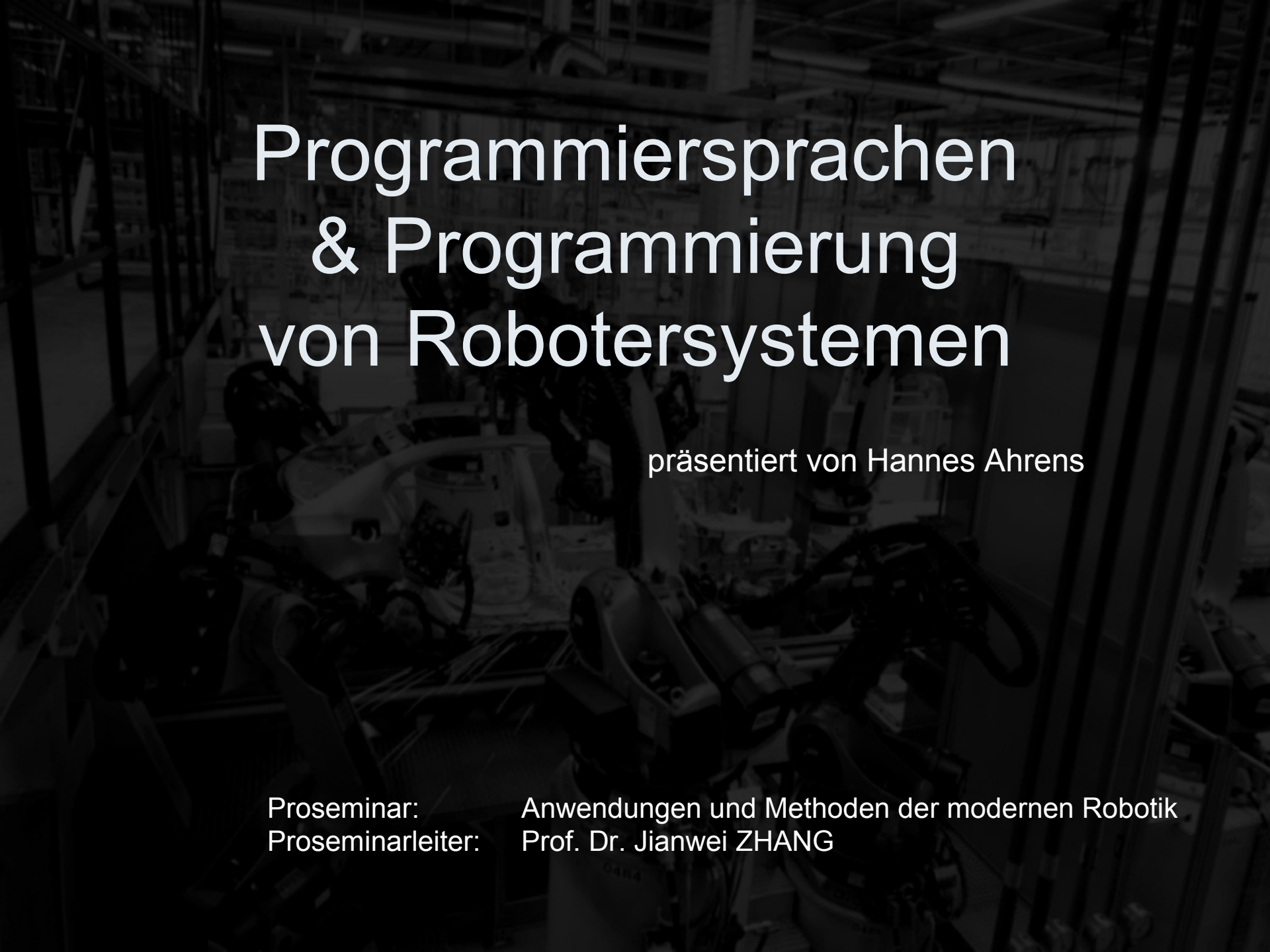


Programmiersprachen & Programmierung von Robotersystemen

präsentiert von Hannes Ahrens

Proseminar: Anwendungen und Methoden der modernen Robotik
Proseminarleiter: Prof. Dr. Jianwei ZHANG

Gliederung

Die Programmierverfahren: was man unter Roboterprogrammierung versteht
- Online
- Offline

Die Entwicklung der Roboterprogrammierung

Verschiedene Programmiersprachen

RCCL (Multi-RCCL)
- Was RCCL ist
- RCCL im Einsatz → Unser Arm mit integrierter Baretthand
- Wie RCCL aufgebaut ist und funktioniert
- Ein Beispiel

Diskussionsideen

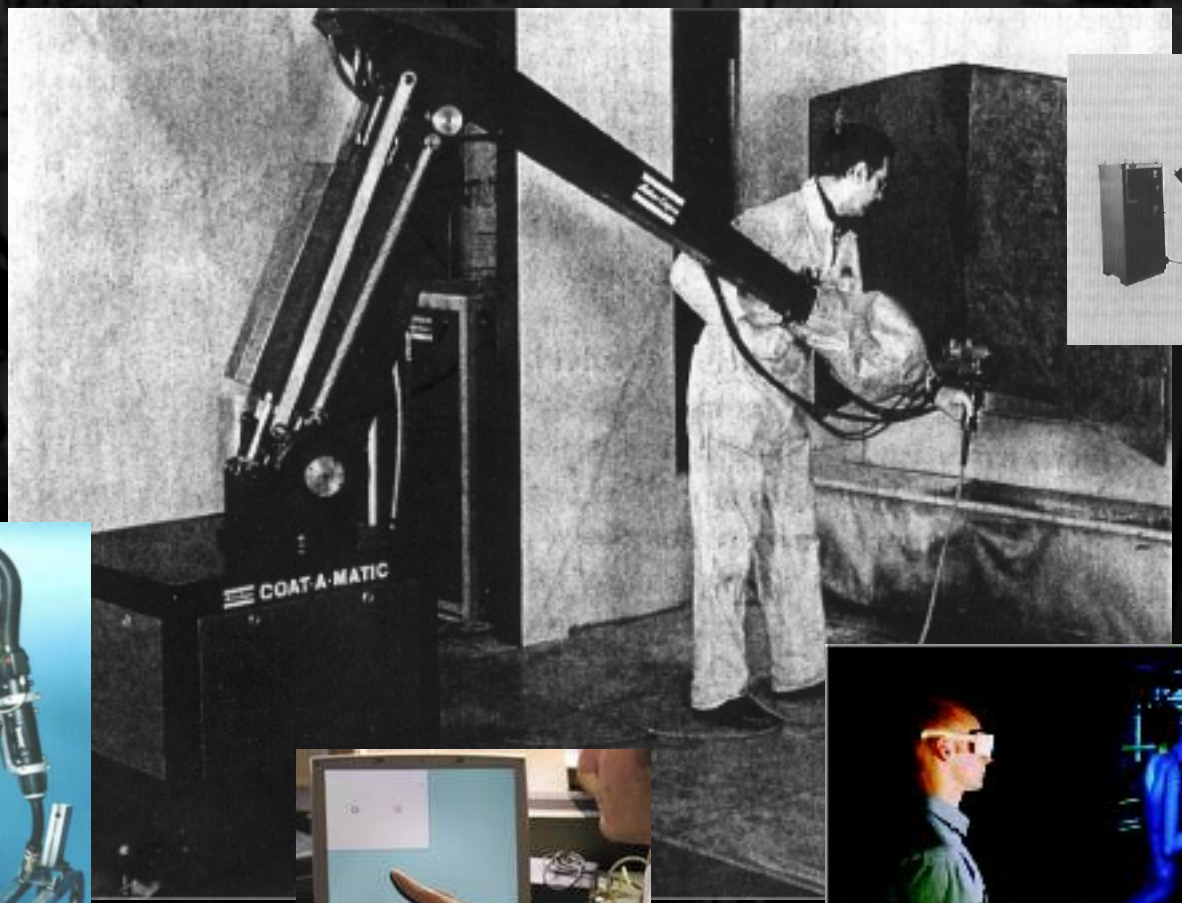
Quellen

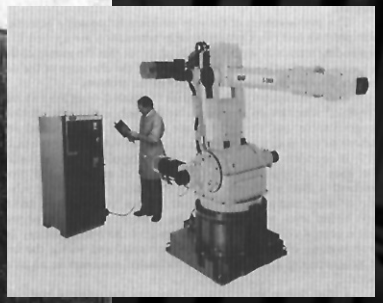
Die Programmierverfahren

- Was man unter Roboterprogrammierung versteht

Online

- Teach-In (Play-Back)





Die Programmierverfahren

- Was man unter Roboterprogrammierung versteht

Online

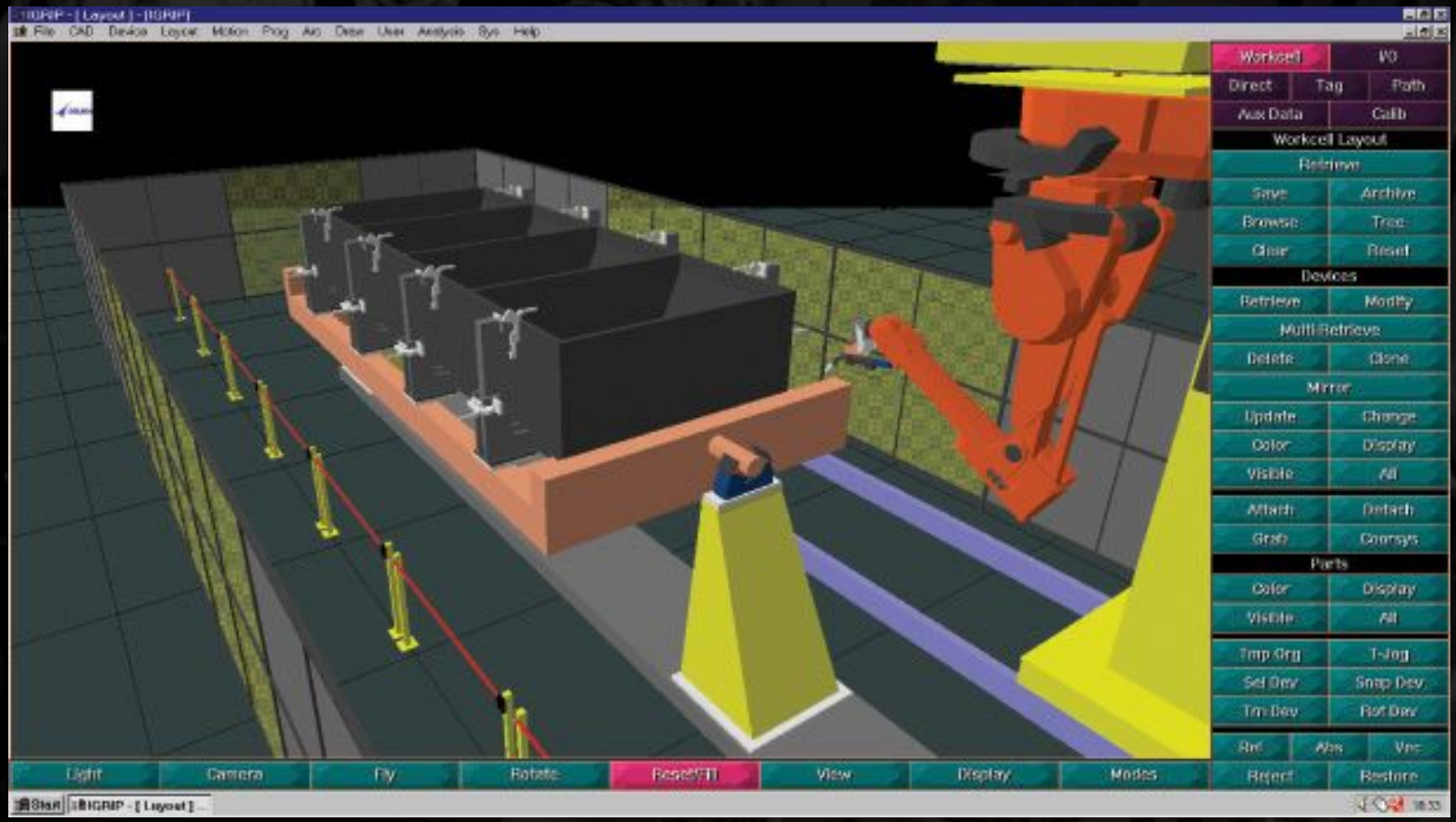
- direkte Parametereingabe per Softwareschnittstelle

Die Programmierverfahren

- Was man unter Roboterprogrammierung versteht

Offline

- 3D-Simulator

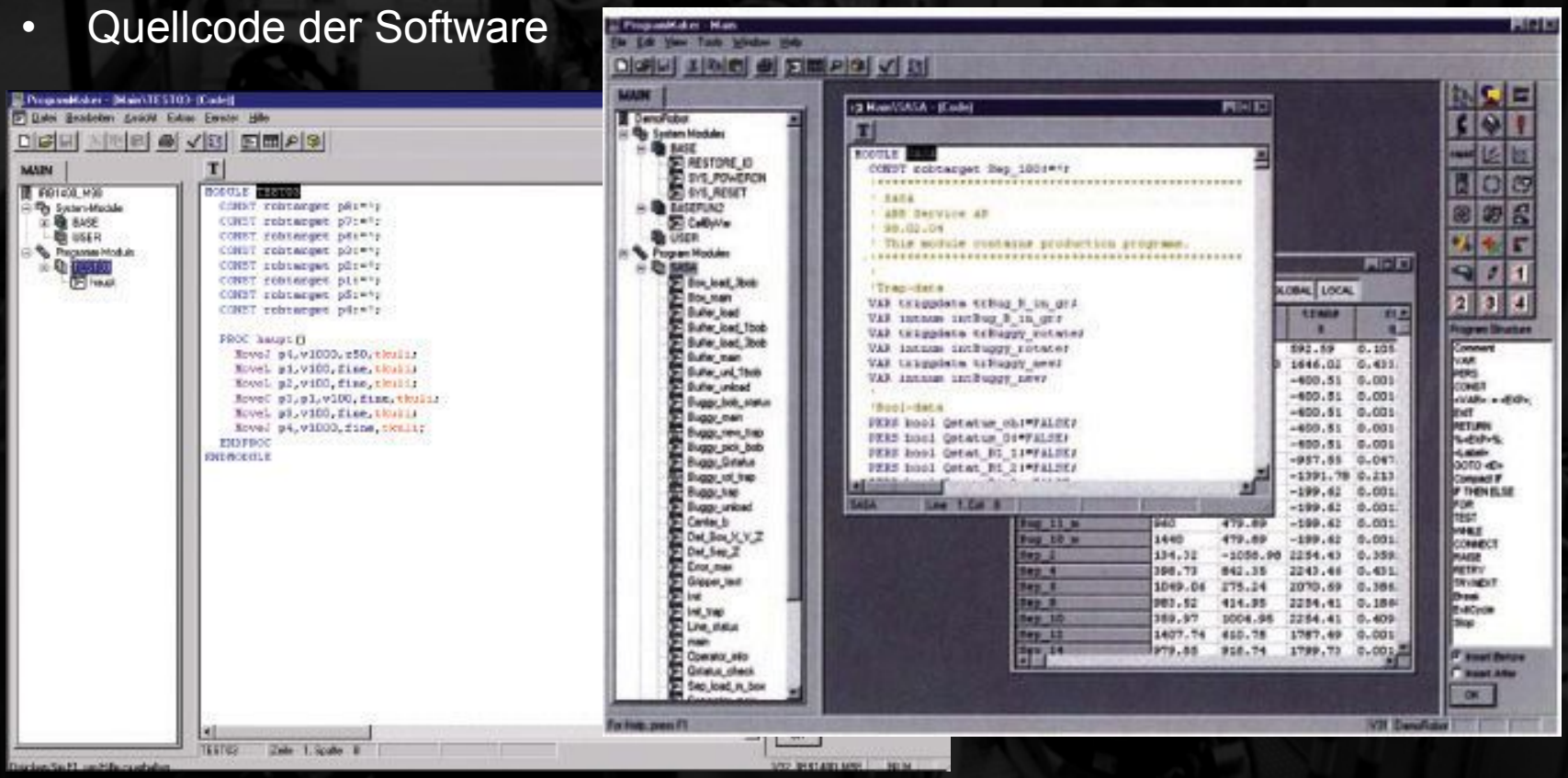


Die Programmierverfahren

- Was man unter Roboterprogrammierung versteht

Offline

- Editor (Ändern von Quelldateien)
- Quellcode der Software



Die Entwicklung

Ab wann von Roboterprogrammierung die Rede sein kann

→ Schon die Ägypter und Griechen entwickelten hoch komplizierte Maschinen, die teilweise wie Roboter automatisch fungierten („fixed automation“)

http://www.zdf.de/ZDFde/inhalt/0/0,1872,2051776,00.html
Heron, ein Tüftler aus der Antike (lebte vor etwa 2000 Jahren)
entwickelte unter anderem die erste heute bekannte Dampfturbine und
die ersten per „Trommelspeicher programmierbaren Roboter“, sowie schwere Kriegskatapulte und wohl auch eine Maschinenarmbrust

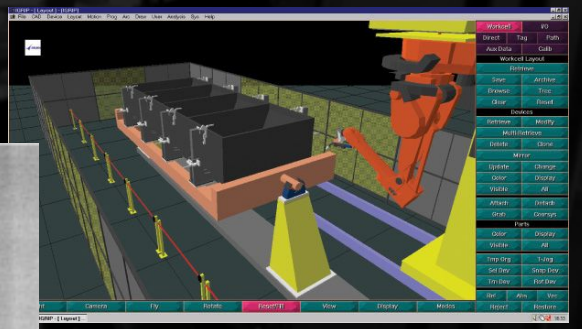


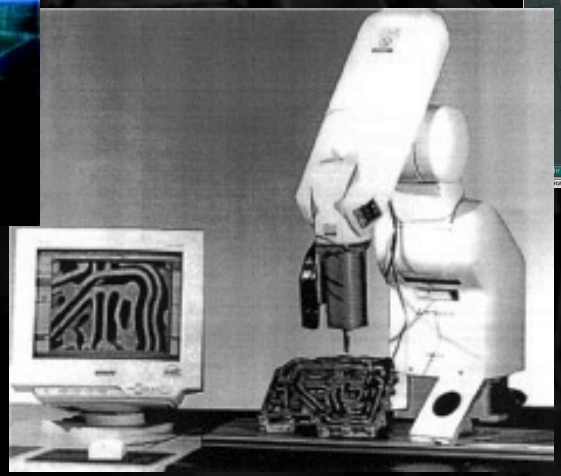


Die Entwicklung

Wo wir heute mit der Programmierung sind
und was sie geschaffen hat

- → Speziell angepasste Sprachen zur Implementierung komplizierter Befehlskonstrukte
- → Häufig grafische Schnittstelle
- → Beibringen im laufenden Betrieb durch Vorführen (Teach-In)
- → Sensorgestützte Programmierung
- → Virtual Reality

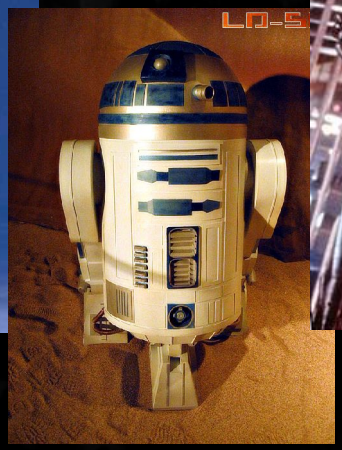


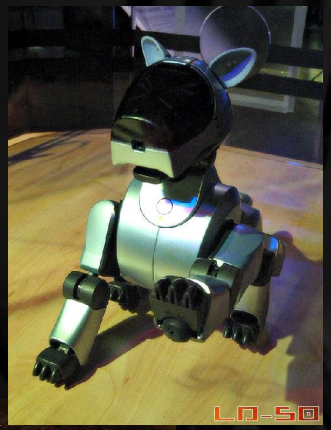


Die Entwicklung

Wohin sich die Roboterprogrammierung fortentwickelt
und möglicherweise „schon bald“ entwickeln wird

- → Befehle durch natürliche Sprache – Zeigen (Deuten), Reden
- → Intuition (Reflexe) – eigene Entscheidungen ohne strikte Vorgabe
- → Eigenständiges Lernen und Anwenden
- → Kommunikation und Organisation (Zielorientiert) untereinander





Programmiersprachen

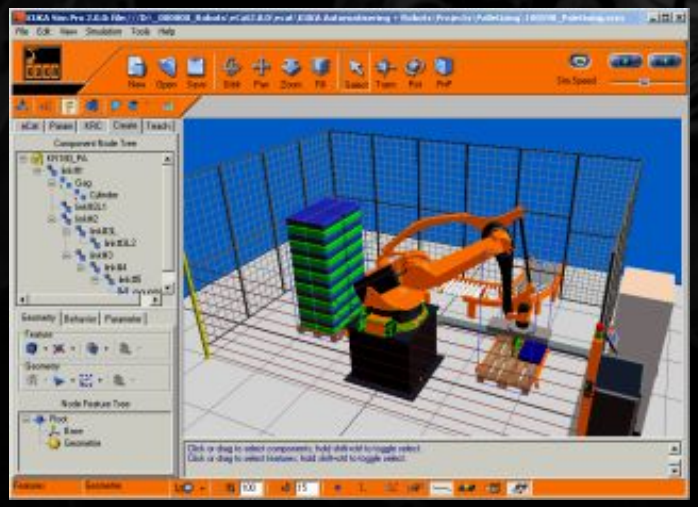
Eine Vielzahl von Sprachen:

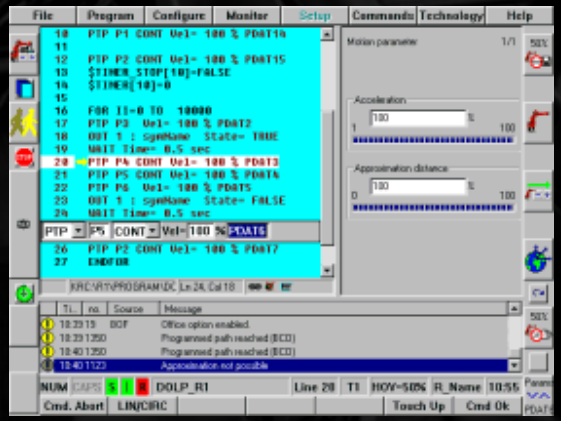
Um nur einige zu nennen:

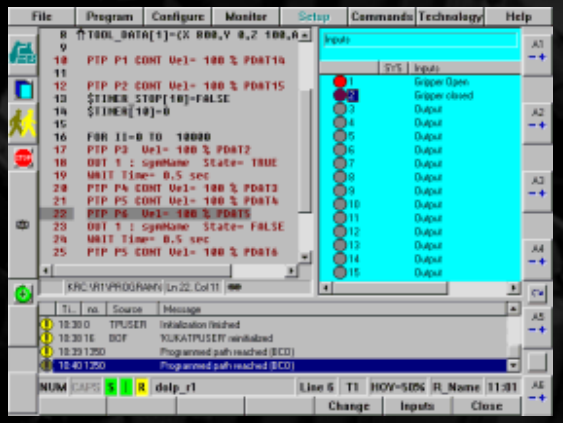
AL, AML, AR-BASIC, ARLA, BAPS, COSIMIR, IRL (IRDATA-Richtlinie), JARS, KAREL, PA-LIB, RAPID, ROBEX, SRL, TPE

Ein paar ausgewählte:

- NQC — für Kleinroboter zB. LEGO-MINDSTORM (von Dave Baum)
- VAL / VAL2 / VAL3 — von Staeubli (Industrieroboterhersteller)
- KRL — von KUKA (Europas größter Roboterhersteller)



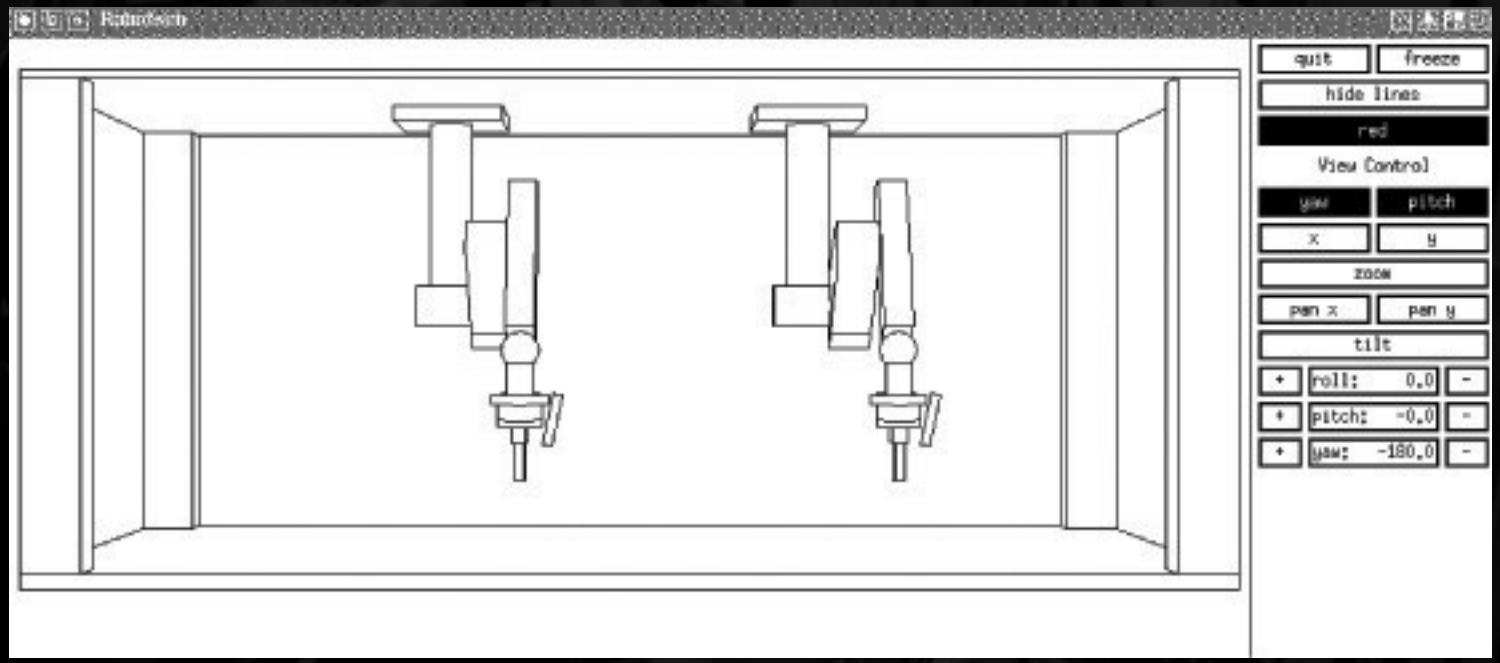




RCCL (Multi-RCCL)

Was ist RCCL?

- RCCL = Robot Control C Library
- C-Library zur Erstellung Aufgaben orientierter (task-level) Software in Unix
- Simulator zum Vorabtest der Ausführung



RCCL (Multi-RCCL)

Was ist RCCL?

- Originalversion entstanden 1983 durch Vincent Hayward (Purdue University)
- Seitdem starke Weiterentwicklung:
  zB Multi-Roboter & Multi-CPU – Unterstützung → Multi-RCCL (RCCL 3.0/4.0)
  (1988 Lloyd, Parker, McClain – McGill University & General Electric ATL)

- Freie Sprache außer für kommerzielle Nutzung
- Offline Programmierung – relativ einfache Handhabung („mächtiges Werkzeug“)
- Keine (kaum) Verwendung in Industrie und Unternehmen, da technisch inzwischen veraltet

RCCL (Multi-RCCL)

RCCL im Einsatz

- Forschungsinstitute
- Unis ☺ → Steuerungssoftware für den Arm mit integrierter Barretthand

→ Verweis: Vortrag über Echtzeitbetriebssysteme

- „Unser“ Roboter mit den vielen CPUs (Multi-CPU)
- Trajektory tasks laufen in Echtzeit

ggf. weiteres im Gespräch (Diskussion)

RCCL (Multi-RCCL)

Architektur – So funktioniert RCCL

- Hintergrund-Task (Bewegungsgenerator – trajectory generator)
  - Aufspaltung in trajectory tasks – einer pro Roboter (Multi-Roboter)
    - zB. mehrere Arme als einzelene Roboter
    - Verteilung auf vorhandene CPUs (Multi-CPU)
  - Für den Programmierer nicht weiter sichtbar (da in Library)
    - Servo-Level-Algorithmen (gekapselt) – eigene Implementierungen nicht gedacht
      - Kann zu Kollisions- und Verrenkungsproblemen führen
  - Steuerung durch spezielle Funktionen und Datenstrukturen
    - Festlegung von Koordinaten oder Vektoren

- „Planungs“-Task
  - Vom Programmierer mit RCCL geschriebenes, C-basiertes UNIX Programm
    - Erzeugt ggf. die Benutzerschnittstelle (Grafische Oberfläche)
    - Beinhaltet ggf. spezielle Problembezogene Befehle (wie: „verschweiße A mit B“)
    - Direkte Implementierung aller Kommandos üblich → Test über in RCCL integrierten Simulatorbetrieb (Robotsim)!
    - Interne Verwendung der speziellen RCCL-Funktionen und Datenstruktren
    - Kommunikation mit den trajectory-Tasks über Shared Memory

RCCL (Multi-RCCL)

Ein Beispiel:

```
#include <rccl.h>                                   // RCCL-Library
#include “manex.560.h“                              // spezielle Deklarationen für den
                                                    // PUMA 560 - Roboter
main()                                              // Beginn der Hauptroutine
{
    TRSF_PTR p, t;                                  // Transformationsmatrizen
    POS_PTR pos;                                    // Zielposition
    MANIP *mnp;                                     // aktuelle Roboterposition/-eigenschaften
    JNTS rcclpark;                                  // Initialposition und -parameter
    char *robotName                                 // Roboterbezeichnung

    rcclSetOptions(RCCL_ERROR_EXIT);                // Option damit das Programm nur bei
                                                    // Laufzeitfehlern abgebrochen wird
    robotName = getDefaultRobot();
    if(!getRobotPosition(rcclpark.v, “rcclpark“, robotName)) // Falls für den Standardroboter keine
    {   printf(“position ‘rcclpark‘ not defined for robot\n“);   // Initialposition ermittelt werden kann,
        exit(-1);   }                               // gib Fehlermeldung aus und beende!
```

RCCL (Multi-RCCL)

Ein Beispiel:

```
t = allocTransXyz("T", UNDEF, -300.0, 0.0, 75.0);          // relative Zielkoordinaten
p = allocTransRot("P", UNDEF, P_X, P_Y, P_Z, xunit, 180.0); // Rotationsmatrix auf Starteinst.
pos = makePosition("pos", T6, EQ, p, t, NULL);             // „kinematische
Positionsgleichung"

mnp = rcclCreate(robotName, 0);                            // initialisiere Trajekory Generator
rcclStart();                                               // startet Trajektory-Task

movej(mnp, &rcclpark);          // Bewegung zur bekannten Initialposition
setMod(mnp, 'c');               // geradlinige kartesische Bewegung
move(mnp, pos);                 // Bewegung zur Zielposition
stop(mnp, 1000.0);              // mache eine Sekunde Pause
movej(mnp, &rcclpark);          // Bewegung zurück zur Startposition
stop(mnp, 1000.0);              // mache eine Sekunde Pause

waitForCompleted(mnp);          // sicherstellen, daß alle Bewegungen abgeschlossen sind
rcclRelease(YES);               // beende Trajektory Generator und schalte Roboter ab
```

Diskussion

<u>Diskussionsideen:</u>

- Ein Job für die Zukunft?
  → Bleibt da überhaupt noch Arbeit für andere übrig?

- Wo liegen die Grenzen der Roboterprogrammierung?
  → Gibt es diese überhaupt?
  → Alles eine Frage der KI?
  → Roboter als fühlende Wesen?
  → Sich selbst programmierende Roboter?
  → Roboter als erfinderische Künstler?
  → Wird es noch einen Unterschied zwischen Mensch und Maschine geben?

- Werden Menschen überhaupt noch gebraucht werden?
  → Roboter als willenslose Arbeitskräfte?
  → Denkende Maschinen? (Eine neue Rasse - SciFi)

Achtung! → Vortrag: Ausblick in die Zukunft

Quellen

- Internet:
  - www.iof.mw.tu-dresden.de/vorlesung/ 000301020401/05_mrb03_Programmierung_Flemming.pdf
  - www.ba-eisenach.de/fileadmin/ _temp_/dokumente/stura/scripte/sem4/rob07.pdf
  - http://de.geocities.com/gehnio/Robotik.pdf
  - http://www.google.de (für allgemeine Recherche und Bilder)
  - http://www.kuka.com/germany/de
  - http://www.staeubli.de
  - nicht weiter verwendet, aber für Lego-Mindstorm-Interessierte:
    http://www.fh-landshut.de/~gschied/robotic/index.html
    http://bricxcc.sourceforge.net/nqc/
  - außerdem interessant: http://www.orocos.org/ **O**pen **Ro**bot **Co**ntrol **S**ystem
- Buch:
  - Introduction to Robotics (John J. Craig)
- Weitere Quellen:
  - Multi-RCCL User‘s Guide (John Lloyd und Vincent Hayward 1992)