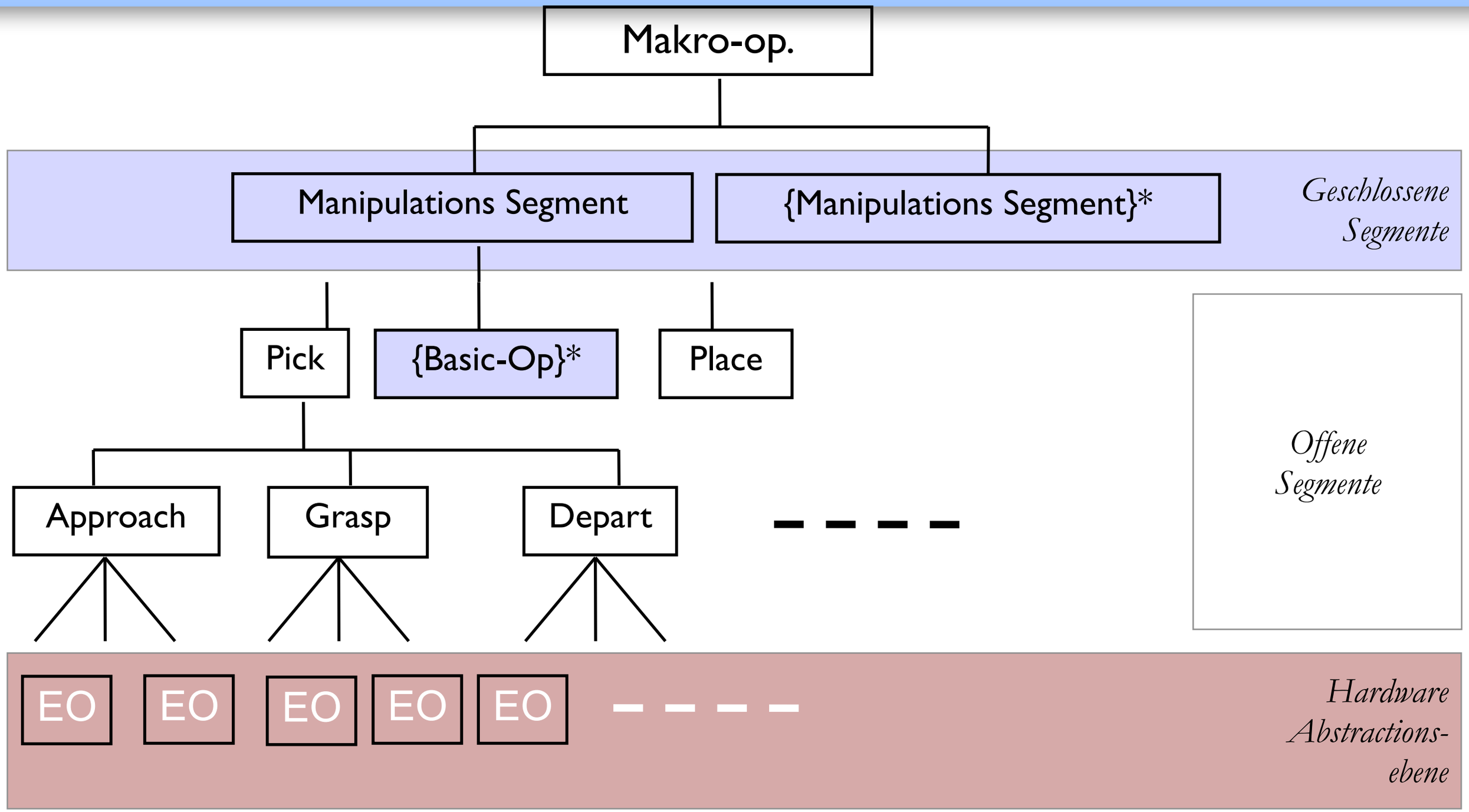
- Hierarchisch



Abstraktion nach:

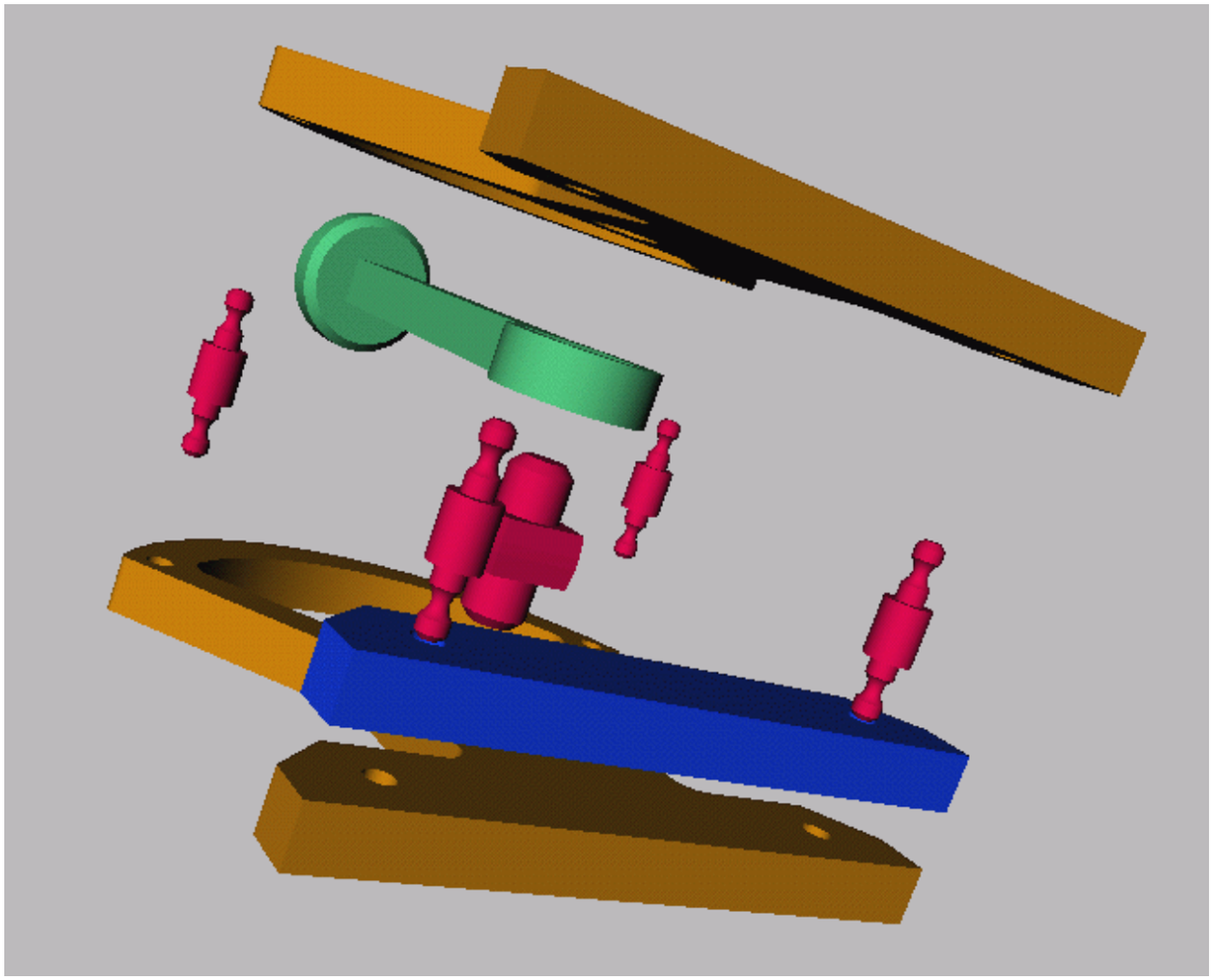
- Raum
- Objekten
- Zeit
- Alternativen

Semantische hierarchische Repräsentation



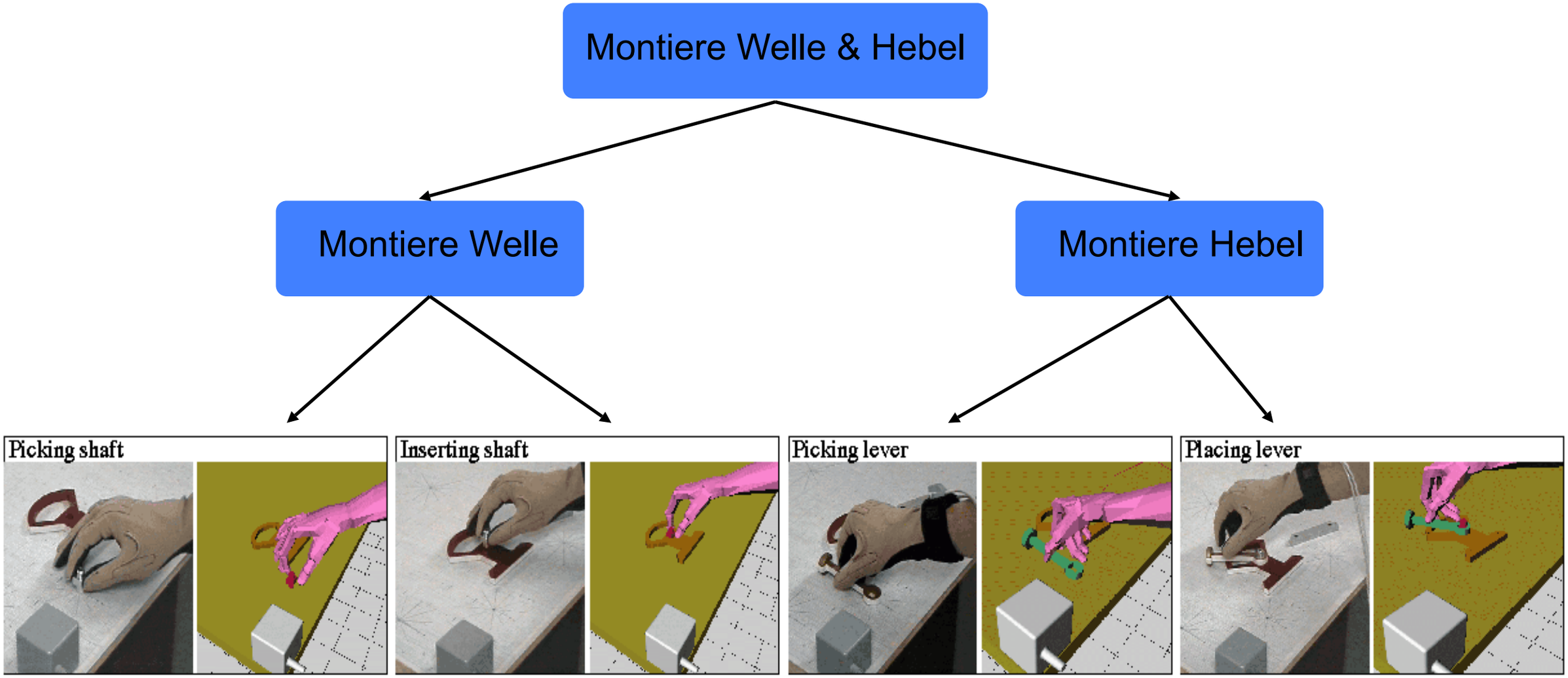
Beispiel

- Cranfield Benchmark:



Programmbeispiel

Montage von Welle und Hebel des Cranfield Benchmarks



Abstraktionsebenen

Semantische Abstraktionsstufen:

Stufe	Ebene	Beispiel
1	Mission	• Exchange • Install
2	Task	• Pick Up • Place
3	Aktion	• Move • Grasp

Diese Stufen benutzen symbolische Parameter
→ Komplexität der Aufgabe beherrschbar
→ Formalismen anwendbar

Exkurs: Wahl der Abstraktionsebenen

- Frage: Wie wird die Abstraktionsebene der *Aktion* gewählt?
  → Relevanz: Aktionen von Roboter A auf Roboter B übertragen!

  - Aktion: Kleinste symbolische Einheit
  - Darunter: Regelungsebene
  - Abhängig von:
    - Komplexität der Aufgabe
    - Grad der Kopplung einzelner Teilhandlungen
    - Hardware: Getrennt geregelte Komponenten werden üblicherweise auch mit getrennten Aktionen modelliert
    - Planungssystem/Beobachtungssystem
- Grundsätzlich: Die Frage ist in der Forschung noch nicht eindeutig geklärt, immer noch ein Streitpunkt!

Modellierung von Aktionen / Tasks

Definition Operator:

OP = (N, O, A, Par, K)

mit:

N = Name des Operators (eindeutig)

O = Liste der beteiligten Objekte

A = Auswahlbedingung

Par = Liste der Parameter, z.B. Positionen, Kräfte,
Beschleunigungen ...

K = Körper des Operators

Modellierung von Aktionen / Tasks

Körper eines Operators:

- ausführbares Programm

=>Realisierung einfacher Fähigkeiten

Aktion

Elementaroperator

- besteht aus weiteren Operatoren

$K = (K_1 K_2 \ldots K_n)$

=> Realisierung komplexer Aktionen

Task

Makro-Operator

Hierarchische Repräsentation: Ausführung

- Im Aufgabenmodell ist die Serialisierung implizit gegeben
  - Minimaler Aufwand zur Ausführung
  - Aber: Bei Parallelitäten von Teilhandlungen: Konflikte möglich!
- Ausführung beinhaltet also:
  - Überprüfung auf Konflikte
  - Gegebenenfalls Serialisierung/Lösen der Konflikte
  - Ausführung der resultierenden Operatorsequenz

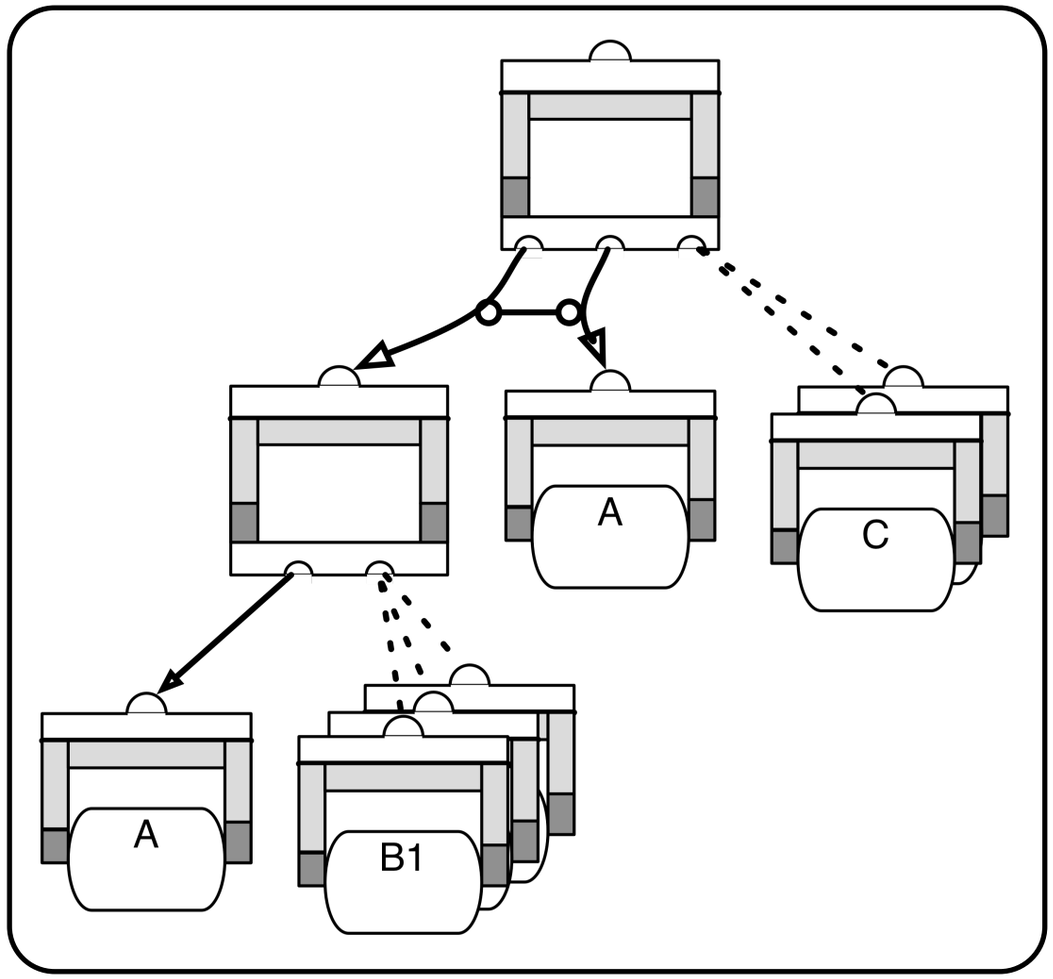
- Vorteil: Zusammengesetzte (Teil-)Programme können wiederverwendet werden
- Repräsentation gut „lesbar“ (verständlich, erklärbar)

Hierarchische Handlungsbeschreibung: Anwendung *in realitas*

- Symbolische Abstraktionsebene zur Roboterprogrammierung: Flexible Programme (FPs)
- FPs:
  - Hierarchische Handlungsbeschreibung
  - Erklärbar, verständlich, intuitiv
  - Wiederverwendbarkeit
  - Parametrierbar
  - Unterstützt: Bedingungen, Verzweigungen, Ressourcenverwaltung, Parallelität

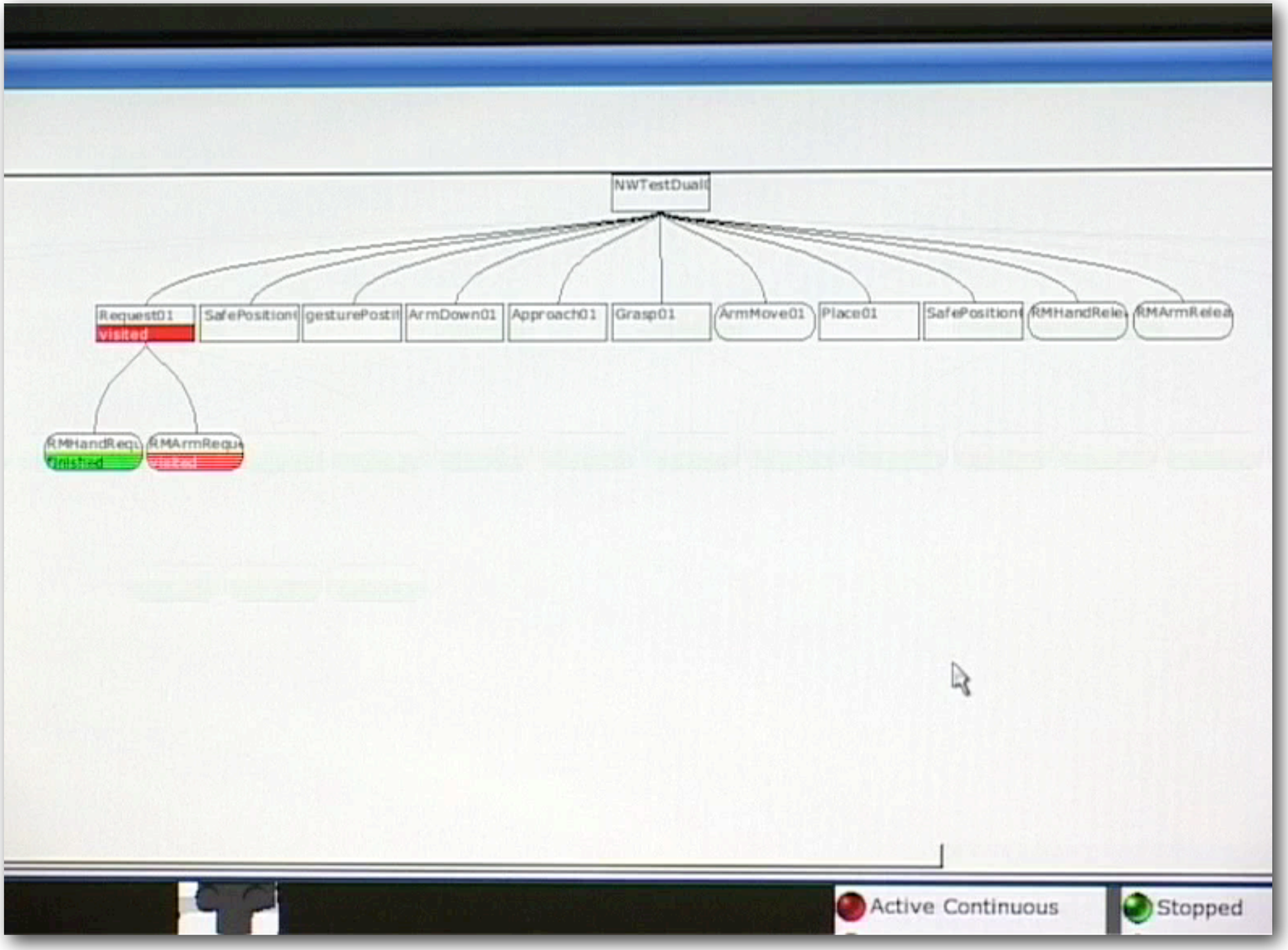
FPs: Aufbau flexibler Programme

- Repräsentation des Handlungs-wissens als Baumstruktur
- Parametrierbare Aktionsbeschreibung
- Blätter entsprechen Roboteraktionen
- Abarbeitung entsprechend einer Tiefensuche
- Instantiierung der Kinder zur Laufzeit (Expansion des Baums)
- Auswahl des geeignetsten Kandidaten beim expandieren
- Parallele Ausführung mehrerer Kinder möglich



Beispiel: FP Ausführung

Beispiel: FP Ausführung



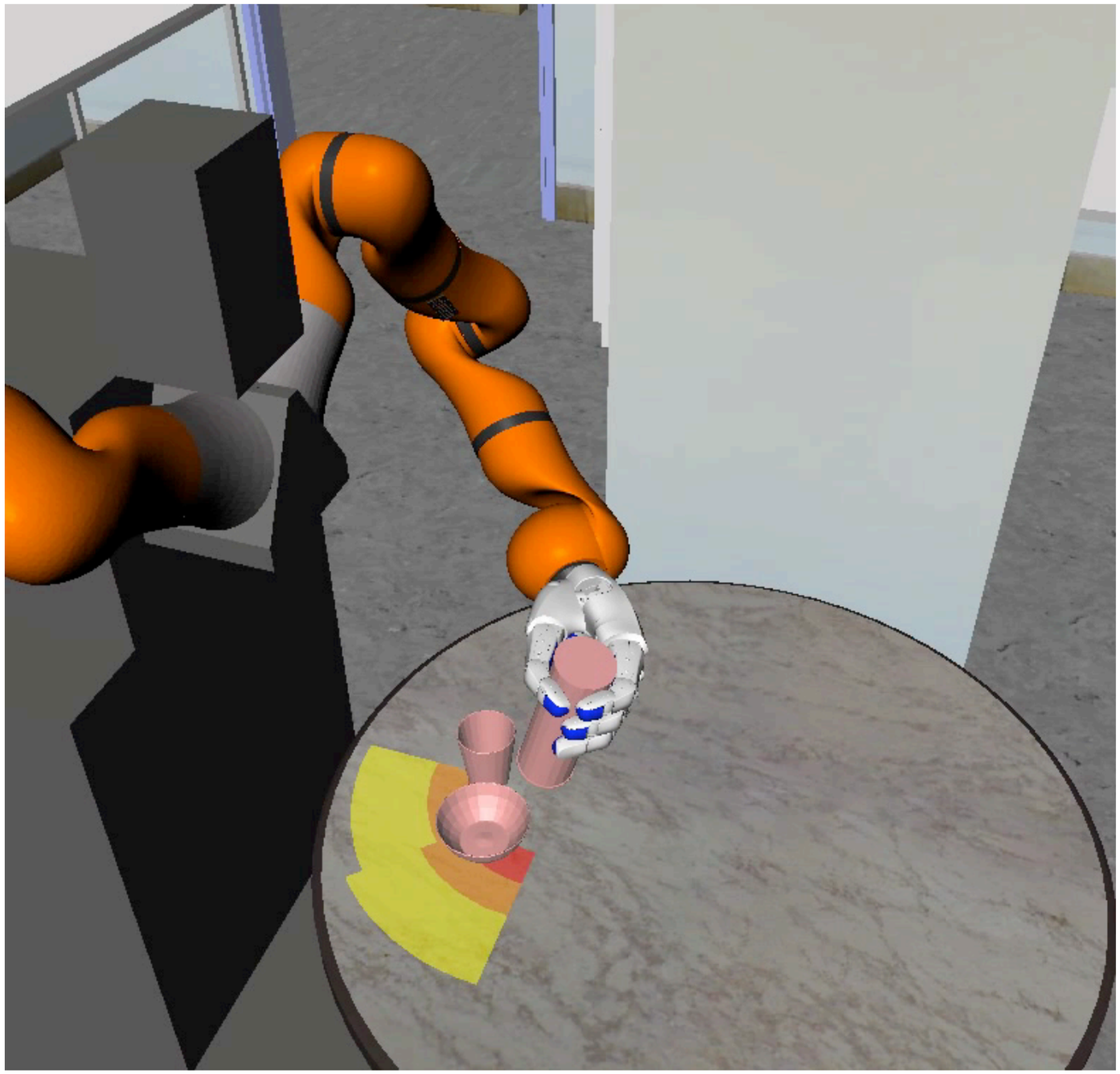
Validierung der Modelle

- Validierung
  1. Simulation der Komponenten: Validierung anhand gegebener Einschränkungen (Kollisionen, Erreichbarkeit, Optimalitätskriterien: Weg, Zeit, Energie, …)
  2. Graphische Animation: Der Anwender überprüft visuell die erstellten Modelle

Simulation: Manipulator

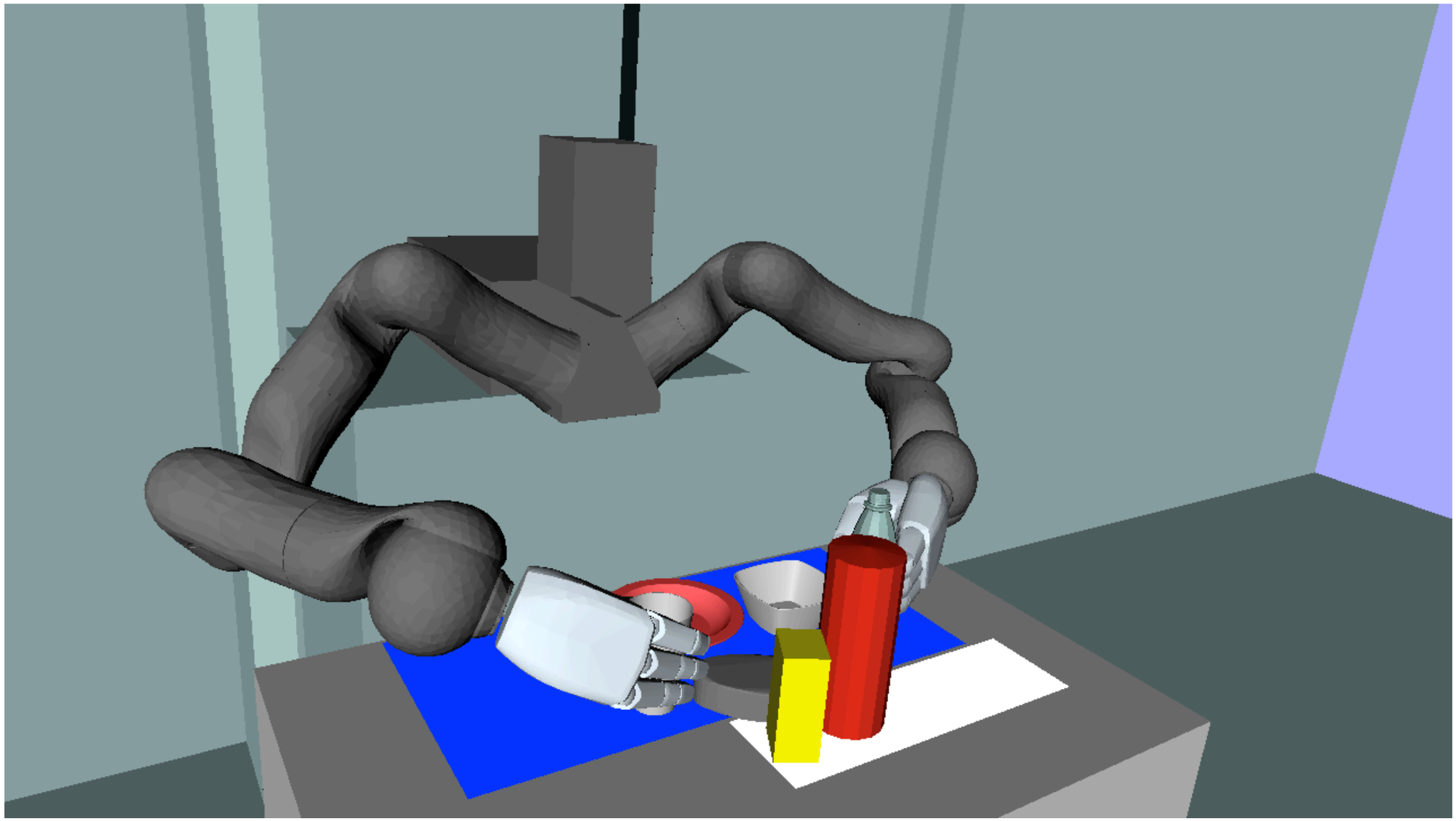
Für die Simulation von Effekten durch Manipulators wird die Physiksimulation verwendet.

- Masse
- Reibung
- Kräfte
- Gelenke



Graphische Animation

Wenn die Robotersimulationen ergeben, daß die Zielstellung angefahren werden kann, wird die Bewegung in einer Animation graphisch dargestellt.



Zusammenfassung Aufgabenmodell

- Aufgabenmodell basiert auf elementaren Operationen
- Oft dreischichtiger Ansatz: Aktion, Task, Mission
- Verknüpfung der elementaren Operationen zu komplexen Aufgaben
- Verknüpfung:
  - Sequentiell
  - Vorranggraph
  - Hierarchisch
  - (Kontrollstrukturen)
- Problem: Validierung der Programme
  - Simulation
  - Animation und Validierung durch den Menschen

Diskussion (1)

- Mobile Plattform ohne Sensorik (zB Roomba)
  - Aufgabenmodell?
  - Umweltmodell?
- Mobile Plattform mit Differentialantrieb und Sensorik zur Lokalisation (zB Transportaufgaben)
  - Aufgabenmodell?
  - Umweltmodell?
- Serviceroboter mit mobiler Plattform, Manipulator, Mehrfingergreifer, komplexe Sensorik
  - Aufgabenmodell?
  - Umweltmodell?

Diskussion (2)

- Mobile Plattform ohne Sensorik (zB Roomba)
- Mobile Plattform mit Differentialantrieb und Sensorik zur Lokalisation
- Serviceroboter mit mob. Plattform, Manipulator, Mehrfingergreifer, komplexe Sensorik

- Wo kommt jeweils das Aufgabenwissen dieser Systeme her?