



HS Augsburg  
Fak. Informatik

# Modellbasierte und komponentenorientierte Softwareentwicklung für Eingebettete Systeme



CIM & Robotik  
Prof. G. Stark

1. Vorstellung Labor für CIM & Robotik
2. Zielkriterien für die Softwareentwicklung
3. Programmierparadigmen und Entwicklungsprozess
4. Modellbasierte Entwicklung
5. Komponentenorientierte Entwicklung
6. Anwendungsbeispiele
7. Ausblick auf neue Verfahren und Technologien

Hochschule Augsburg, Labor für CIM & Robotik, Prof. Georg Stark  
Email: [Georg.Stark@hs-augsburg.de](mailto:Georg.Stark@hs-augsburg.de)

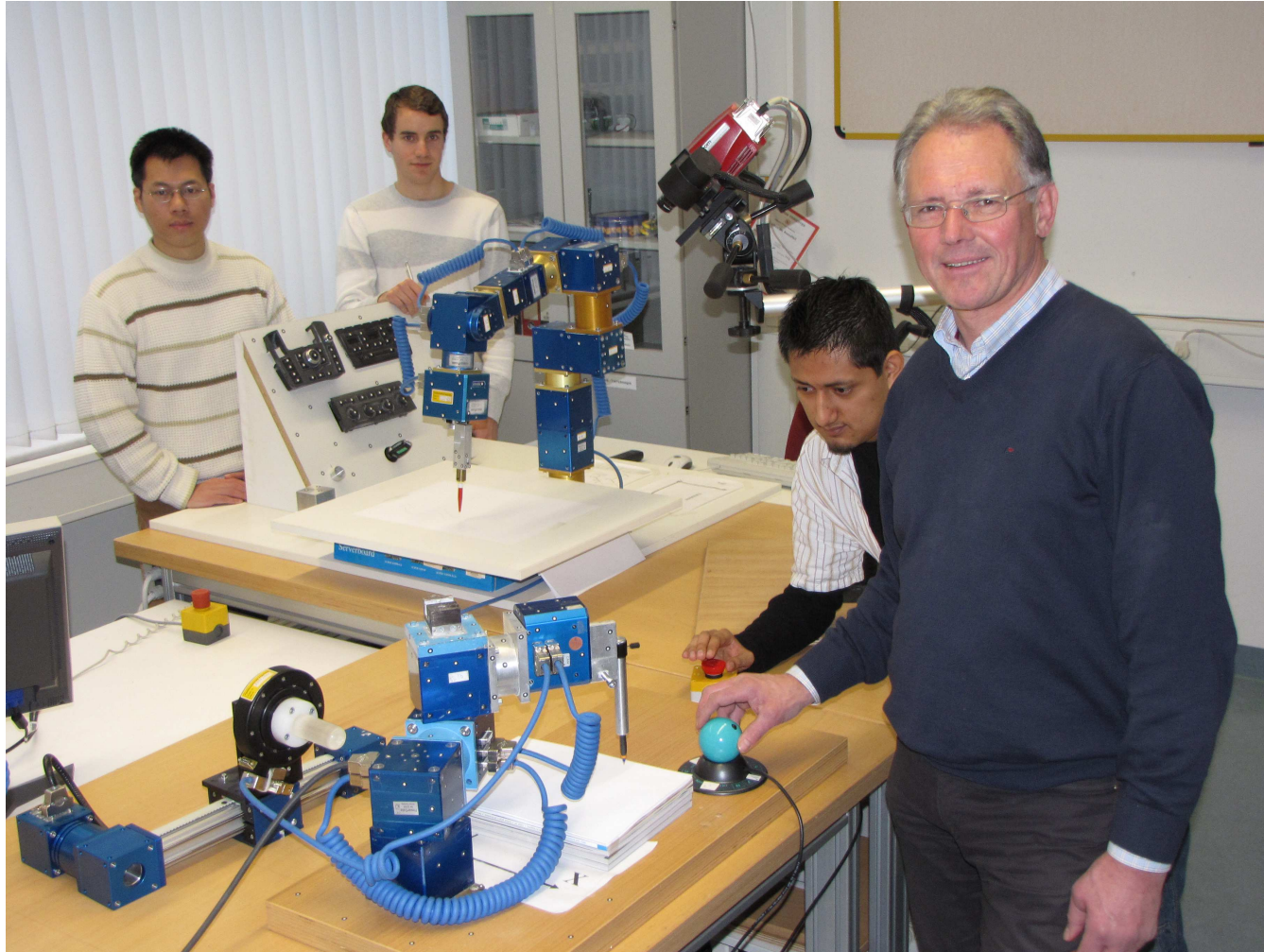


HS Augsburg  
Fak. Informatik

# Vorstellung Labor CIM & Robotik



CIM & Robotik  
Prof. G. Stark





# Schwerpunkte F & E

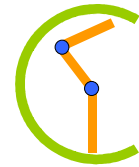


- **Praktische Robotik**
  - Neue Programmierverfahren
  - Weiterentwicklung eigene Software MRobot
  - Sensorbasierte Steuerungsfunktionen
  - Kooperierende Systeme
- **Angewandte Robotik**
  - Mobile Roboter
  - Assistenzroboter für die Fertigung
- **3D-Bildverarbeitung**
  - ATOS-Kamerasystem, Fa. GOM

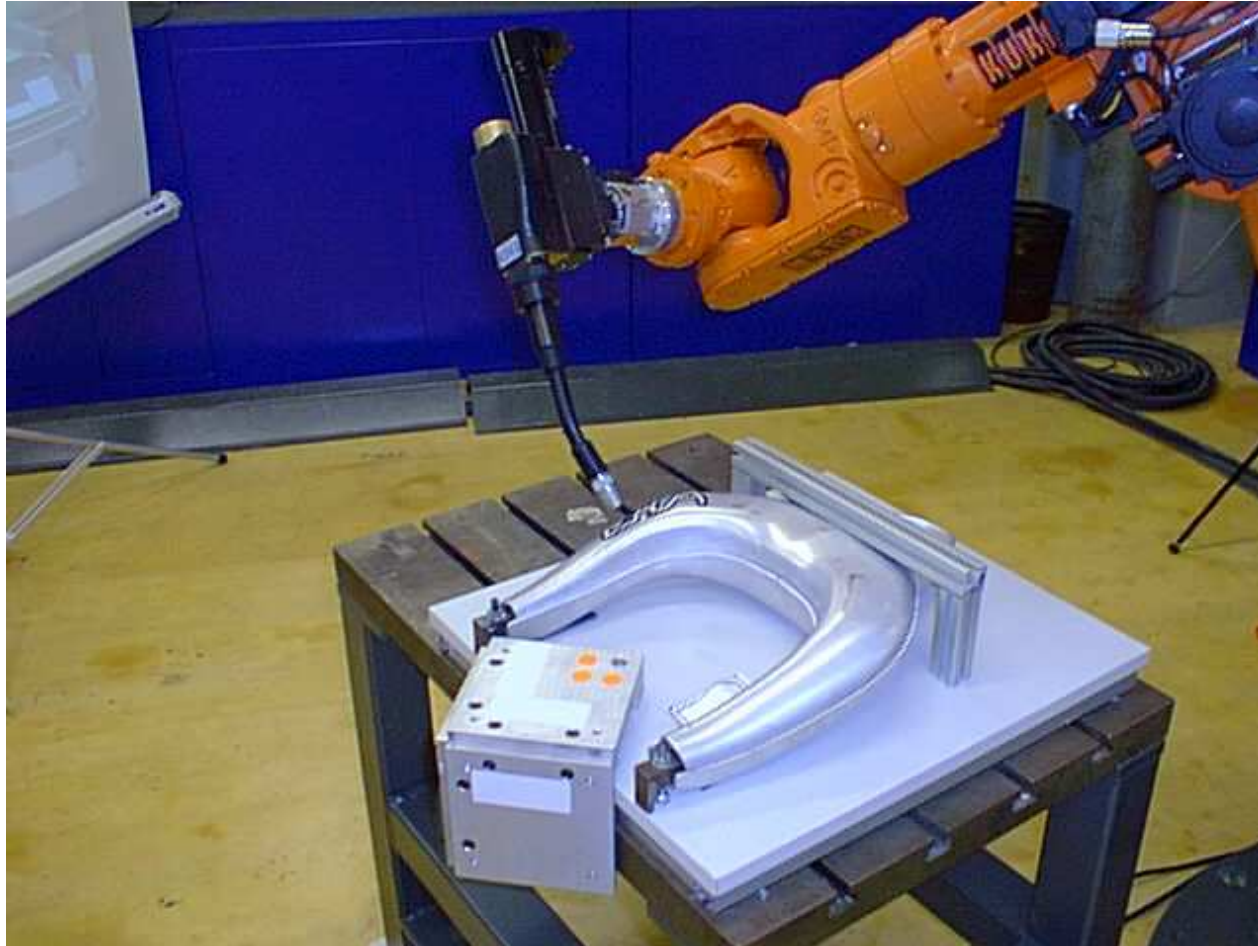


HS Augsburg  
Fak. Informatik

# Programmierung mit 3D-Bilddaten



CIM & Robotik  
Prof. G. Stark



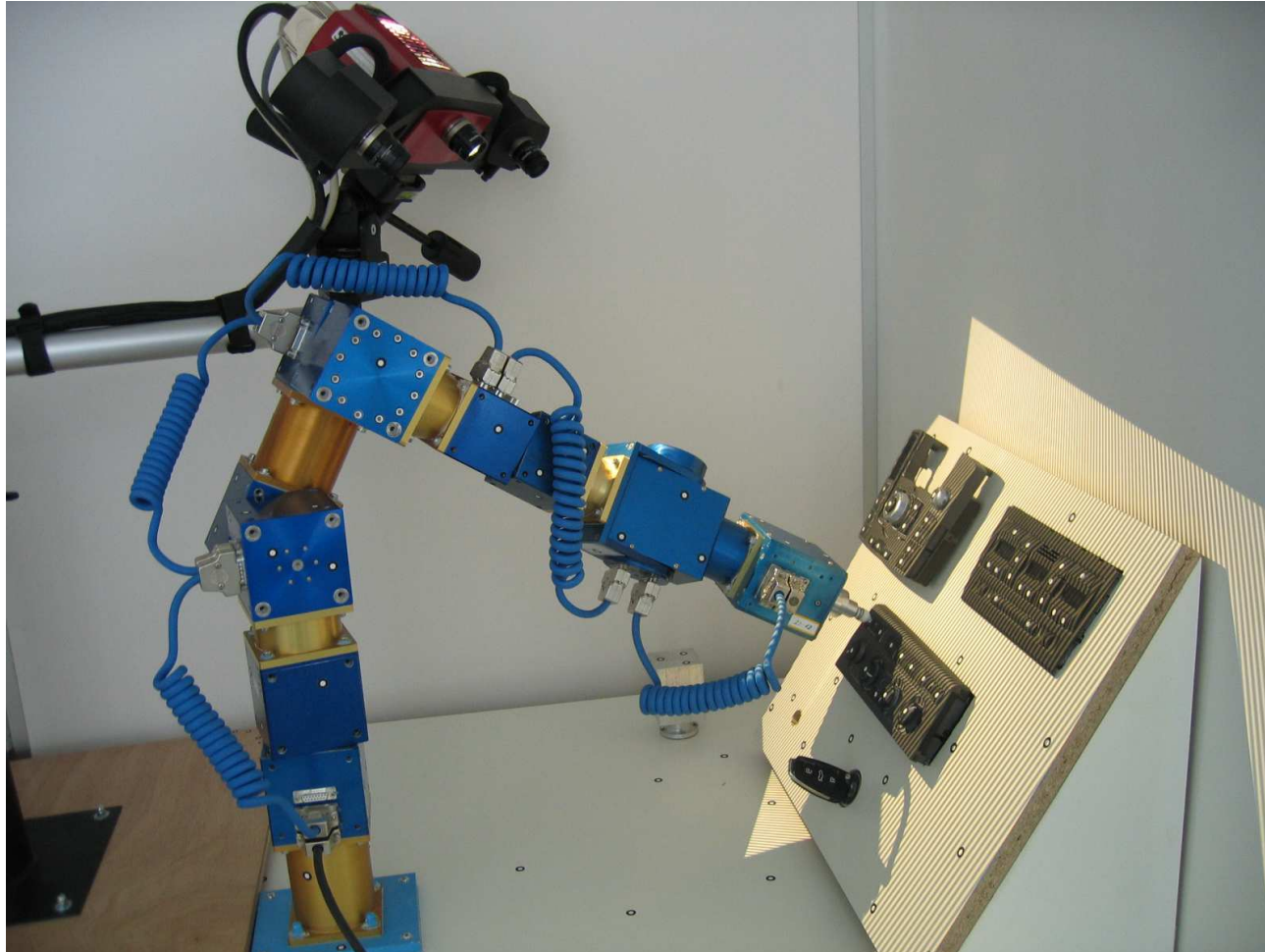


HS Augsburg  
Fak. Informatik

# Automatischer Test von Bedieneinrichtungen

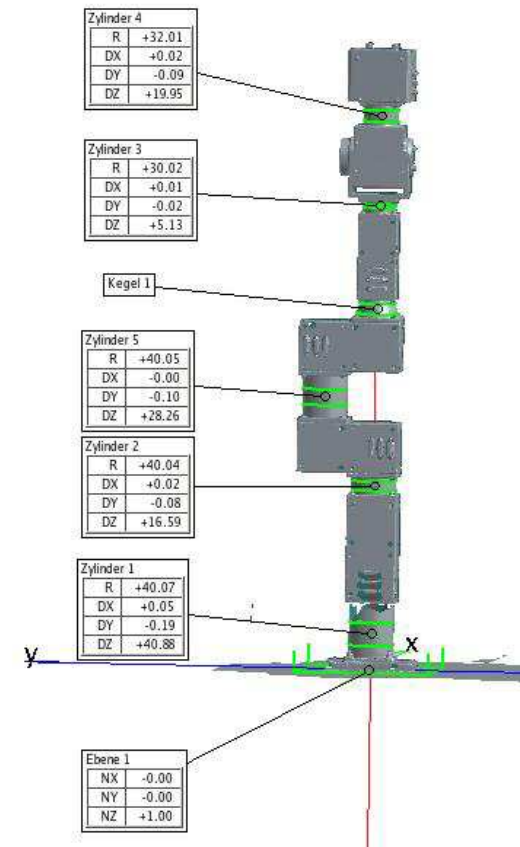
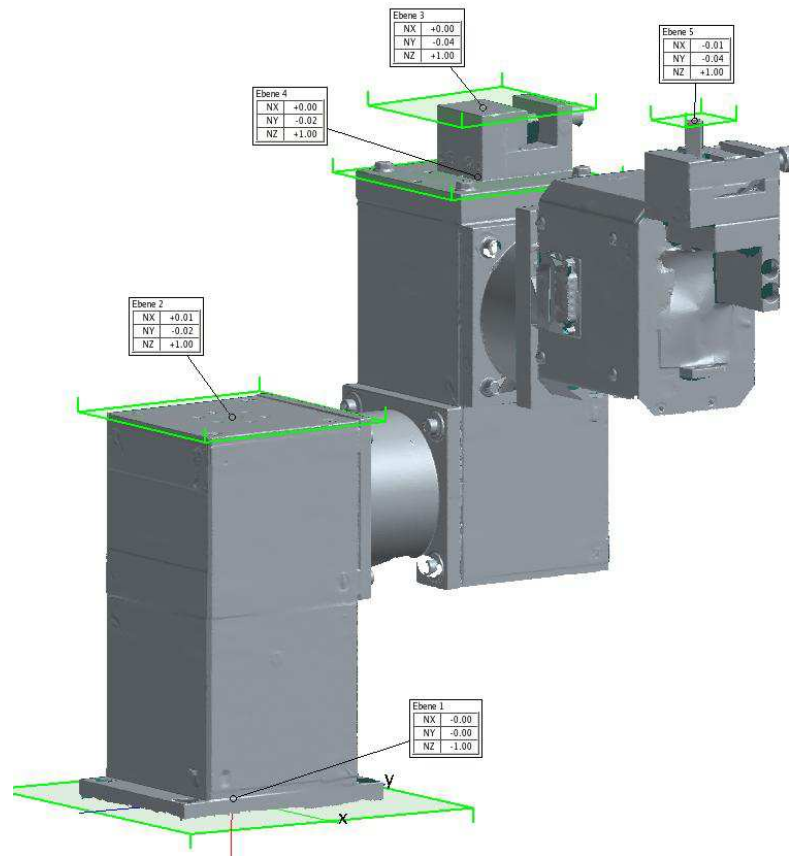


CIM & Robotik  
Prof. G. Stark





# Vermessung mit 3D-Bilddaten



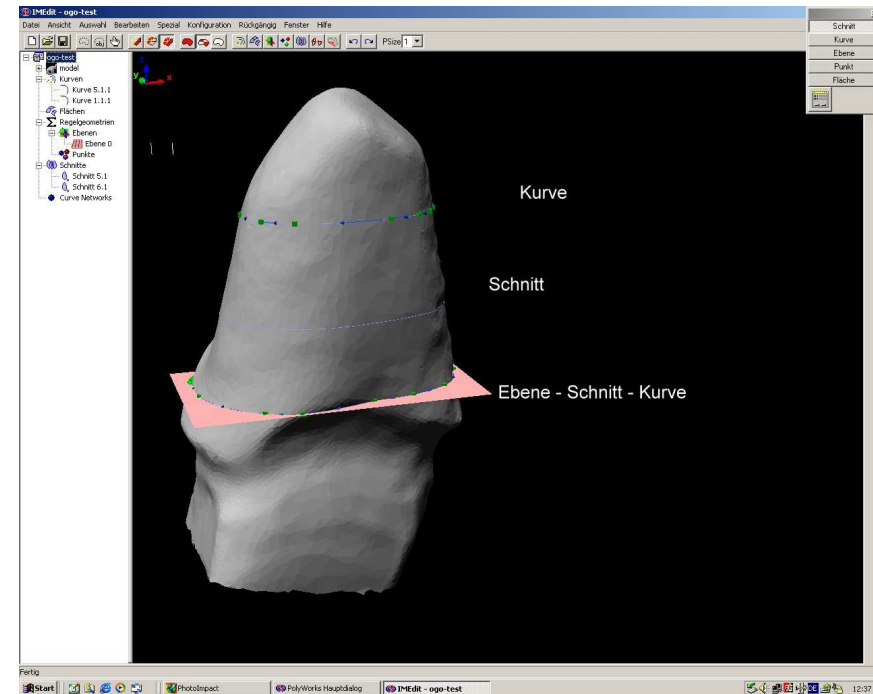


HS Augsburg  
Fak. Informatik

# Fräsen von Zahnkronen



CIM & Robotik  
Prof. G. Stark



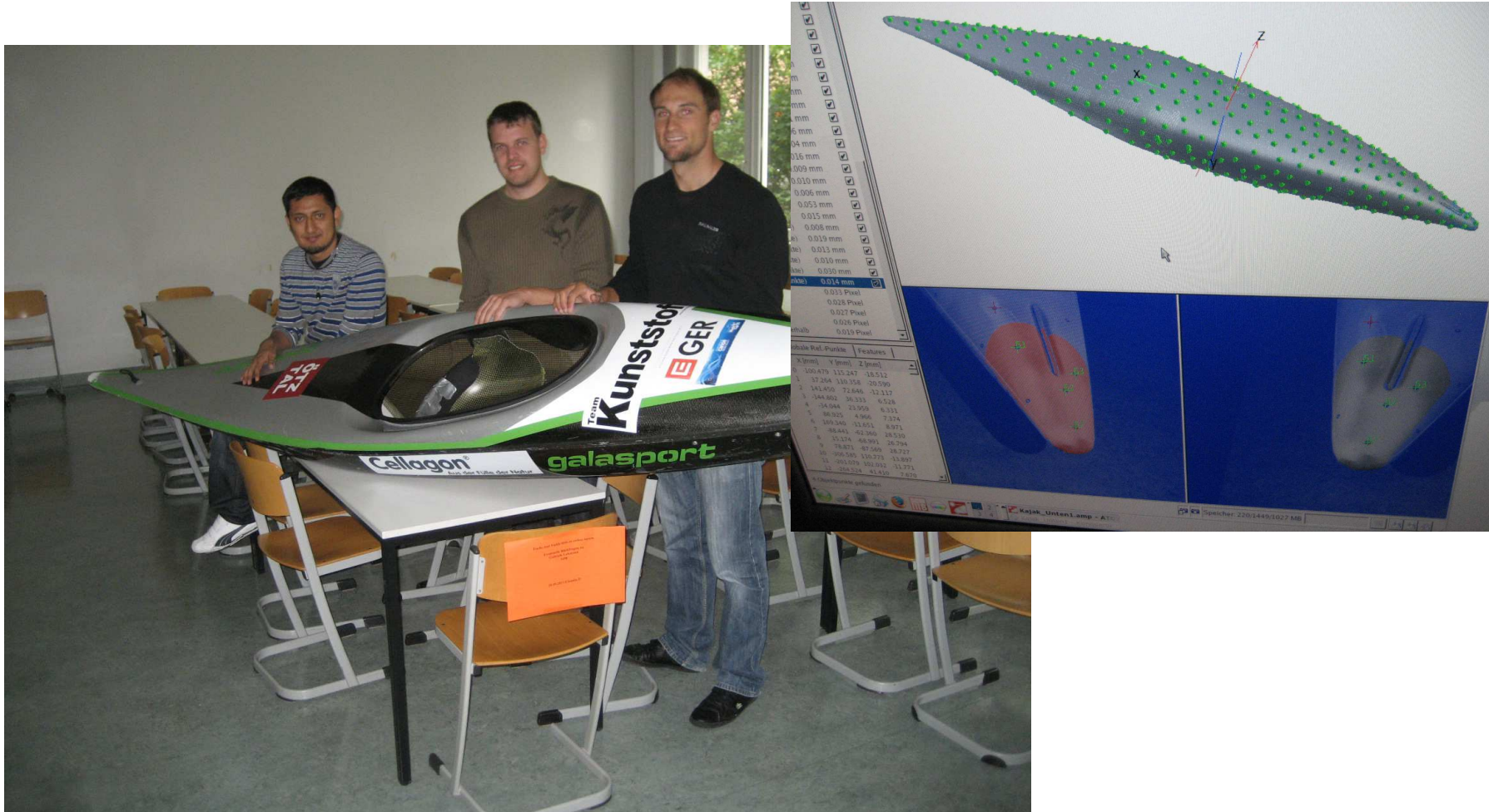


HS Augsburg  
Fak. Informatik

# 3D-Bildverarbeitung: Olympia-Kajak



CIM & Robotik  
Prof. G. Stark







HS Augsburg  
Fak. Informatik

# Messepräsentation mit Mathworks



CIM & Robotik  
Prof. G. Stark





HS Augsburg  
Fak. Informatik

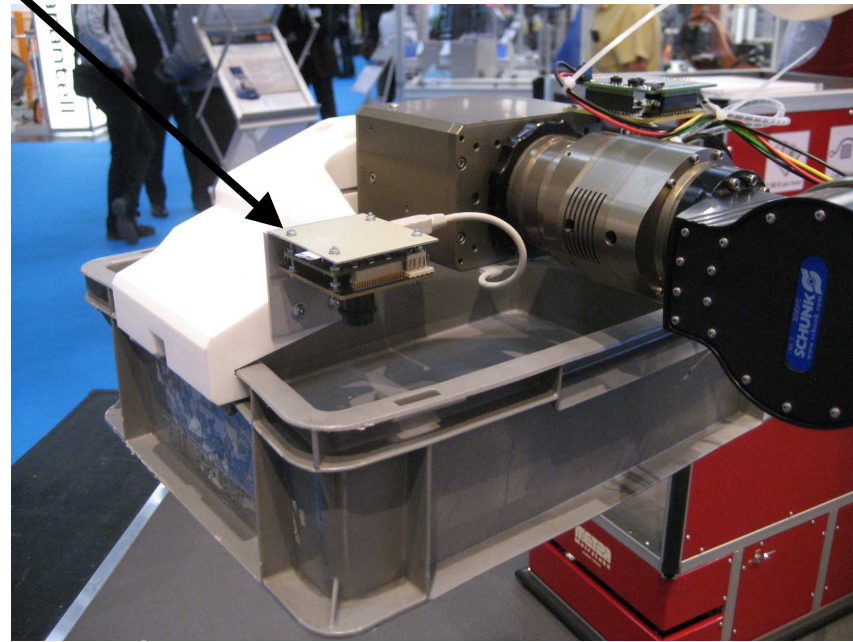
# Forschungsprojekt ECHORD-KANMAN



CIM & Robotik  
Prof. G. Stark

- **Greiferkamera mit eigener Software *MRobot***  
Andocken und Objekterkennung
- **Kanban-Fertigungsprozess**  
Unterstützung durch mobile Robotersysteme

## Greiferkamera





# Allgemeine Zielkriterien für Softwareentwicklung



- Gute **Wartbarkeit** der Software
  - Fehlerbeseitigung
  - Erweiterungen
- Optimaler Informationsfluss zwischen allen beteiligten Personengruppen
- Kosteneffizienz
- Hohe Funktionalität der Software



# Besonderheit bei Eingebetteten Systemen



Rechnersysteme, fest integriert  
in einen technischen Kontext

**Was sind Eingebettete  
Systeme?**

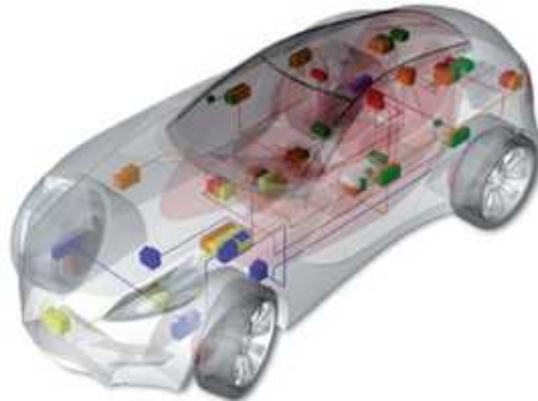
- Enge Kopplung zum technischen Prozess
- Verteilte Systeme mit hoher Dynamik
- Hohe Zuverlässigkeit gefordert
- Wartungsfreundlichkeit und Portierbarkeit der Software



# Wichtige Anwendungsgebiete



Verkehr



Automation

Mechatronik  
Robotik



Konsum

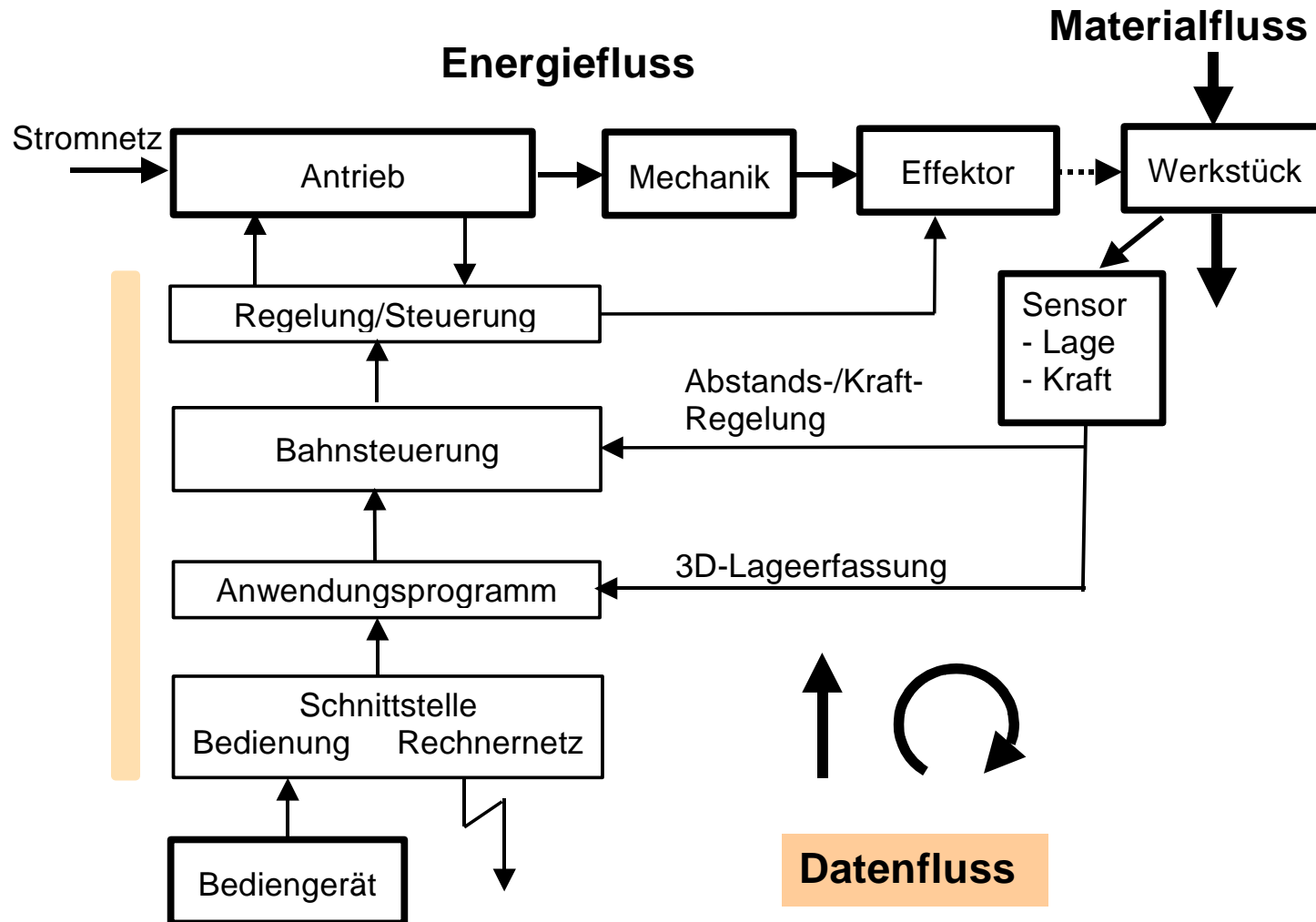


Medizin ...



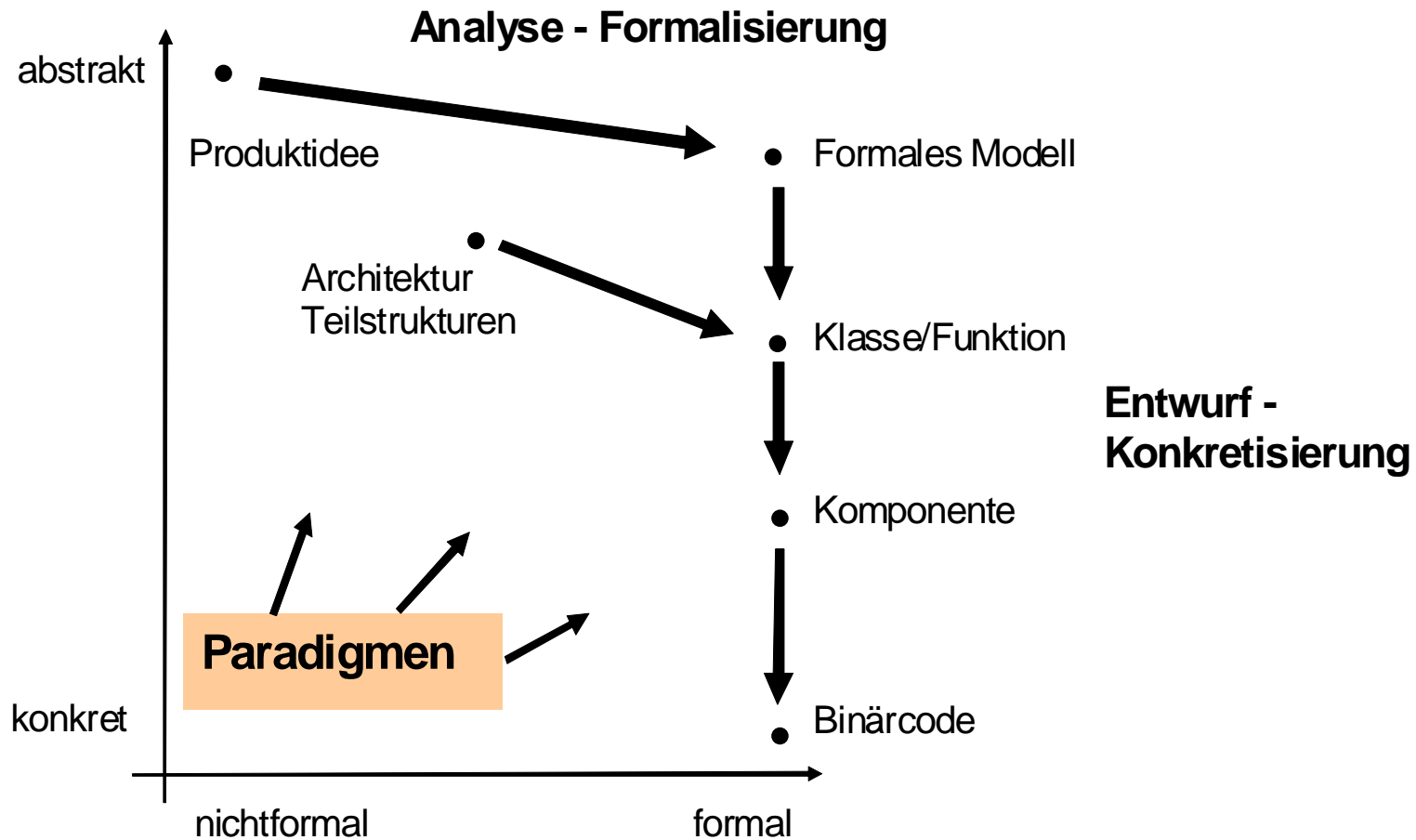


# Anwendung - Mechatronische Systeme





# Darstellung des Programmwissens in der Entwurfsebene





# Alternative Ansätze bei Programmier-Paradigmen



- Primäre Darstellung
  - prozedural
  - deklarativ
- Unterstützung der Wissensdomäne
  - Struktur des Wissens
  - Inhaltliche Darstellung des Wissens
- Abstraktionsebene (Semantische Lücke)





# Vergleich der Paradigmen



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



# Modellbasierte Entwicklung



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

**Was ist die  
Besonderheit ?**

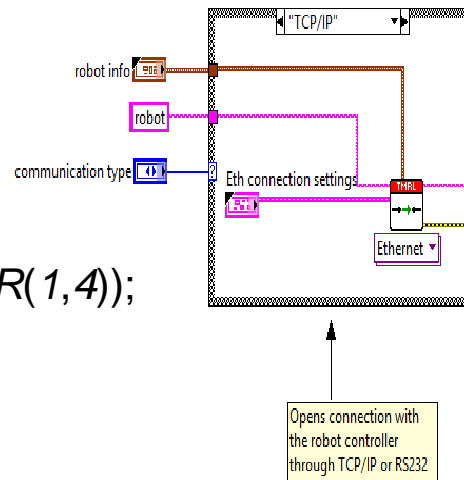


# Domänensprachen Text/Grafik

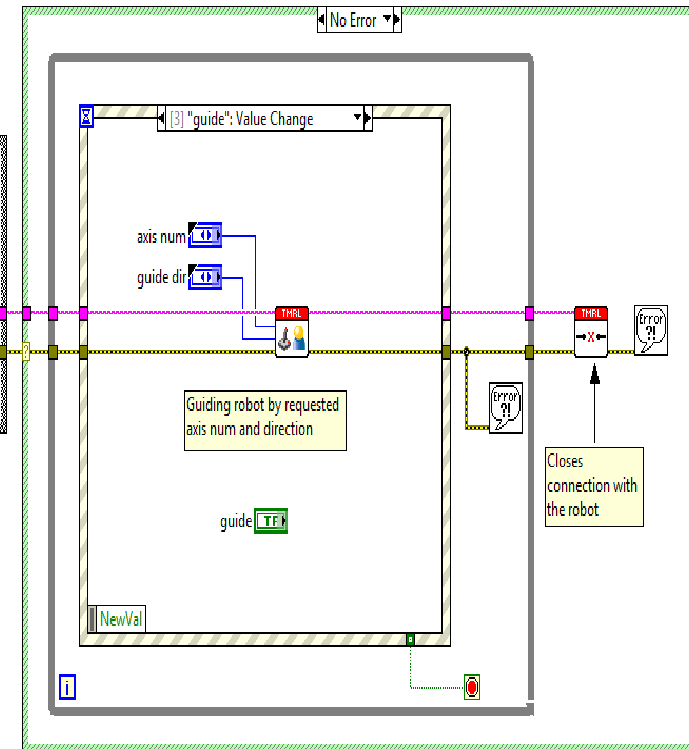


## Matlab-Skript

```
% 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;  
phi=atan2(P2_KR(2,4),P2_KR(1,4));
```

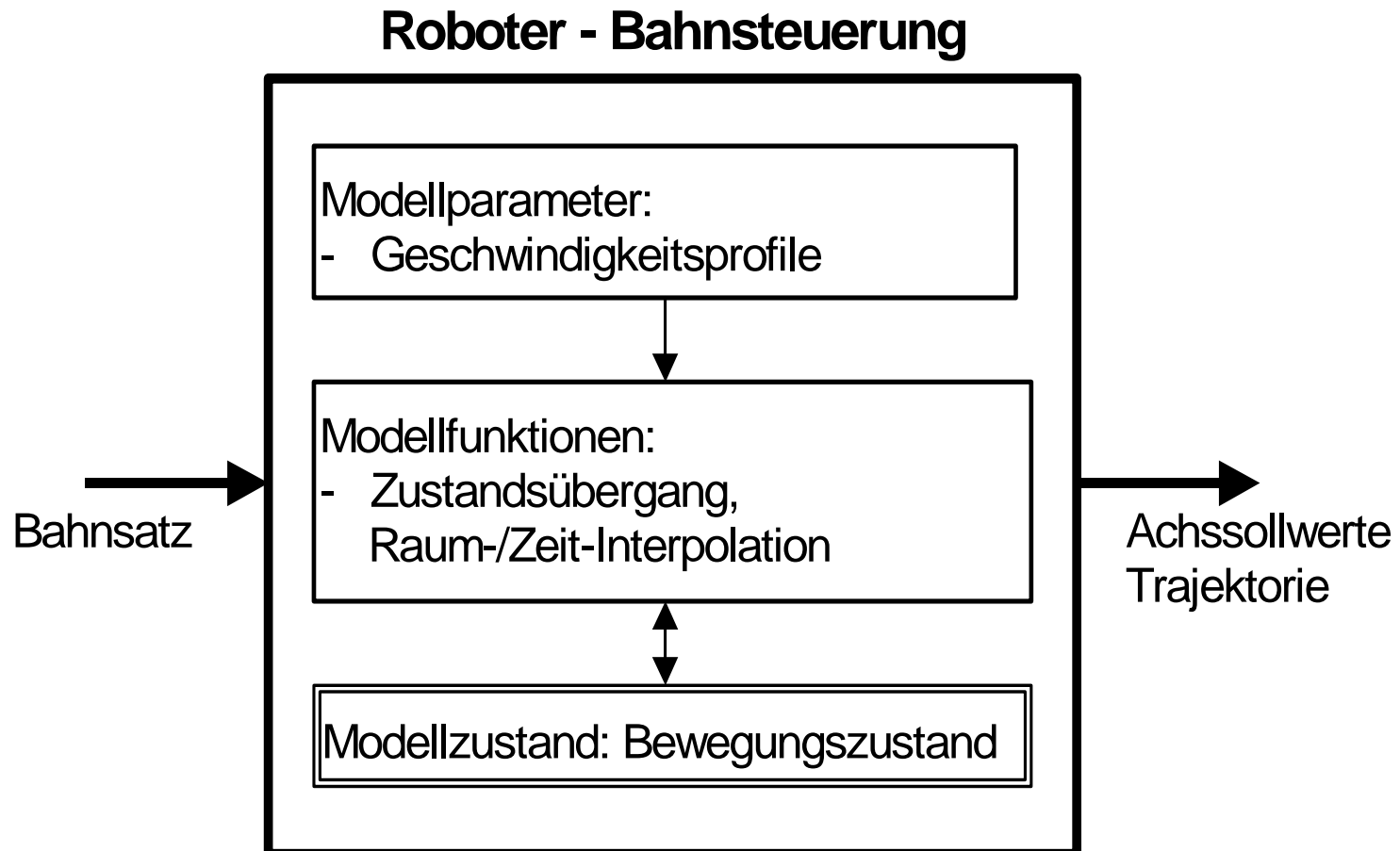


## Labview - Grafik



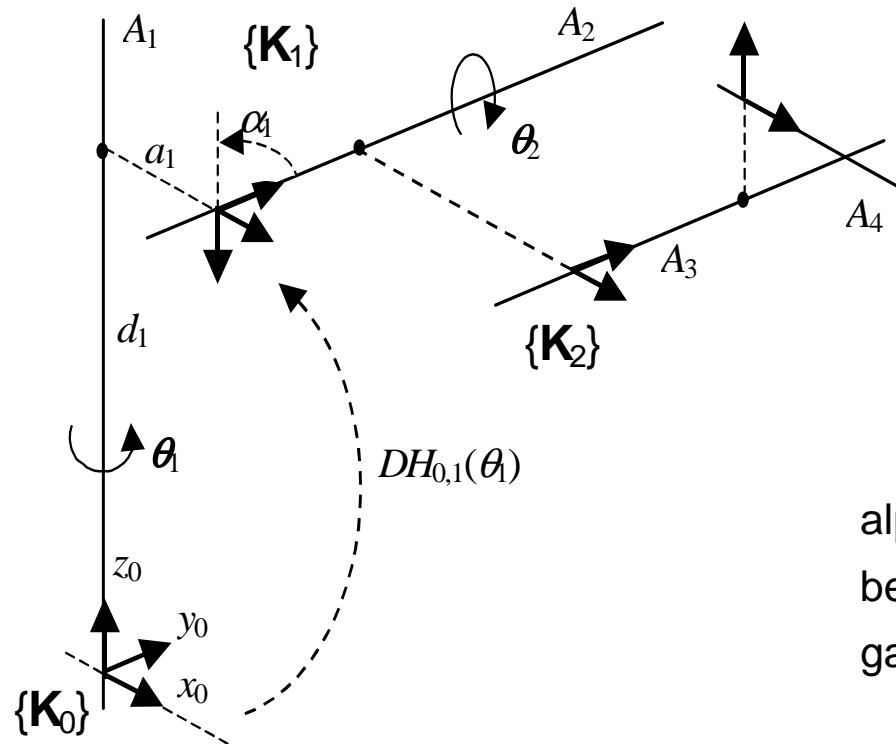


# Beispiel - Zustandsmodell Bewegung





# Roboter - Kinematikmodell



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

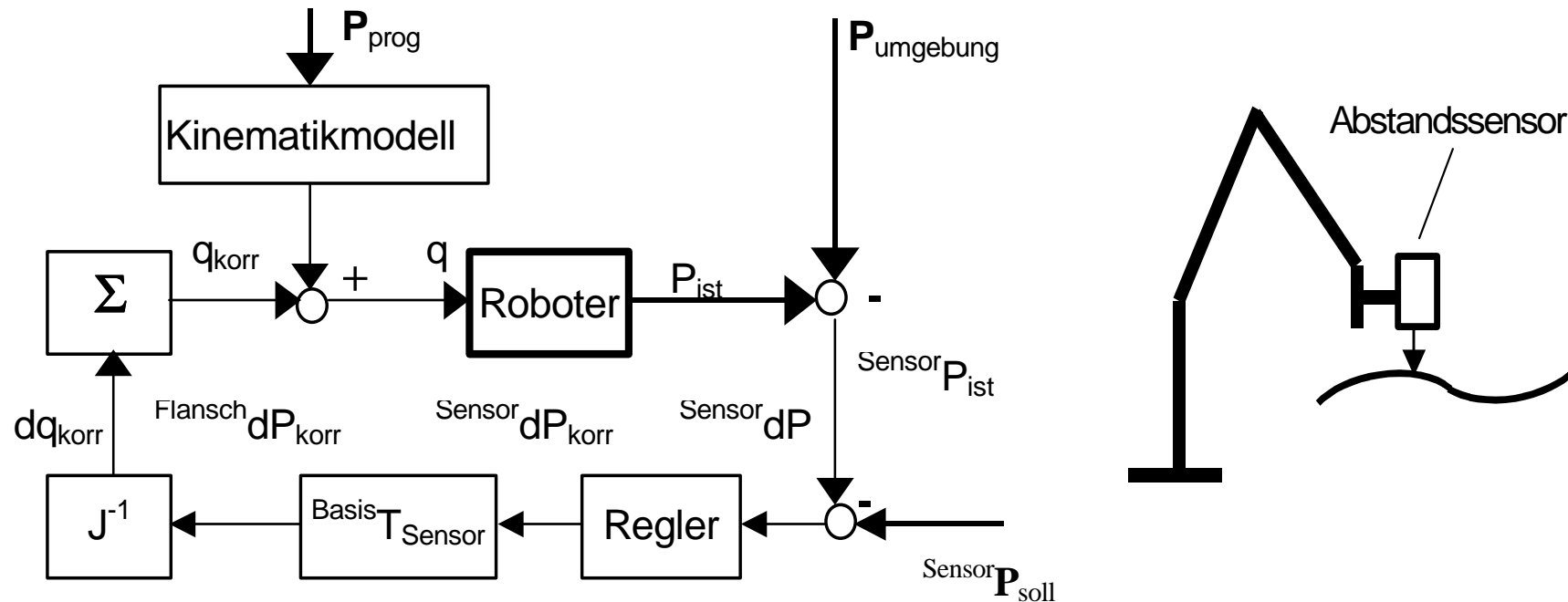
$$\alpha = \text{atan2}(hy1, hx1);$$

$$\beta = \text{acos} \left( \frac{oa^2 - ua^2 + l^2}{2 * l * oa} \right);$$

$$\gamma = \text{acos} \left( \frac{oa^2 + ua^2 - l^2}{2 * oa * ua} \right);$$



# Modell Abstandregelung



P: kartesische Werte, q: achsbezogene Werte



# Komponentenorientierte Programmierung

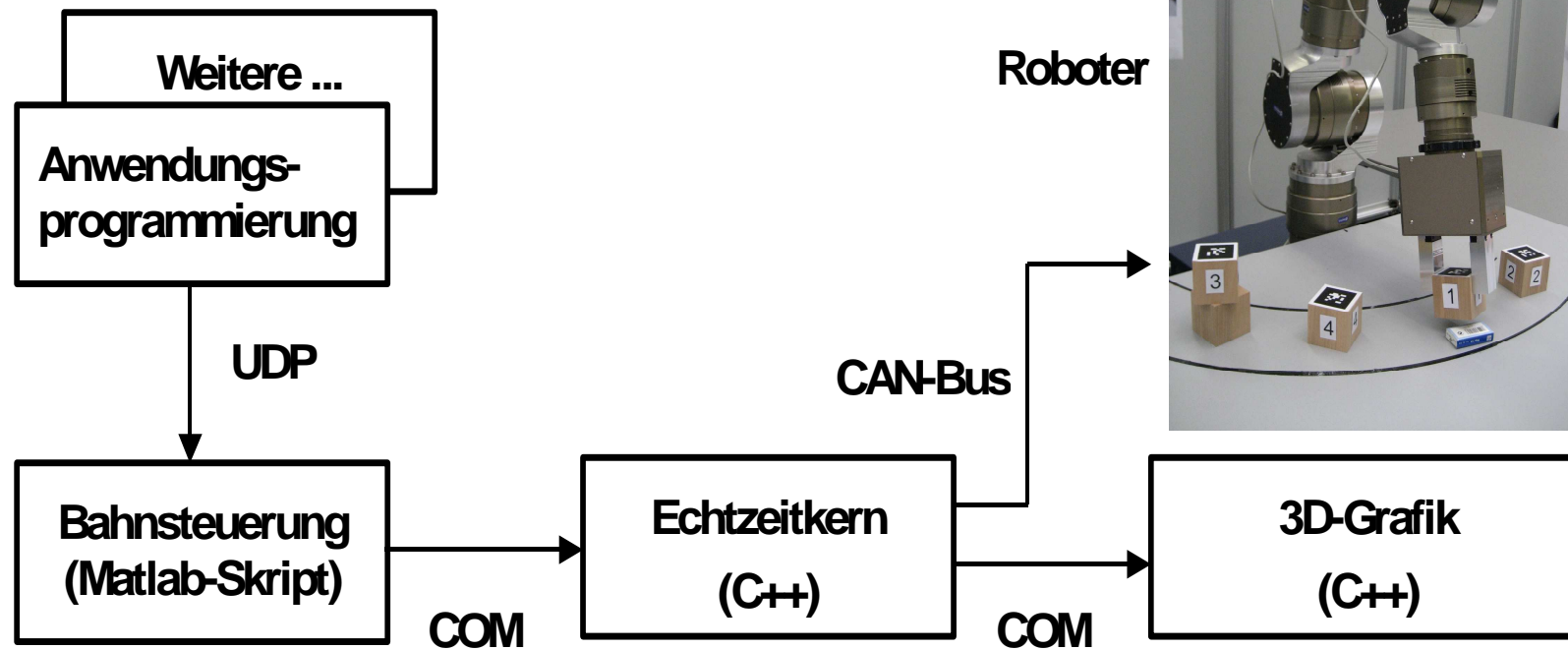
## Leitgedanken



- Softwarekomponenten sind ausführbare Softwareteile, die über standardisierte Schnittstellen benutzt werden.
- Mit Hilfe von Komponenten kann eine Framework-Plugin-Architektur realisiert werden.
- Softwarekomponenten können mit beliebigen Programmiersprachen implementiert werden.
- Softwarekomponenten können dynamisch erzeugt werden.
- Beispiele für standardisierte Schnittstellen:
  - COM von Microsoft
  - CORBA von der OMG ([www.omg.org](http://www.omg.org)),
  - JavaBeans
  - TCP/IP, UDP (allg. Netzwerkprotokolle)



# Komponentenstruktur MRobot / SunSim







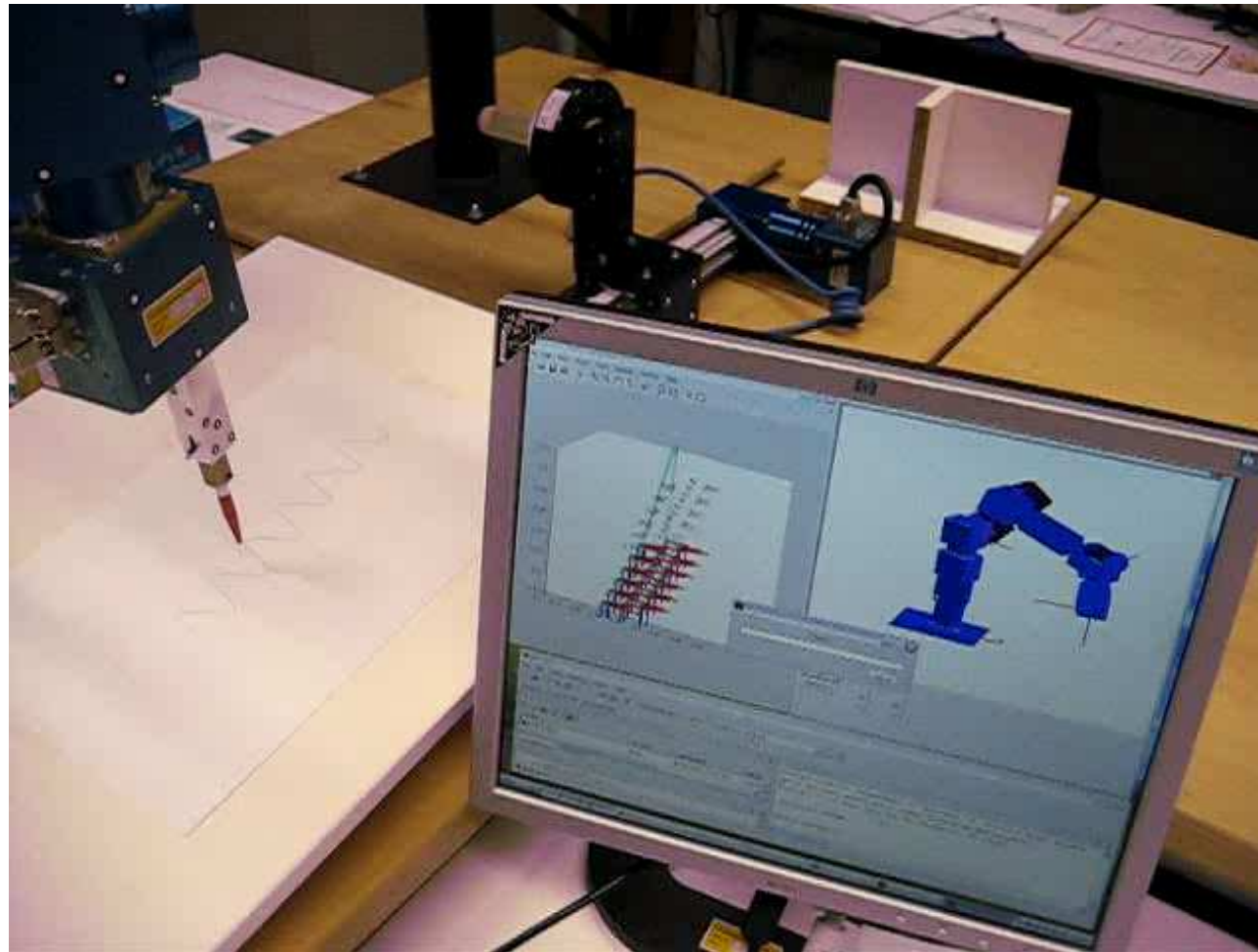
HS Augsburg  
Fak. Informatik

# Anwendungsbeispiele

## Synchrone Robotersteuerung und Simulation



CIM & Robotik  
Prof. G. Stark



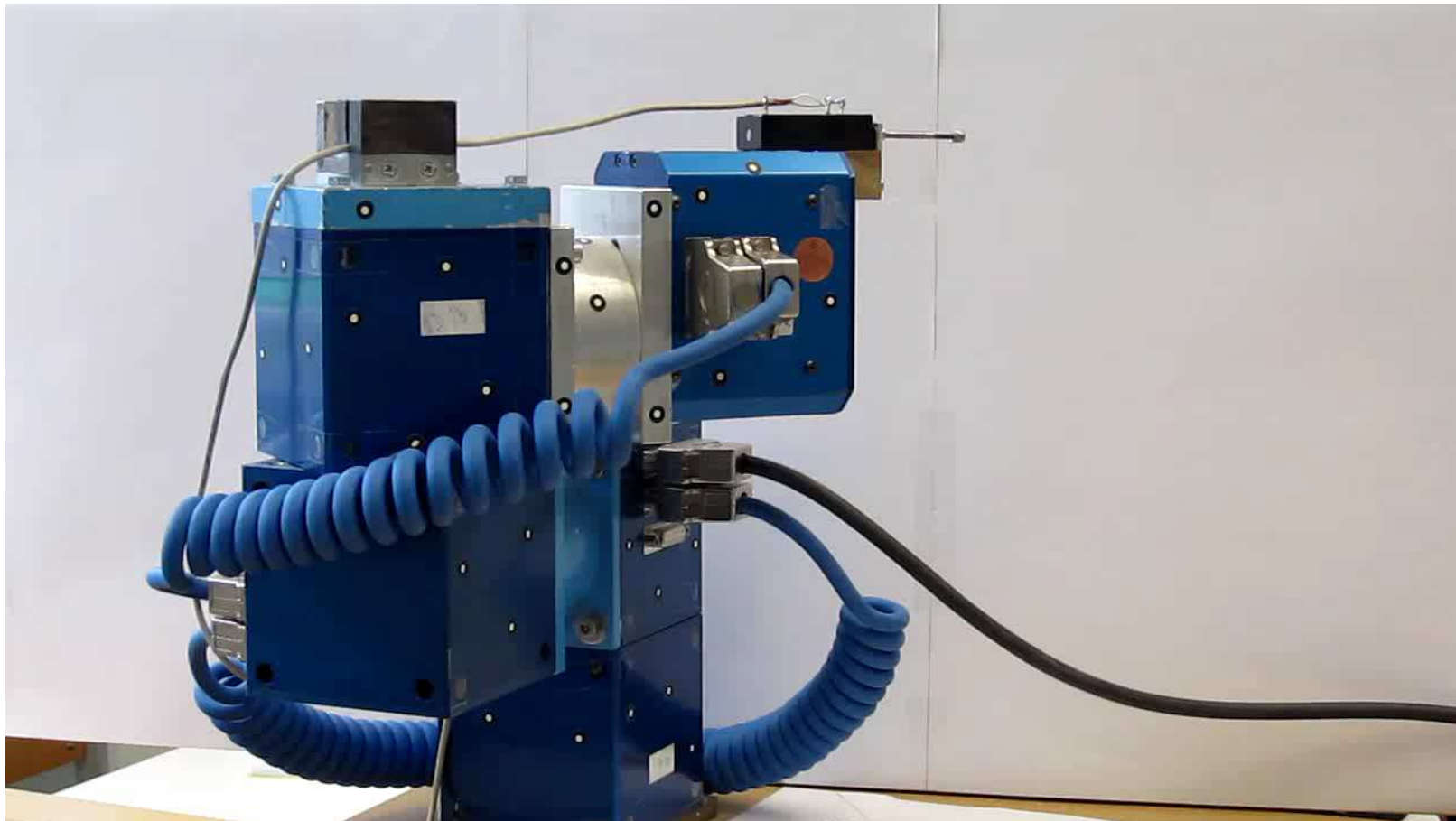


HS Augsburg  
Fak. Informatik

# Sensorsteuerung



CIM & Robotik  
Prof. G. Stark





HS Augsburg  
Fak. Informatik

# Schreiben – ELECTRONICA 2012



CIM & Robotik  
Prof. G. Stark





HS Augsburg  
Fak. Informatik

# Objektlokalisierung mit Greiferkamera



CIM & Robotik  
Prof. G. Stark





HS Augsburg  
Fak. Informatik

# Nullraum-Steuerung KUKA youBot



CIM & Robotik  
Prof. G. Stark





HS Augsburg  
Fak. Informatik

# Simulationskomponente für KUKA Sunrise Workbench



CIM & Robotik  
Prof. G. Stark

Debuggen - MRobot/src/application/RobotApplication.java - Sunrise Workbench

RobotApplication [Zeile: 46] - run()  
RobotApplication [Zeile: 64] - run()

```
public void run() {  
  
    this.mRobot.move(BasicMotions.Lin( getFrame("/P1") ) );  
    this.mRobot.move(BasicMotions.Lin( getFrame("/P2") ) );  
    this.mRobot.move(BasicMotions.Lin( getFrame("/P3") ) );  
  
    JointPosition jointPosition = new JointPosition(8);  
    double[] axisAngles = { 0, 0, -40, 20, -180, 90, 0, -90 }; // LBR  
    double[] axisAngles2 = { 0, 0, 0, -40, -180, 70, -15, -90 };  
    double[] axisAngles3 = { 0, 0, 0, 40, -200, 70, -15, -90 };  
}
```

Konsole

RobotApplication [Java-Anwendung] C:\Program Files (x86)\KUKA\Sunrise Workbench\jdk\bin\javaw.exe (29.04.2014 15:14:  
LIN to 'P2 [X=557.77 Y=-131.02 Z=409.26 A=0.00 B=3.14 C=0.00]'  
Rueckgabe vom Server:RUECK

LIN to 'P3 [X=410.75 Y=-131.31 Z=527.14 A=0.00 B=3.14 C=0.00]'  
Rueckgabe vom Server:RUECK

3D-Fenster

OK  
Abbrechen  
Zoom  
IN  
OUT

Analytischen und Teachen

Bahn	Schrittweise neu abfahren	Analyse	Teachen
Unterbrechen	Schritte 5	Achse 1	Handverfahr
Rückwärts	Start	Achse	Pos teachen
Wiederholung	Vorwärts	Kartesisch	Punktliste



HS Augsburg  
Fak. Informatik

# Simulation KUKA omniRob



CIM & Robotik  
Prof. G. Stark

The screenshot displays the Sunrise Workbench environment. The main window is titled "Debuggen - MRobot/src/application/RobotApplication.java - Sunrise Workbench". It contains a Java code editor with the following code:

```
JointPosition jointPosition = new JointPosition(8);
double[] axisAngles = { 0.2, 0.4, 0, 80, -180, 90, 0,
double[] axisAngles2 = { 0, 0, 0, -40, -180, 70, -15,
double[] axisAngles3 = { 0, 0, 0, 40, -200, 70, -15,
double[] axisAngles4 = {0.799, 0.399, 0.000, -90.000,
double[] axisAngles5 = {0.799, 0.399, 0.000, -69.87,

jointPosition.set(axisAngles4);
this.mRobot.move(BasicMotions.ptp(jointPosition).setJc

jointPosition.set(axisAngles5);
this.mRobot.move(BasicMotions.ptp(jointPosition).setJc

this.mRobot.move(BasicMotions.lin( getFrame("/P1") )
this.mRobot.move(BasicMotions.lin( getFrame("/P2") )
this.mRobot.move(BasicMotions.lin( getFrame("/P3") )

jointPosition.set(axisAngles2);
this.mRobot.move(BasicMotions.ptp(jointPosition).setJc
```

The 3D-Fenster shows a yellow KUKA robot arm on a yellow base, positioned next to a green rectangular block on a blue surface. The control panel at the bottom right, titled "Analysieren und Teachen", includes buttons for "Unterbrechen", "Rückwärts", "Wiederholung", "Schrittweise neu abfahren", "Schritte" (set to 5), "Start", "Vorwärts", "Analyse", "Achse" (set to 1), "Kartesisch", "Teachen", "Handverfahren", "Post teachen", and "Punktliste".



# Neue Verfahren und Technologien

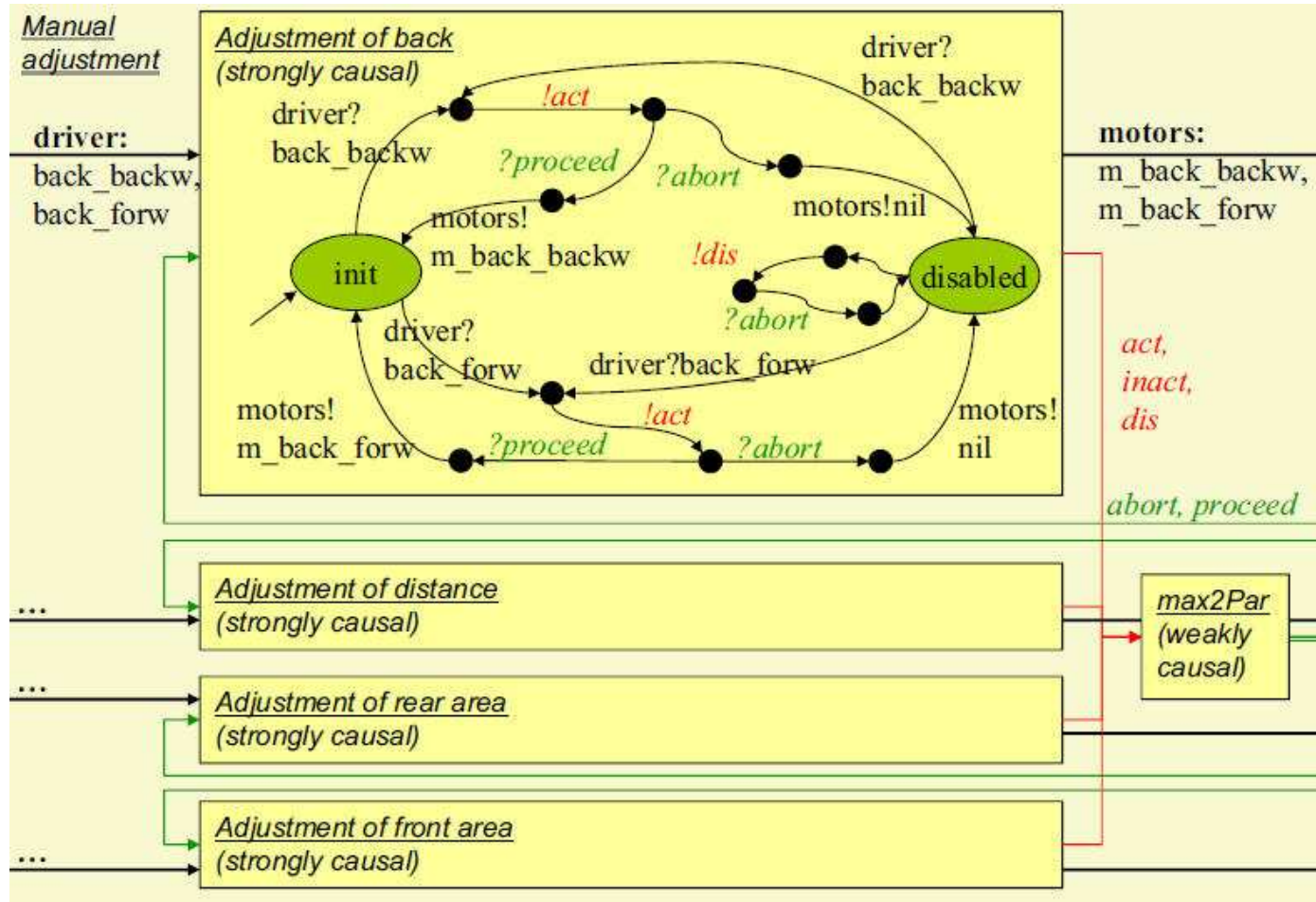


- Modellierung von realen Systemen, Prozessen
  - Matlab/Simulink, Realtime Workshop
  - Modelica
  - Labview
  - ASCET SD
  - SCADE
  
- Datenmodelle
  - Eclipse, Modelling Framework (EMF)
  - Visual Studio, Modellierungs-SDK



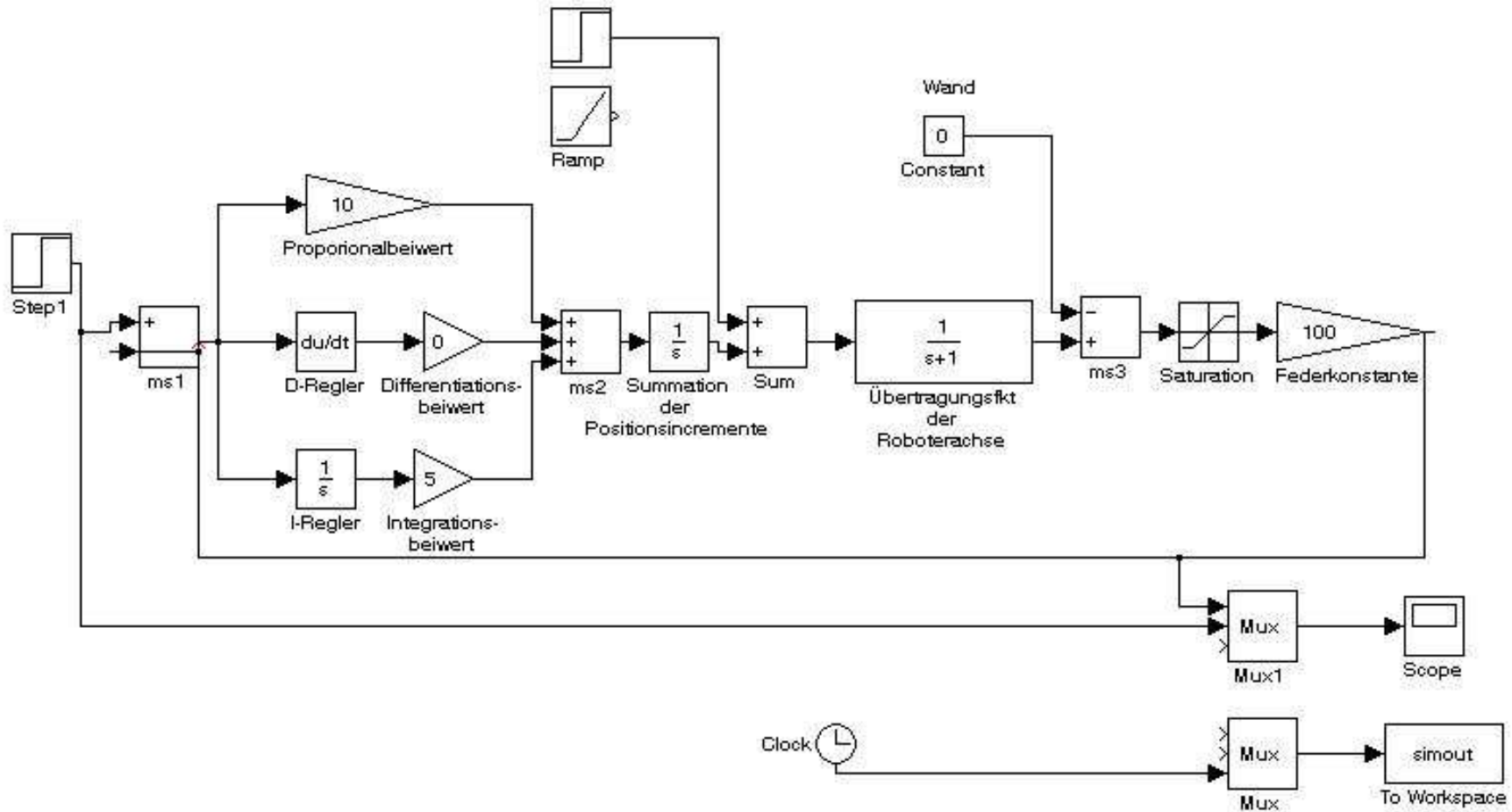


# Beispiel - Prozessmodell





# Matlab/Simulink - Signalfluss





# Modelica – Modellierung phys. Systeme



## Example: Hierarchical Modelica Model

textual representation

```

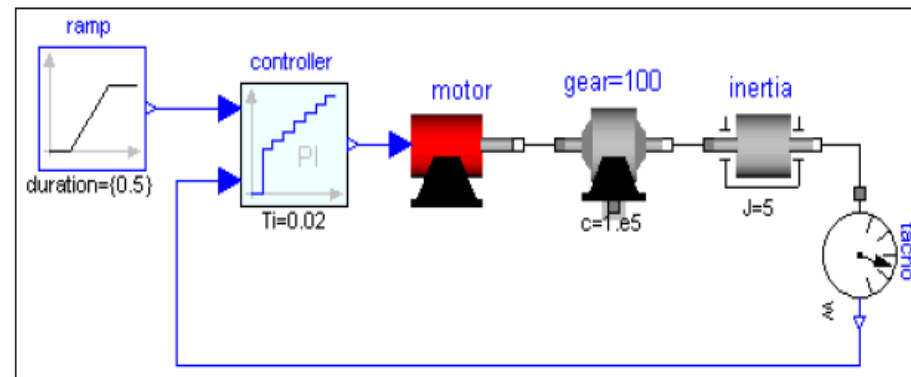
model MotorDrive
  PI controller;
  Ramp ramp;
  Motor motor;
  Gearbox gear ratio = 100;
  Inertia inertia (J = 10);
  SpeedSensor tacho;
equation
  connect (controller.y      , motor.i_ref);

```

Class name

Modifier

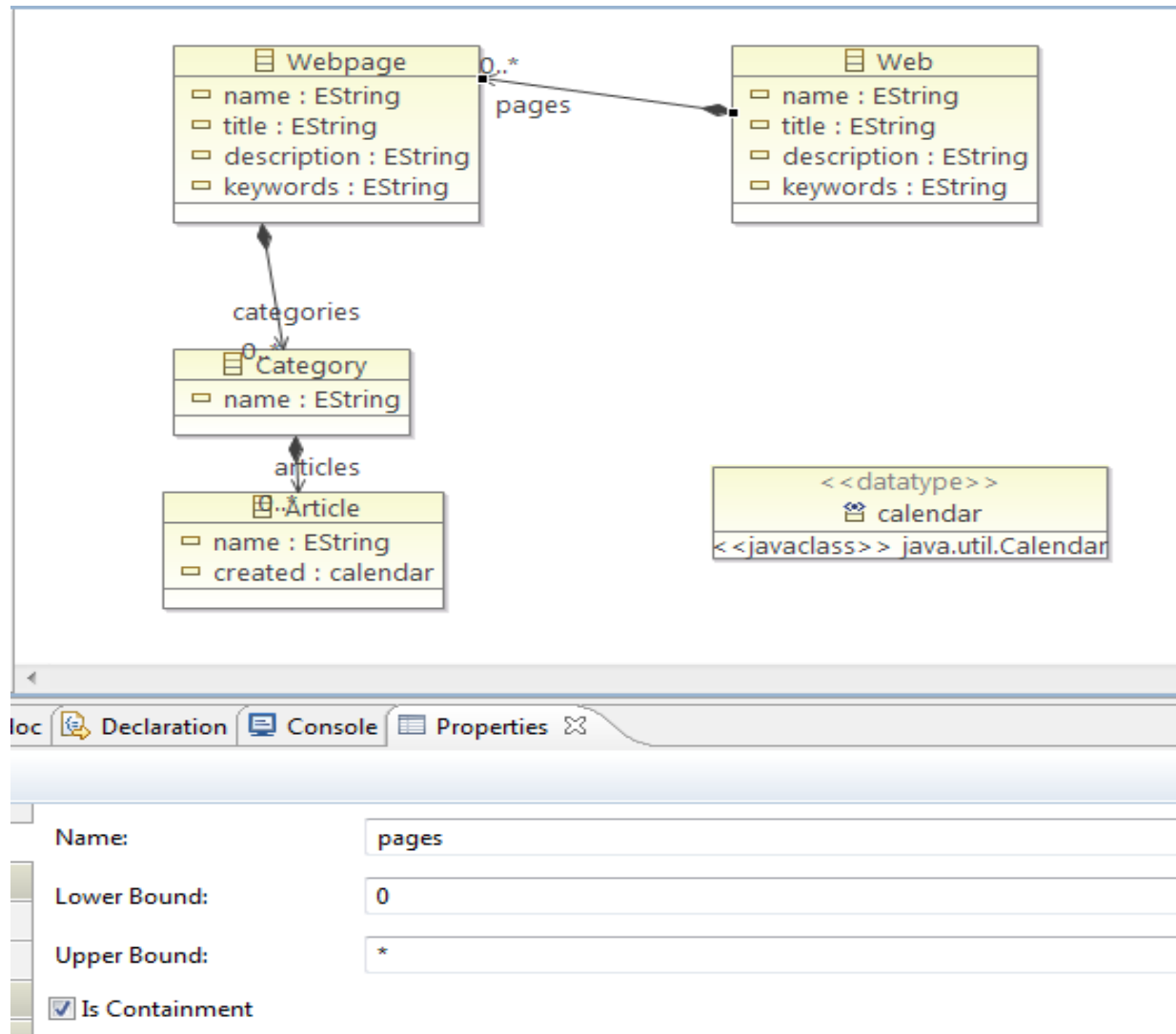
Instance name



graphical representation



# Datenmodell in EMF





# Zusammenfassung



- Die Programmierung von Eingebetteten Systemen erfordert Wissen über das
  - umgebende System,
  - die auszuführenden Prozesse.
- Die modellbasierte Programmierung unterstützt die
  - formale Darstellung ohne Realisierungsdetails,
  - weitgehende Plattformunabhängigkeit.
- Die dargestellten Modelle können sofort simuliert und zur Codegenerierung benutzt werden.
- Die Aufteilung der Software auf binäre Komponenten erleichtert Installation und Wartung.
- Das Ergebnis ist kosteneffiziente Software mit guter Wartbarkeit und hoher Qualität

**Quintessenz ?**



HS Augsburg  
Fak. Informatik

# Modellbasierte Prog: Robotik mit Matlab



CIM & Robotik  
Prof. G. Stark





HS Augsburg  
Fak. Informatik



CIM & Robotik  
Prof. G. Stark

**Vielen Dank für ihre  
Aufmerksamkeit!**

