

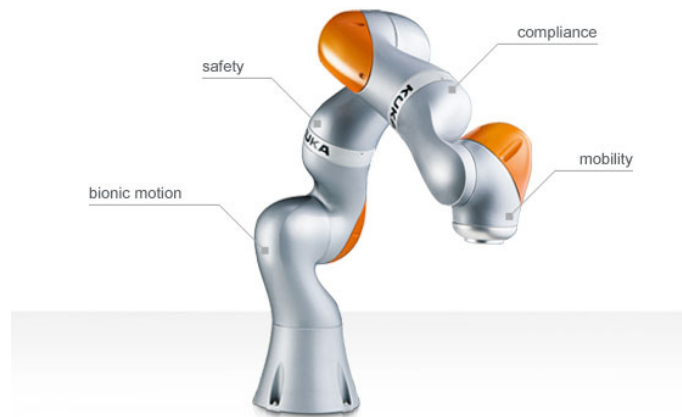


HS Augsburg
Fak. Informatik



CIM & Robotik
Prof. G. Stark

Modellbasierte Programmierung einer Simulationskomponente für die KUKA-Robotersteuerung *Sunrise*



Hochschule Augsburg, Labor für CIM & Robotik, Prof. Dipl.-Ing. Georg Stark
Email: Georg.Stark@hs-augsburg.de



Übersicht



1. Hohe Softwarequalität durch modellbasierte Programmierung
2. Intelligente und mobile Robotersysteme
3. Simulationskomponente **SunSim** für die Steuerung **Sunrise** von KUKA Roboter
4. Schnittstelle für Realistische Robotersimulation
5. Anwendung an der Hochschule Augsburg



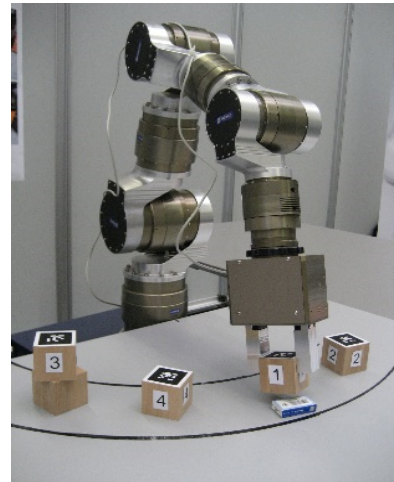
HS Augsburg
Fak. Informatik

Labor CIM & Robotik 1

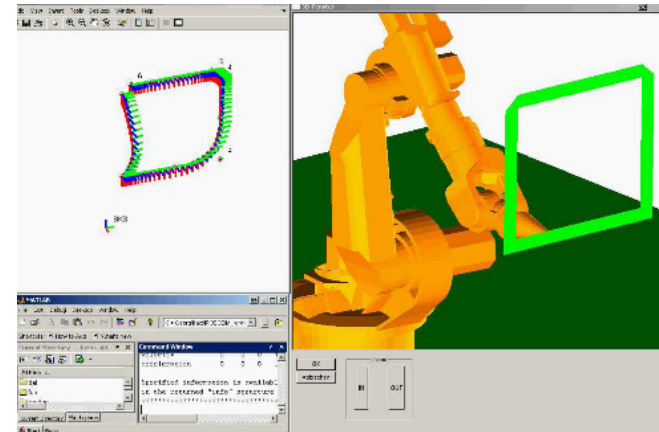


CIM & Robotik
Prof. G. Stark

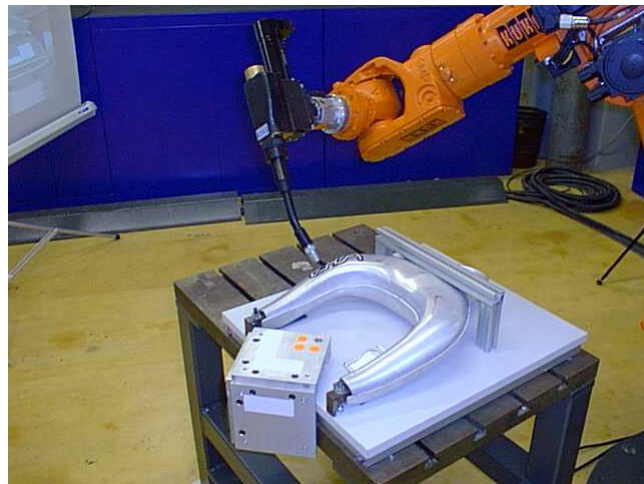
Eigene Steuerung
MRobot



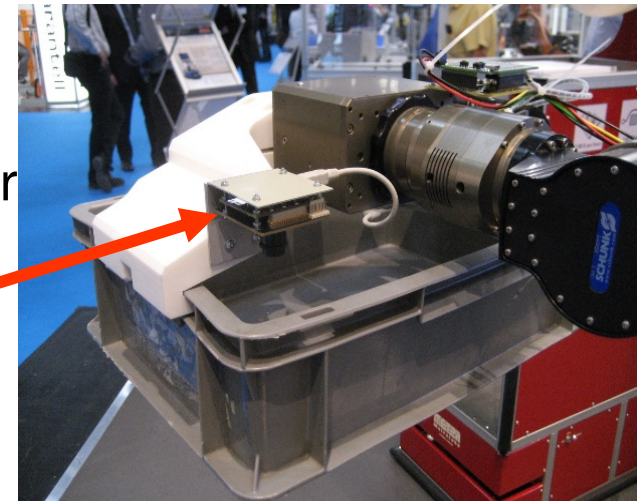
Simulation



Prog. mit 3D-Bilddatendaten



EU-Projekt
Mobile Roboter
Greiferkamera





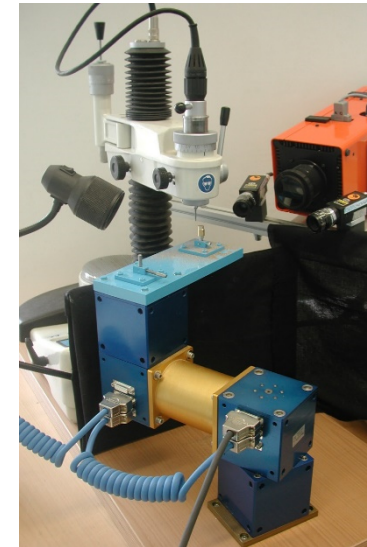
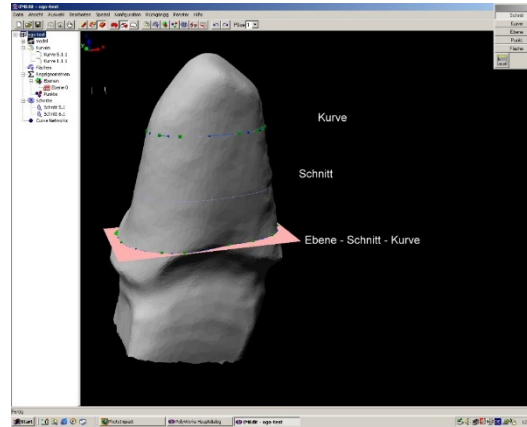
HS Augsburg
Fak. Informatik

Labor CIM & Robotik 2

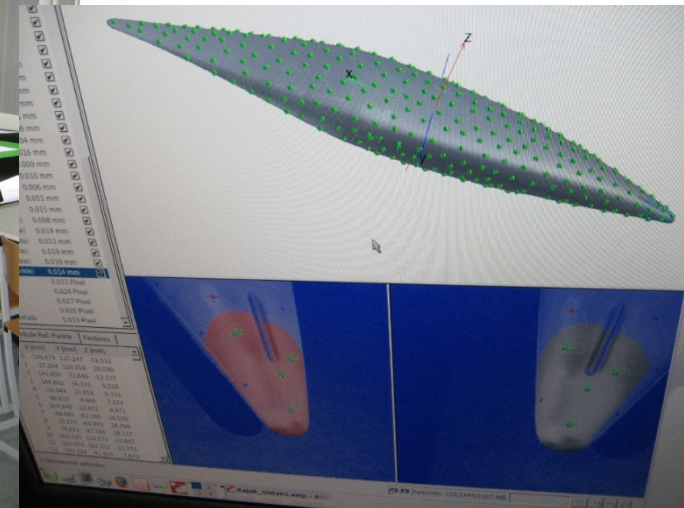


CIM & Robotik
Prof. G. Stark

Fräsen
Zahnkronen

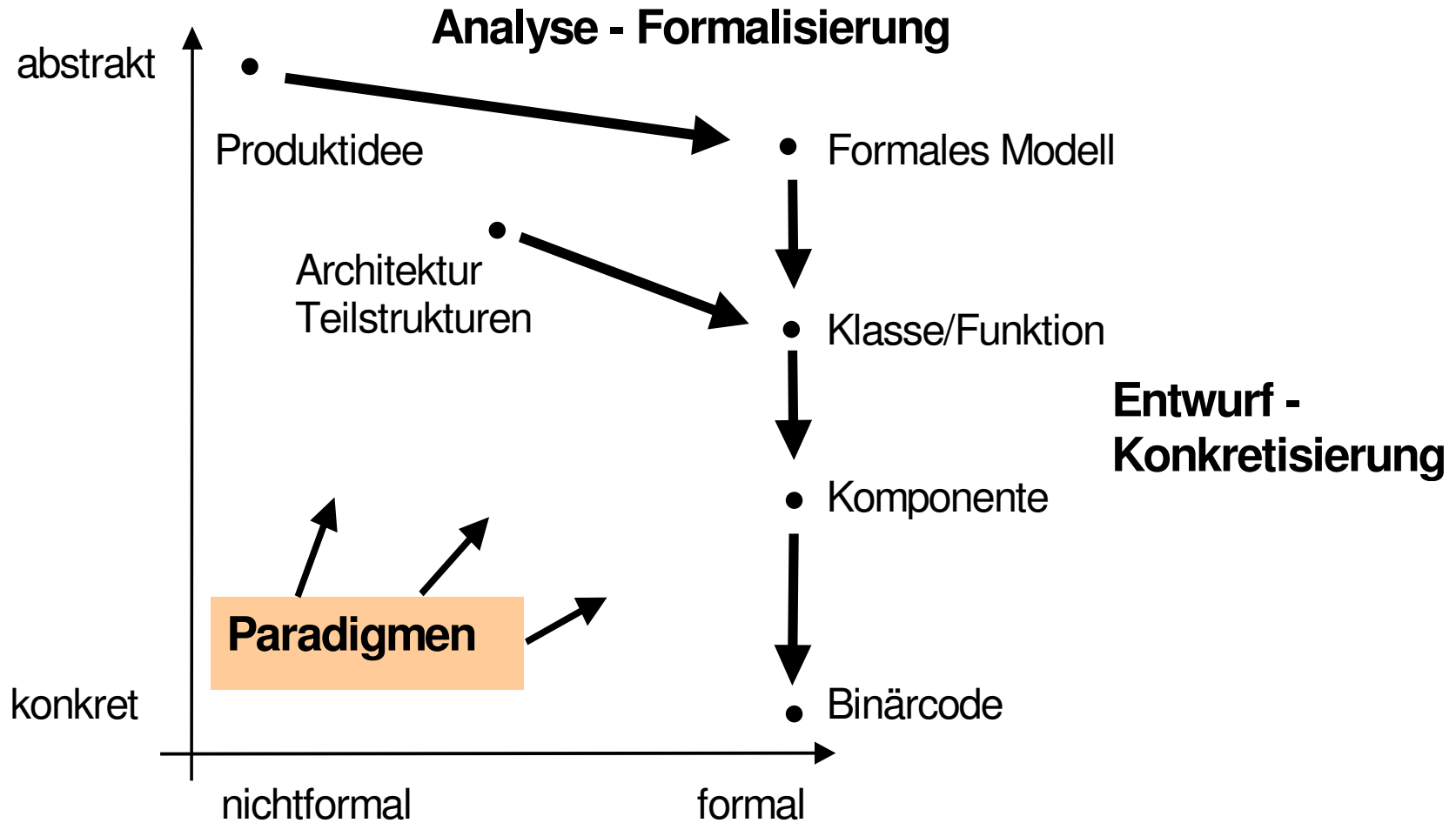


Kajak
Olympia-
Sieger
A. Grimm



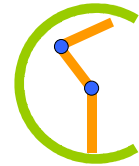


Darstellung des Programmwissens in der Entwurfsebene





Vergleich der Paradigmen



Paradigma	Primäre Darstellung	Wissensdomäne	Abstraktions-ebene
Imperativ	prozedural	Prozess (parallel, sequentiell)	niedrig
Objektorientiert	deklarativ	Struktur	mittel
Komponenten-orientiert	deklarativ	Struktur	mittel
Datenfluss-orientiert	deklarativ	Inhalt	hoch
Modellbasiert	deklarativ	Inhalt	hoch



Modellbasierte Softwareentwicklung

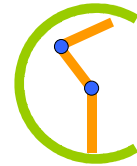


- Formale, abstrakte Darstellung der Wissensinhalte
 - keine Realisierungsdetails
 - weitgehende Plattformunabhängigkeit
- Unterschiedliche Modellbegriffe
 - Physikalische Systeme / Prozesse
 - Datenstrukturen
- Domänenspezifische Sprache – Text/Grafik
 - einfach und prägnant
 - Benutzung ohne umfangreiche Programmierkenntnisse
- Simulation und Codegenerierung

**Was ist die
Besonderheit ?**



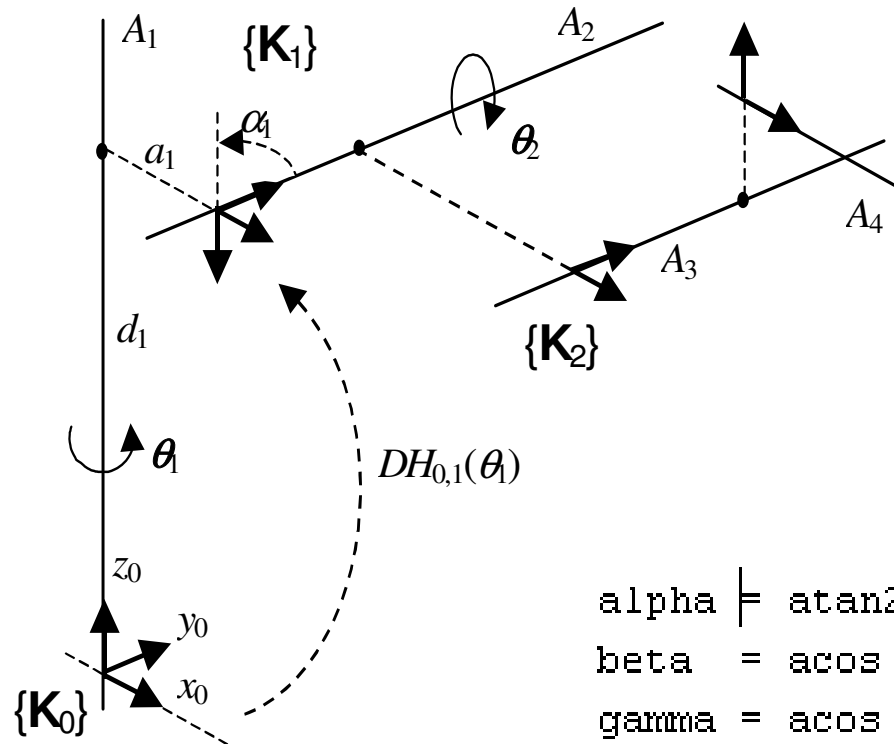
MATLAB-Code: Planung Kreisinterpolation



```
File Edit Text Desktop Window Help
[Icons]
1 function [s1 KR rad]=kreis_plan(P1, P2, P3)
2
3 global q_akt rob_para ipo_takt
4
5 u1=P1(1:3,4); u2=P2(1:3,4); u3=P3(1:3,4);
6 zk=cross(u3-u1,u2-u1); zk=zk/norm(zk);
7 hv1=(u1+u2)/2; rv1=cross(u2-u1, zk);
8 hv2=(u1+u3)/2; rv2=cross(u3-u1, zk);
9
10 A=[rv1 -rv2]; k=hv2-hv1;
11 l=A\k;
12 m1=hv1+l(1)*rv1;
13 m2=hv2+l(2)*rv2;
14 m=(m1+m2)/2; % Kreismittelpunkt
15 rad=norm(u1-m); % Radius
16
17 xk=u1-m; xk=xk/norm(xk);
18 yk=cross(zk,xk);
19 KR=[xk yk zk m; 0 0 0 1]; % Lokales Koordinatensystem
20 P2_KR=inv(KR)*P2;
21 phi=atan2(P2_KR(2,4), P2_KR(1,4));
22 if phi<0 phi=2*pi+phi; end % Winkelbereich: 0<phi<2*pi
23 s1=phi*rad; % Bogenlänge Kreissegment
```




Beispiel: Geometriemodell - Kinematik

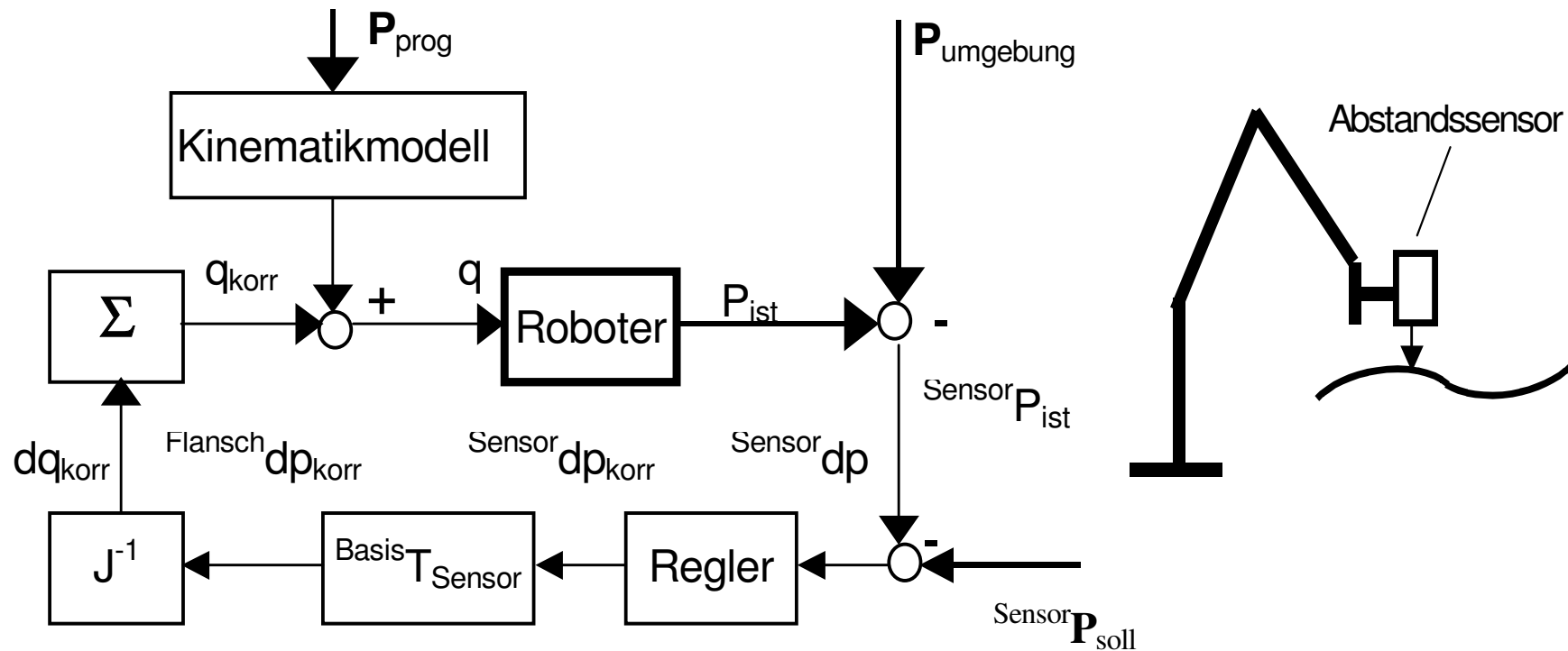
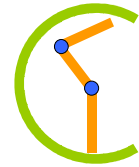


$$dhp = \begin{bmatrix} 0 & 0.4 & 0 & -\pi/2; \\ 0 & 0 & 0.2 & 0; \\ -\pi/2 & 0 & 0 & -\pi/2; \\ 0 & 0.3 & 0 & -\pi/2; \\ 0 & 0 & 0 & \pi/2; \\ 0 & 0.1 & 0 & 0 \end{bmatrix};$$

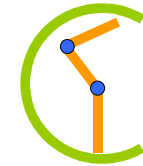
```
alpha = atan2 (hy1, hx1);
beta = acos( (oa^2 - ua^2 + l^2) / (2 * l * oa) );
gamma = acos( (oa^2 + ua^2 - l^2) / (2 * oa * ua) );
```



Beispiel: Signalflussmodell - Regelung



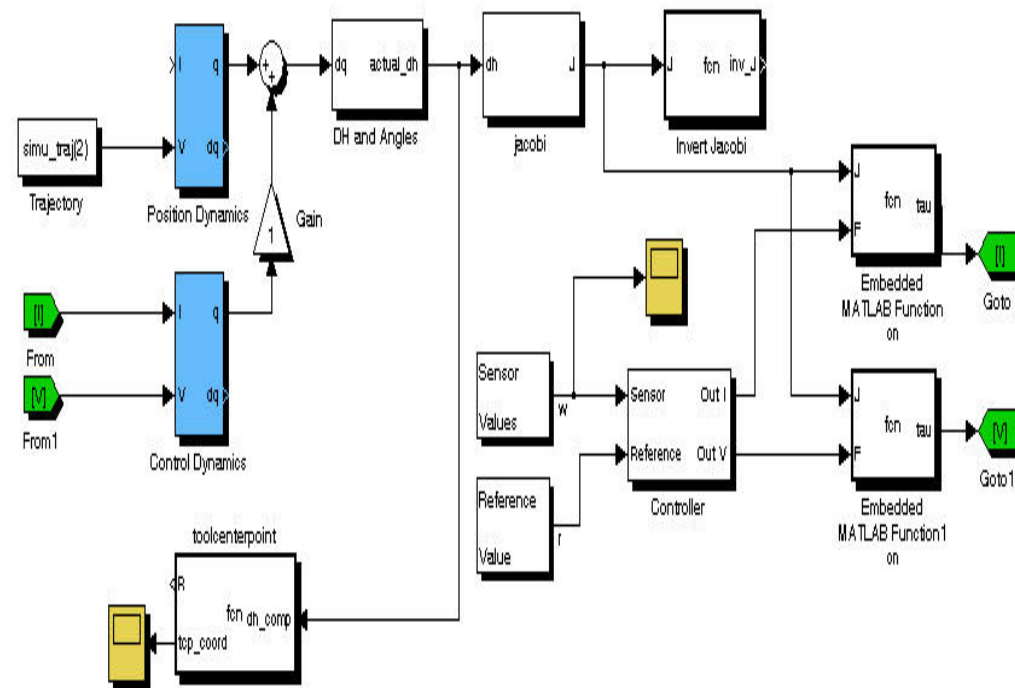
P: kartesische Werte, q: achsbezogene Werte



Matlab - Text

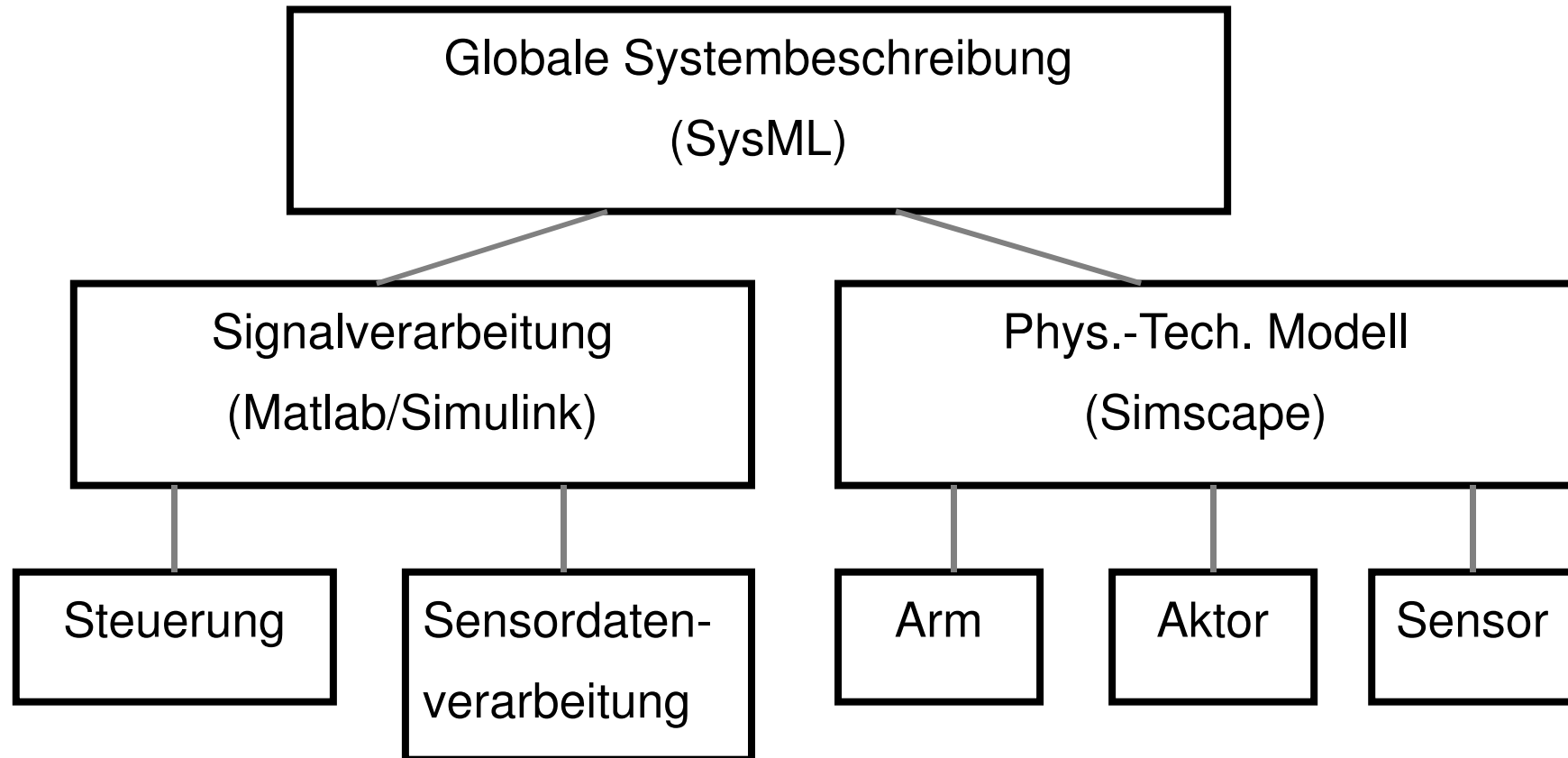
```
% Transformation P2 nach KR  
xk=u1-mk;   xk=xk/norm(xk);  
yk=cross(zk, xk);  
KR=[xk yk zk mk; 0 0 0 1];  
P2_KR=inv(KR) * P2;  
P2_KR(1,4));  
phi=atan2(P2_KR(2,4),0
```

Simulink - Grafik





Modellbasierte Systementwicklung

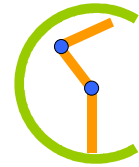


SysML: standardisierte Modellierungssprache für Systeme



HS Augsburg
Fak. Informatik

Intelligente Robotersysteme in der Fertigung

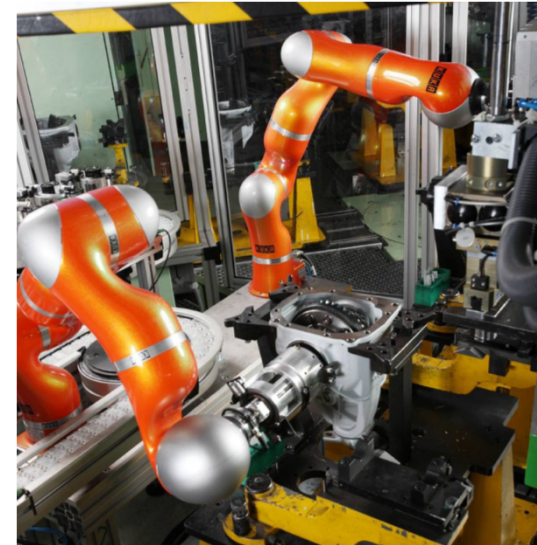


CIM & Robotik
Prof. G. Stark

Getriebemontage



Roboter als mobiler, flexibler
Assistent (flexFELLOW)

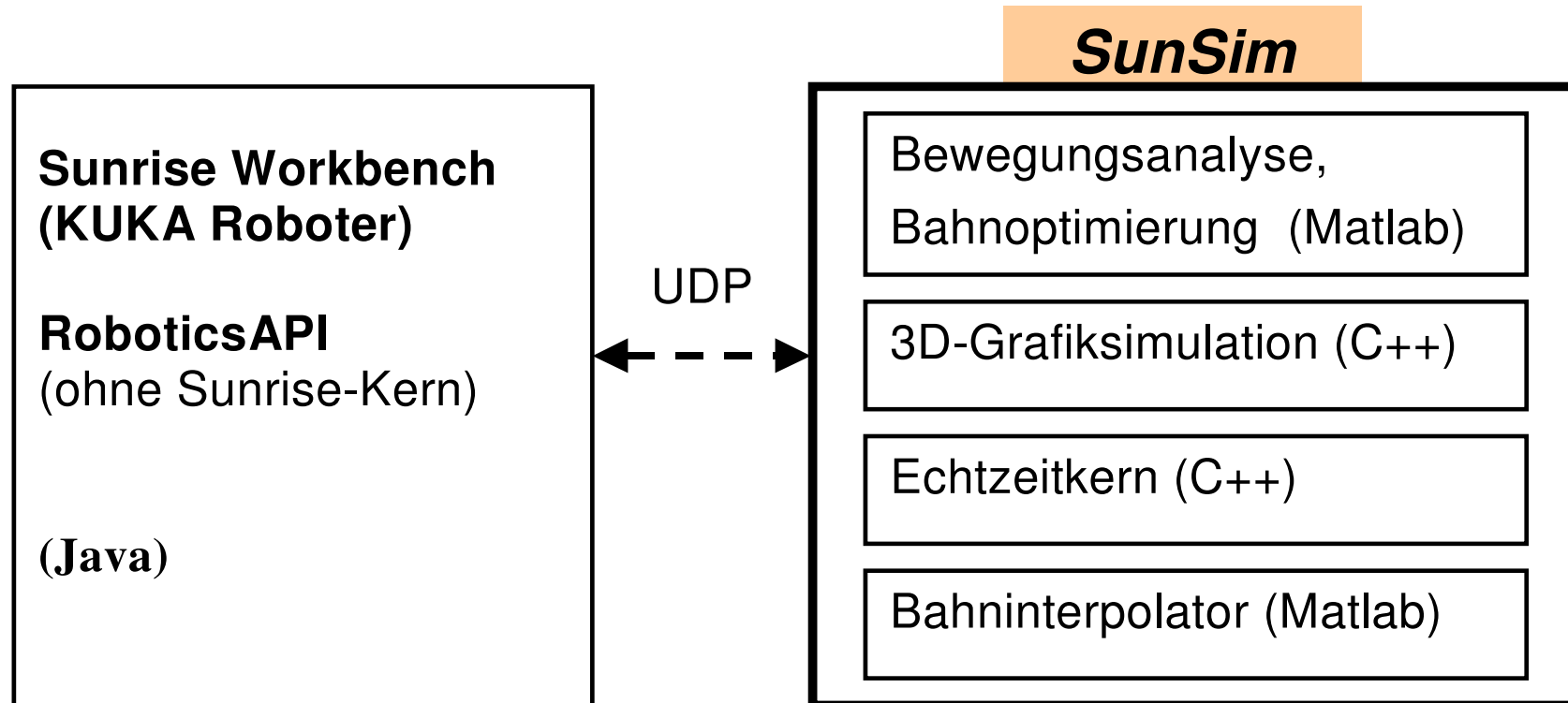


Mobile Plattform





Struktur der Simulationskomponente *SunSim*



UDP: einfaches, weitverbreitetes Netzwerkprotokoll



Bewegungssimulation - Verfahrenarten



Der *Sunrise*-Kern wird emuliert.
Dies ergibt eine stabile,
wartungsfreundliche Software
mit ausreichender Genauigkeit.

**Realisierungsprinzip
Sunrise-Kern?**

Intelligenter Roboterarm

- PTP
- LIN, mit Überschleifen
- CIRC
- SPLINE

Omnidirektionale Plattform

- PTP (entspricht LIN)
- Relativbewegung
- Globale Bewegung mit Zielmarke
(entspricht Navigation)



Bedienoberfläche



Sunrise Workbench (KUKA)

SunSim

```
public class RobotApplication extends RoboticsAPIApplication {
    private Controller kuka_Sunrise_Cabinet_1;
    private LBR_lbr_iiwa_7_R800_1;
    private MRobot mRobot;

    public void initialize() {
        Iterator<Controller> it = this.getContext().getControllers()
        Controller controller = it.next();

        // Set Controller, Name, IP-Address of Server, Network Port
        this.mRobot = new MRobot(controller, "LBR1", "141.82.165.12"
    }

    public void transport() {

        this.mRobot.move(platptp(getApplicationData().getFrame("/_PL2"))

        //
        this.mRobot.move(ptp(getApplicationData().getFrame("/R1")).setJc
        this.mRobot.move(lin(getApplicationData().getFrame("/R2")).setJc
        this.mRobot.move(lin(getApplicationData().getFrame("/R1")).setJc
        this.mRobot.move(ptp(getApplicationData().getFrame("/RHome")).se

        this.mRobot.move(platptp(getApplicationData().getFrame("/_PL3"))
        this.mRobot.move(platptp(getApplicationData().getFrame("/_PL4"))

        this.mRobot.move(ptp(getApplicationData().getFrame("/R3")).setJc
        this.mRobot.move(lin(getApplicationData().getFrame("/R4")).setJc
        this.mRobot.move(lin(getApplicationData().getFrame("/R3")).setJc
        this.mRobot.move(ptp(getApplicationData().getFrame("/RHome")).se

        this.mRobot.move(platptp(getApplicationData().getFrame("/_PL1"))
    };
```

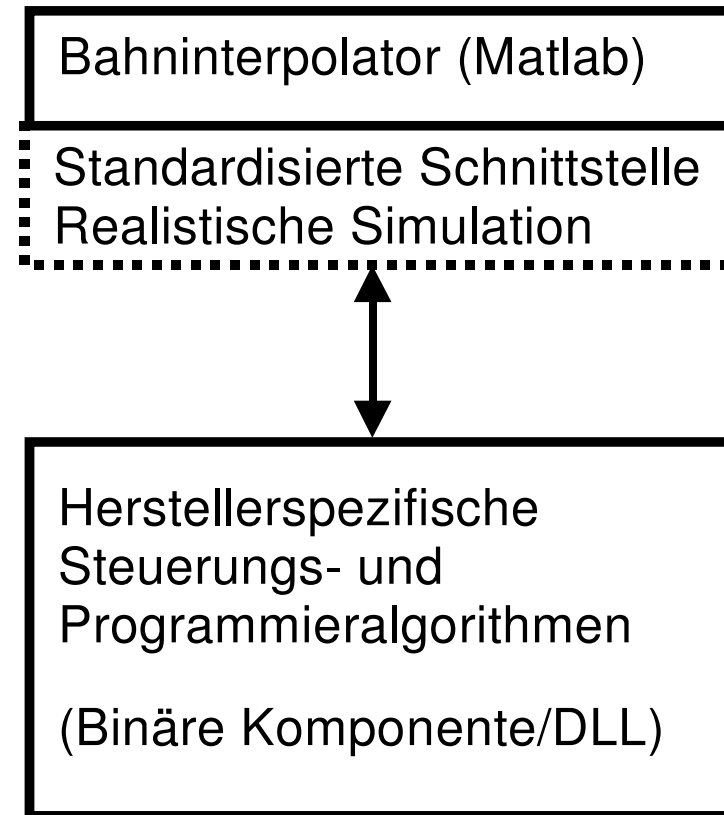
The interface also includes a 3D window showing a yellow KUKA robot arm in a virtual environment with a blue floor and green walls. There are two purple cubes on the floor. A tree view on the right shows the robot's structure: MRobot, GraspData, World, and various joints like _PL1, _PL2, _PL3, _PL4, _ZZ, R1, R2, R3, R4, RHome, and XXX. At the bottom, there is a control panel for 'Analysieren und Teachen' with buttons for 'Unterbrechen', 'Rückwärts', 'Wiederholung', 'Schrittweise neu abfahren', 'Schritte' (set to 5), 'Start', 'Vorwärts', 'Analyse', 'Achse' (set to 1), 'Kartesisch', 'Teachen', 'Handverfahr', 'Achse', 'Schleppabstand', 'TCP', and 'Picken'.



Schnittstelle Realistische Robotersimulation



- Darstellung der herstellere-spezifischen Verfahren in Matlab-Skript oder C/C++
- Integration in *SunSim* als binäre Softwarekomponente (DLL)

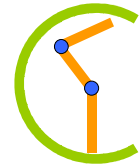




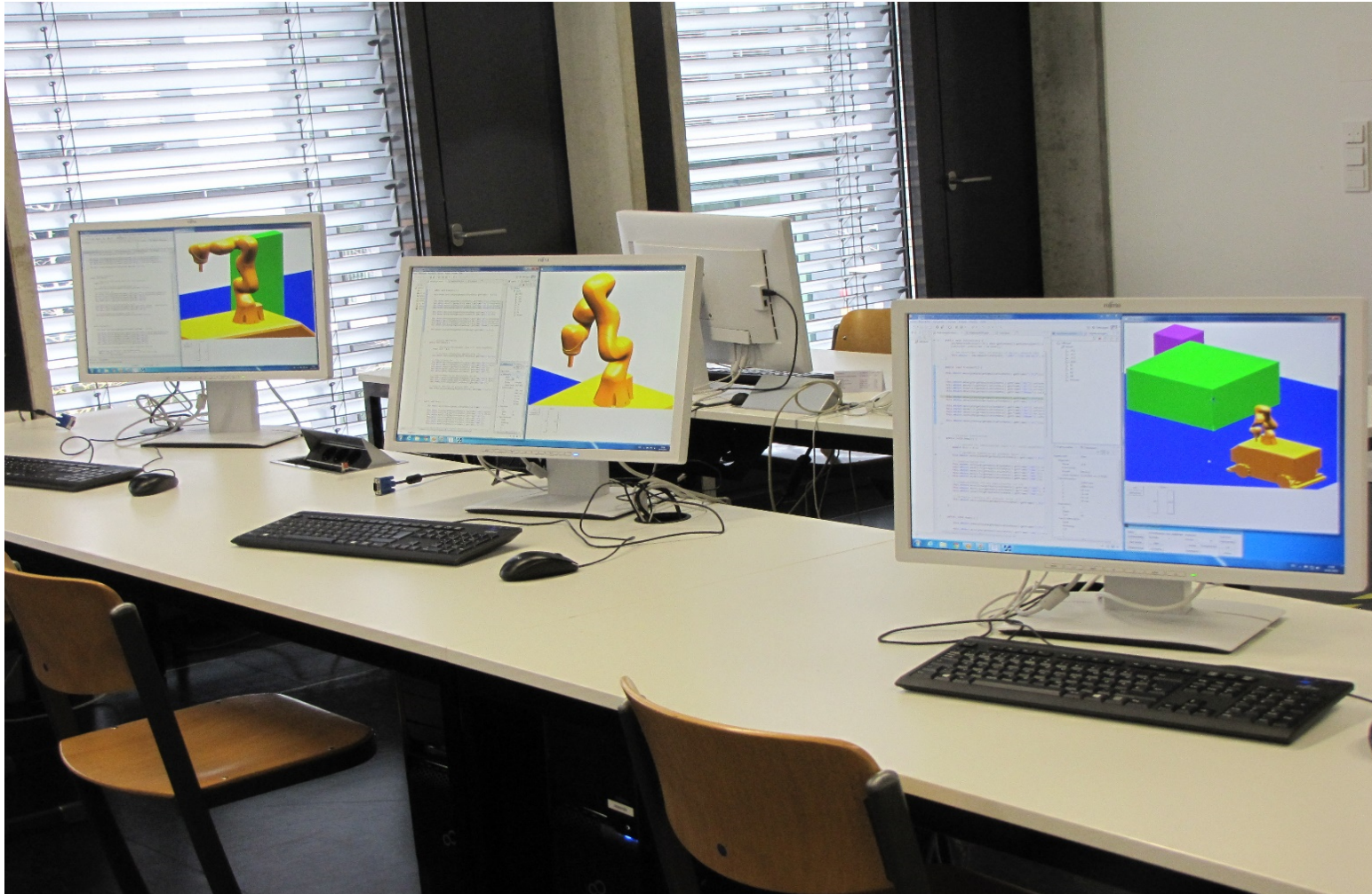
HS Augsburg
Fak. Informatik

Anwendung an der Hochschule Augsburg

Praktikum Robotersimulation



CIM & Robotik
Prof. G. Stark





Zusammenfassung



- Erweiterbare Simulationssoftware **SunSim** für die neue KUKA-Steuerung *Sunrise*
- Geschützte Integration herstellerspezifischer Algorithmen
- Modellbasierte Programmierung
 - Implementierung mit MATLAB
 - Hohe Softwarequalität bei reduzierten Kosten
- Programmierung und Optimierung von mobilen Robotersystemen für die flexible Fertigung.



HS Augsburg
Fak. Informatik

Modellbasierte Prog: Robotik mit Matlab



CIM & Robotik
Prof. G. Stark



Lehrbuch

Programmierung von
Robotersteuerungen

Geeignet für

- Technikerschulen
- Oberstufe Gymnasien
- Hochschulen
- Universitäten



HS Augsburg
Fak. Informatik



CIM & Robotik
Prof. G. Stark

**Vielen Dank für ihre
Aufmerksamkeit!**

