Robotics research in Odense

Henrik Gordon Petersen
Professor, Head of Section, SDU Robotics
Maersk Mc-Kinney Moller Institute
University of Southern Denmark (Odense)

SDU Robotics

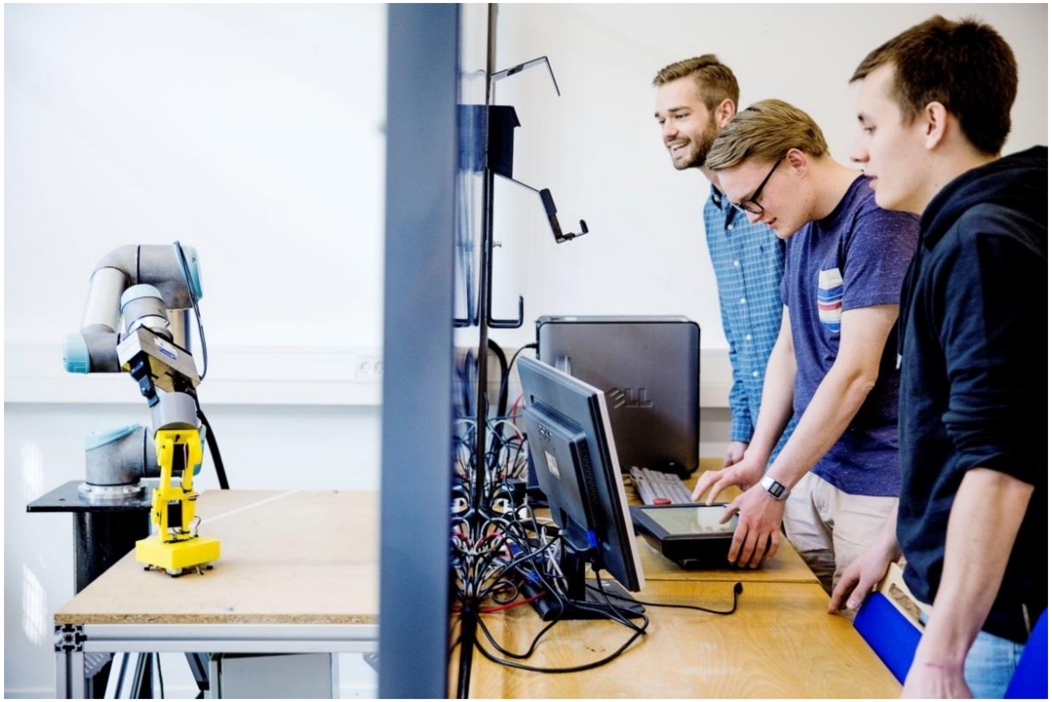
Education

Robot systems Engineering
M.Sc. (2 years)
(Intake 2019: 60 students),

BSc (MSc) (3 (5) years)
(Intake 2019: 60 students)
Cutoff: 8.0

B.Eng.
(Intake 2018: 90 students)
Total: Around 210 new students

Welfare Technology
M.Sc., B.Sc.
(Intake: around 30 students)



Research

Robot Control

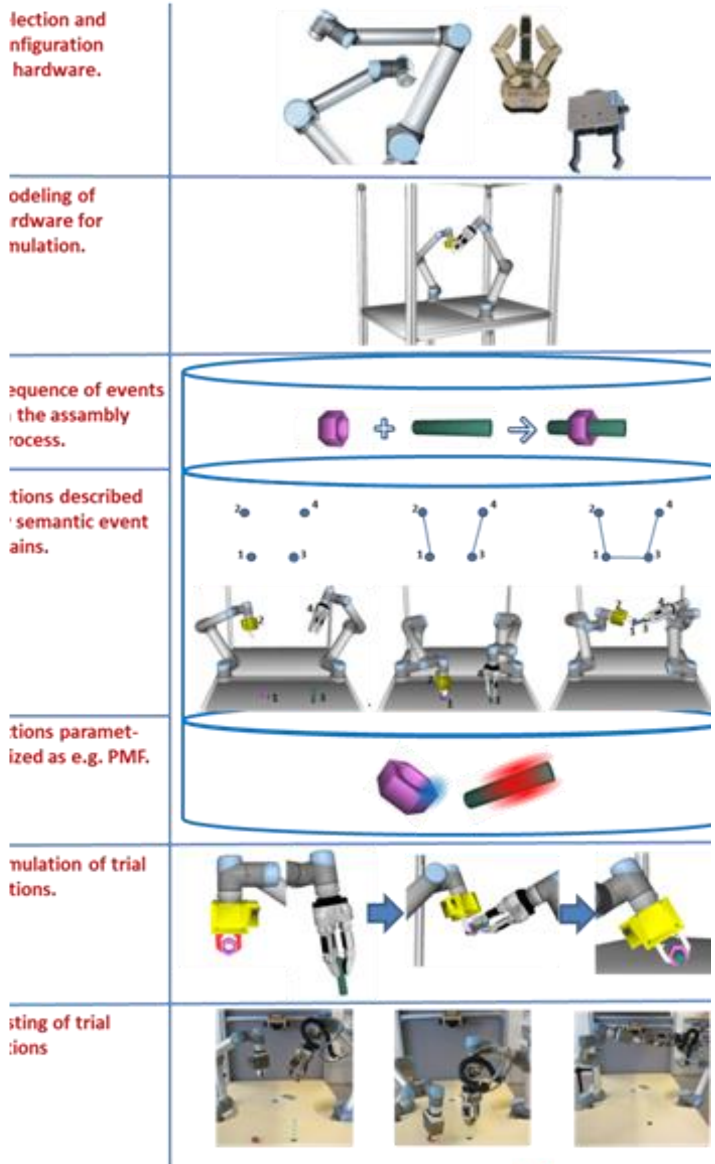
- Kinodynamic modelling and planning
- Force-based control
- Process Optimization

Computer Vision

- Object recognition and Pose estimation
- Optical issues in Computer Vision
- Inspection (NEW)

Model Based Robot Systems Engineering

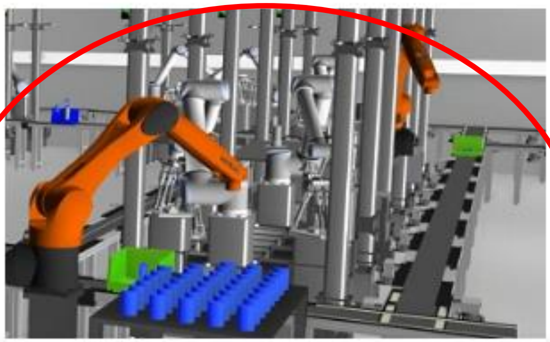
- Digital Twins
- Visual Programming
- Co-simulation with MATLAB, ROS, …



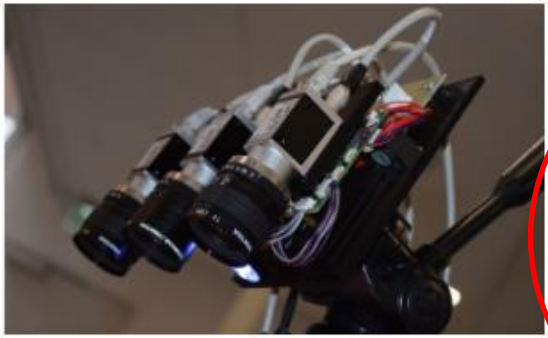
Robotic Solutions for Industrial Applications

Welfare Robotics

Surgical Robotics



Mathematical Modelling and Simulation of Robotic Systems and Processes



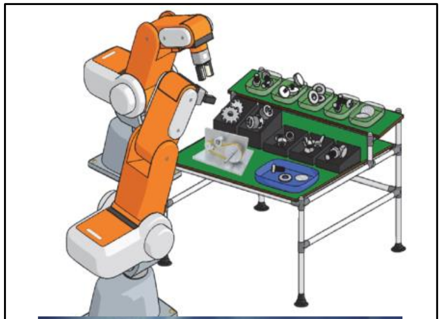
Computer Vision based Object Recognition and Pose Estimation

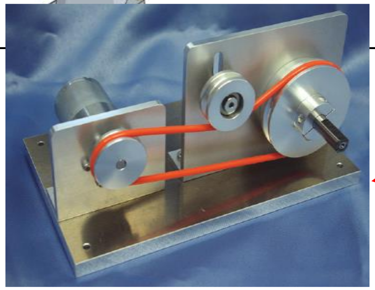
Model and Learning-based Control of Robotic Systems

Industrial Robotics
World Robotic Challenge

SDU Robotics wins the "unofficial World Cup" in industrial robotics

- 4 day competition in Tokyo in October 2018
- 16 teams selected from 250 applicants
- Relevant industrial assembly challenges



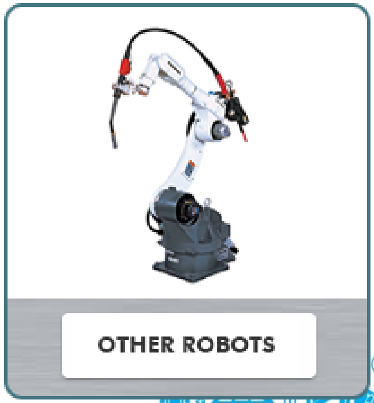


History

- Unimate 1956-70
  - Hydraulic

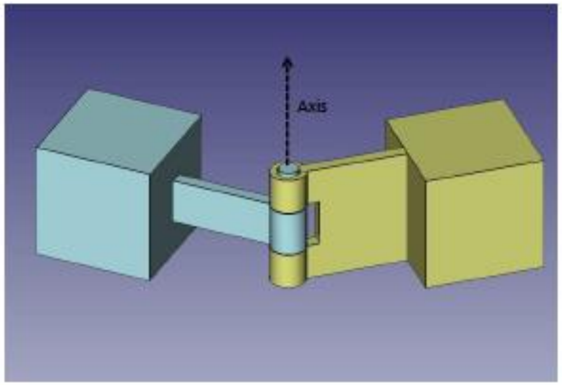
- 1970's
  - Electric
  - Up to 6 degrees of motion

Pictures of different robots

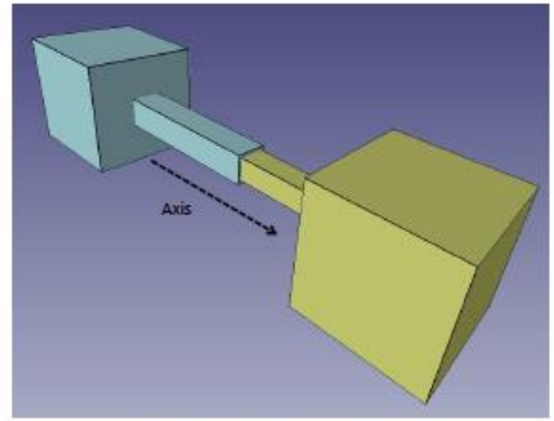


The next 23 slides will illustrate why an industrial robot manipulator arm is just a stupid machine that is easy to use as any other tool…

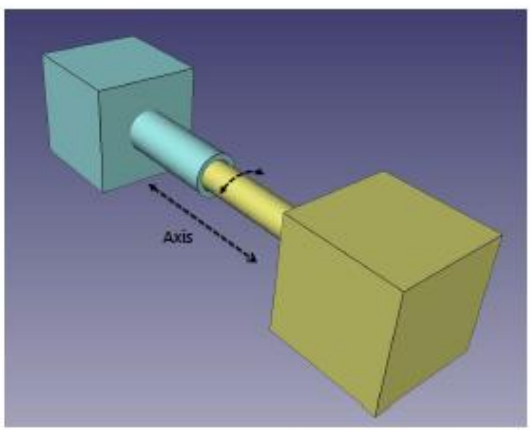
Robot joints



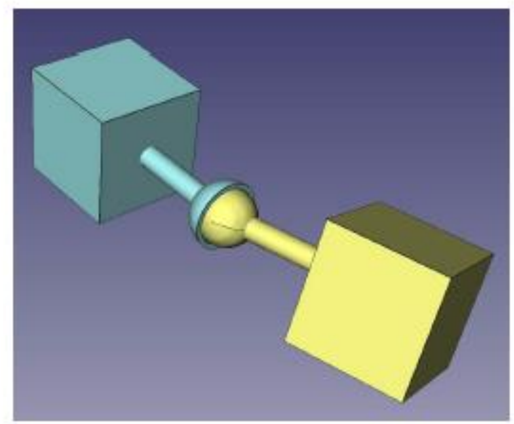
(a) 1 DOF Revolute joint.



(b) 1 DOF Prismatic joint.



(c) 2 DOF Cylindrical joint

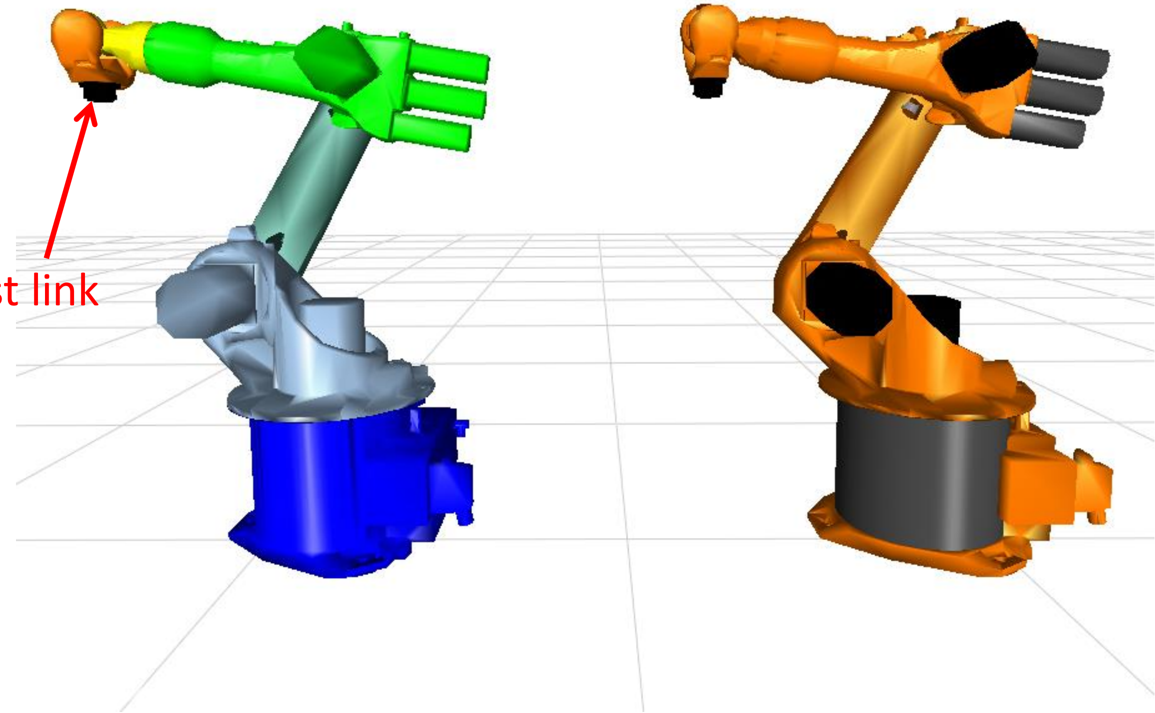


(d) 3 DOF Spherical joint

Figure 2.1: Joint types.

Robot links

Links marked with different colours



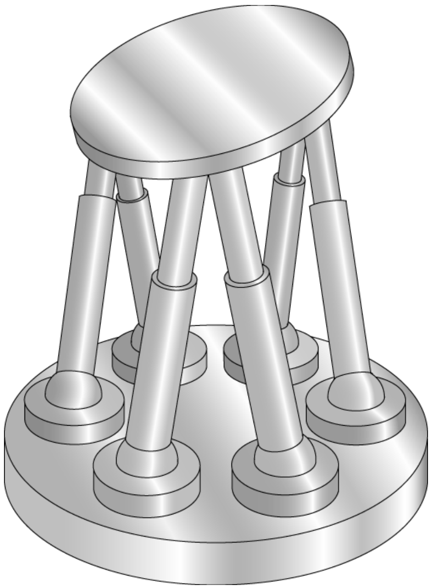
Outermost link is almost invisible

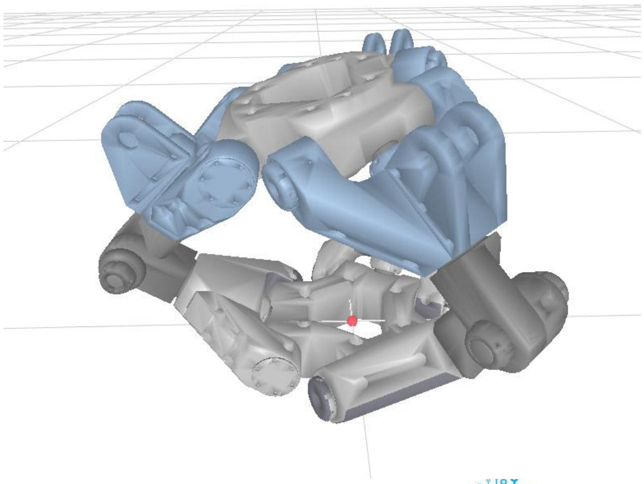
Constraints on Actuation and Topology

- The robot consists of a set of links and joints connected into a single structure
- Any two links are connected by at most one joint.
- Some, but not necessarily all, joints are motorized
- Each motorized joint can move independent of the other motorized joints.
- If all motorized joints are fixed at a given position, the robot will be kinematically stiff (typical case). Otherwise the robot is *underactuated*.

Robots with Closed Loop Topology

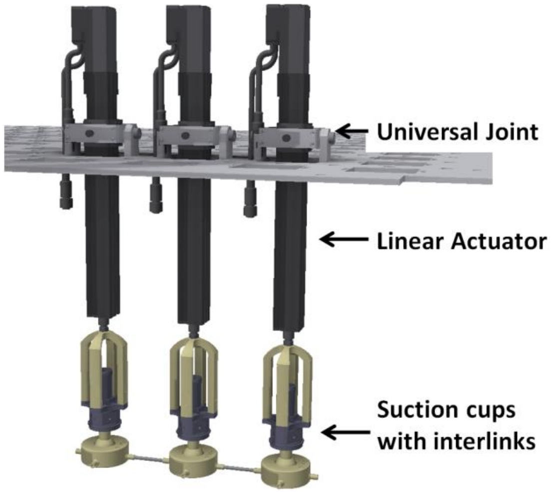
- The robot consists of a set of links and joints connected into a single structure

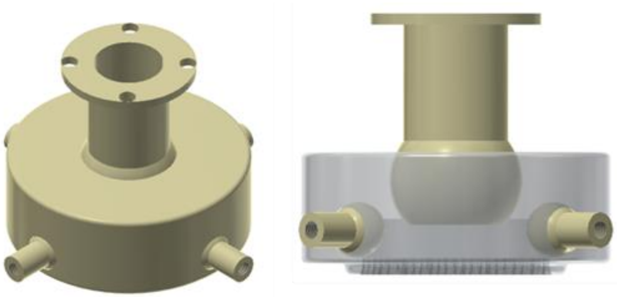




Palletizing robot with 4-bar linkages

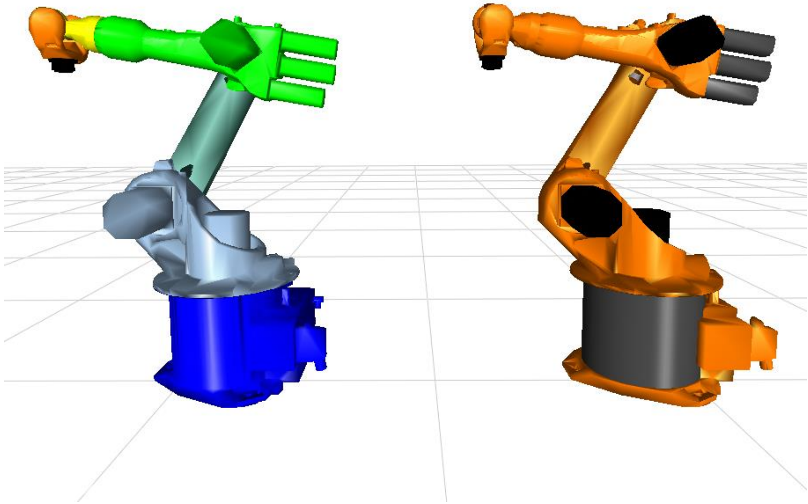
Underactuated Robot





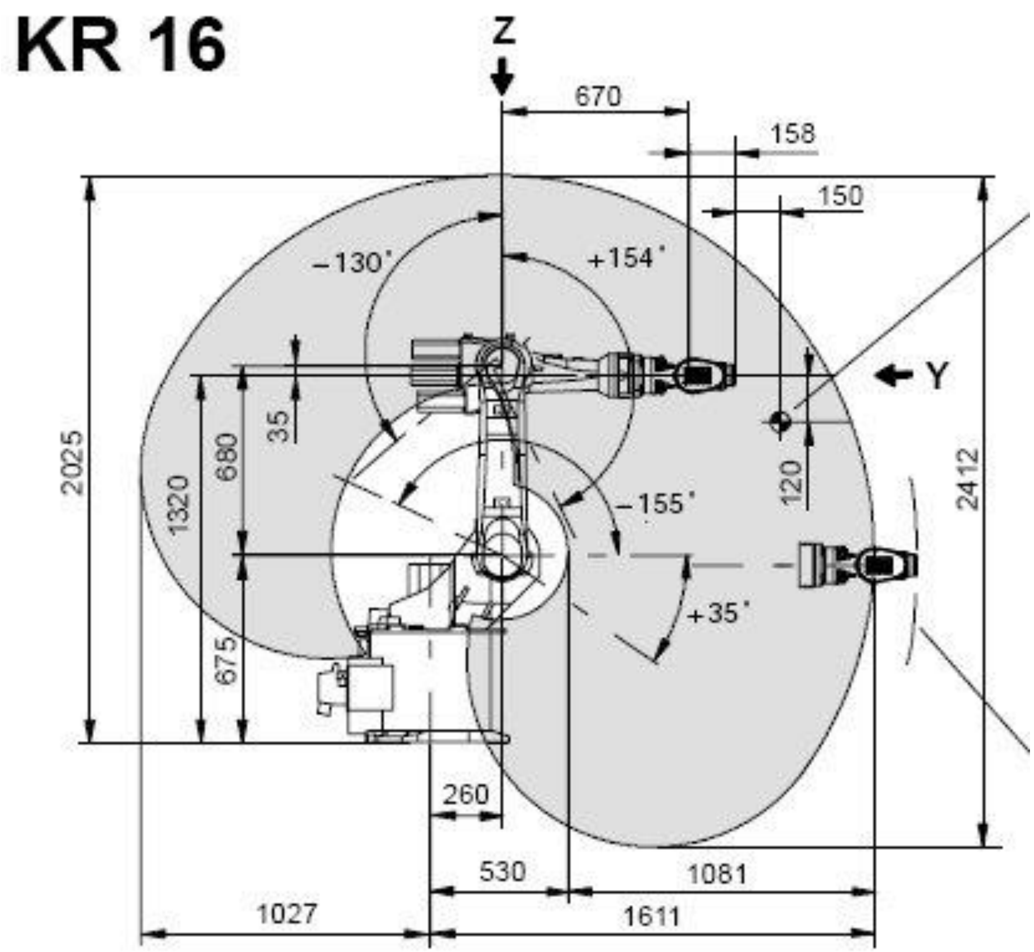
Kuka KR16

How do we describe the kinematics ?

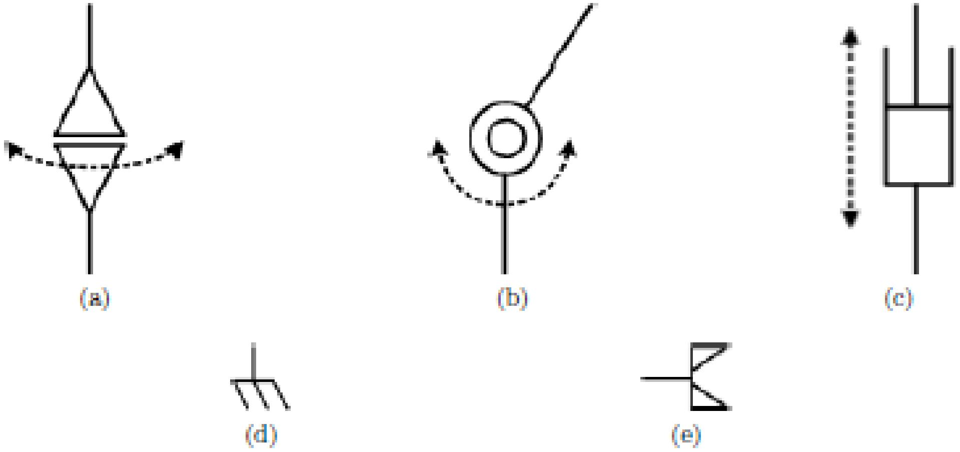


Kuka Kr16

Typical drawings

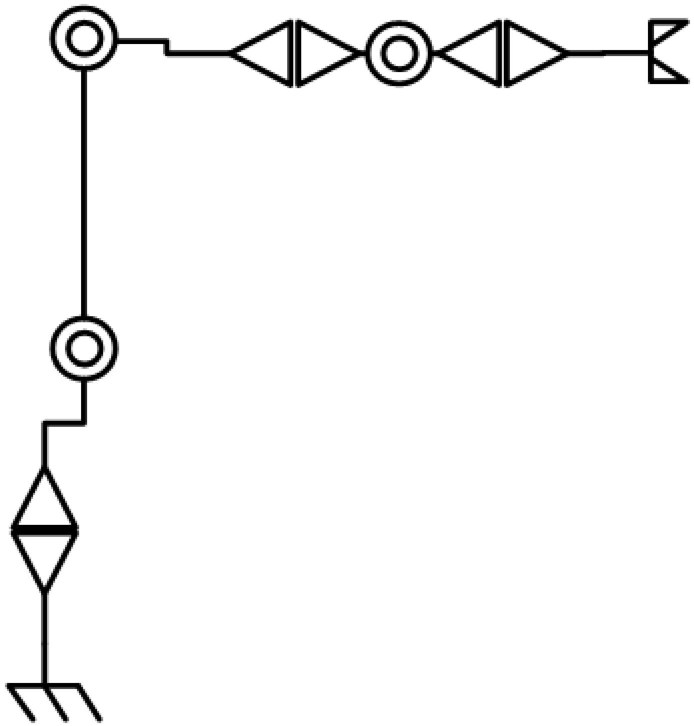


Joints, base and tool



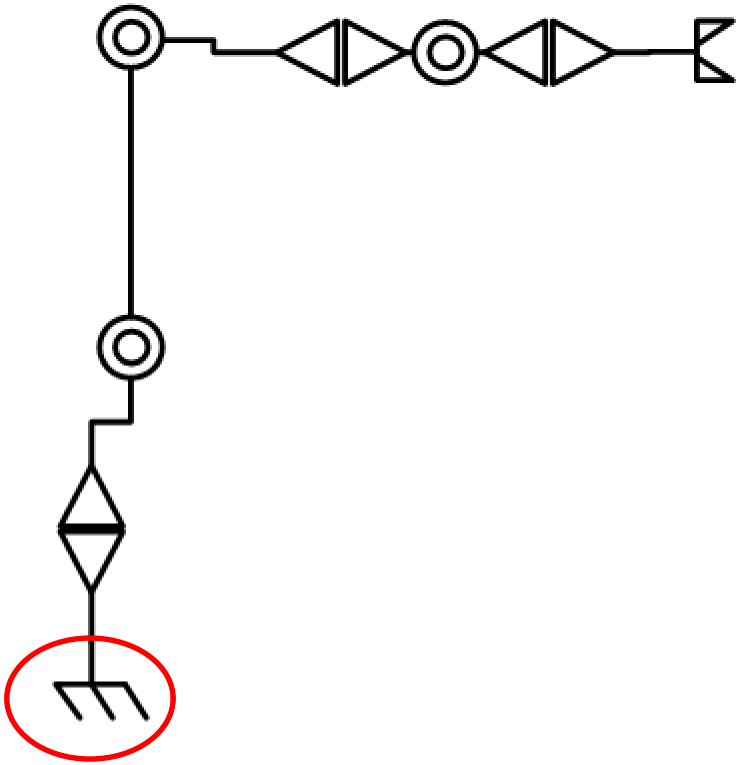
a): Revolute joint (vertical rotation axis)
b): Revolute joint (rotation axis perpendicular to drawing plane)
c): Prismatic (translational) joint
d) Robot base
e) Robot tool

Schematic drawing of the Kuka KR16



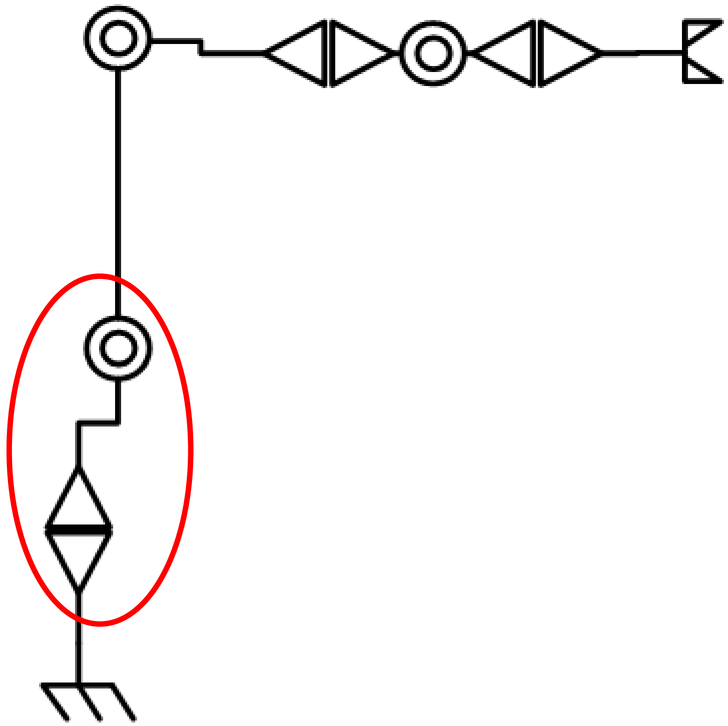
Schematic drawing of the Kuka KR16

(Robot) Base



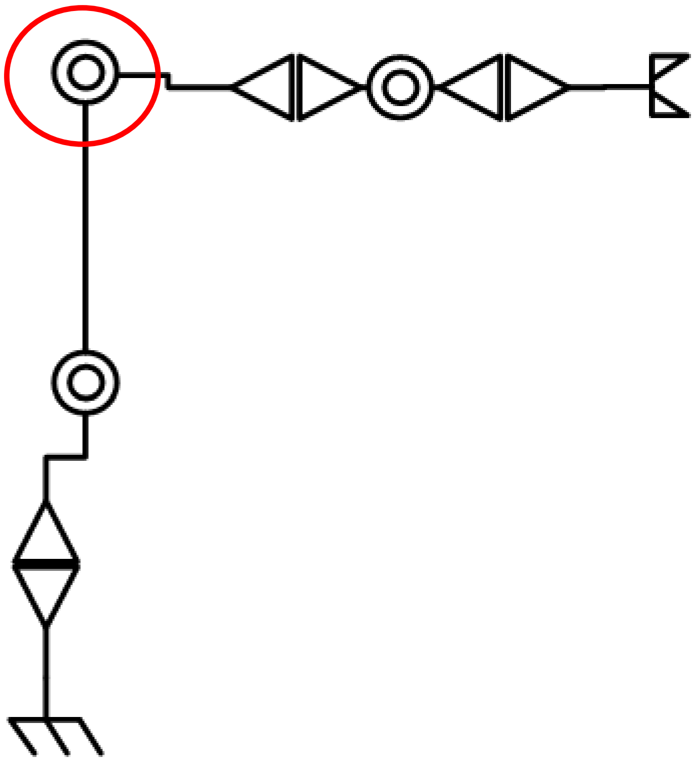
Schematic drawing of the Kuka KR16

Shoulder joints (2)



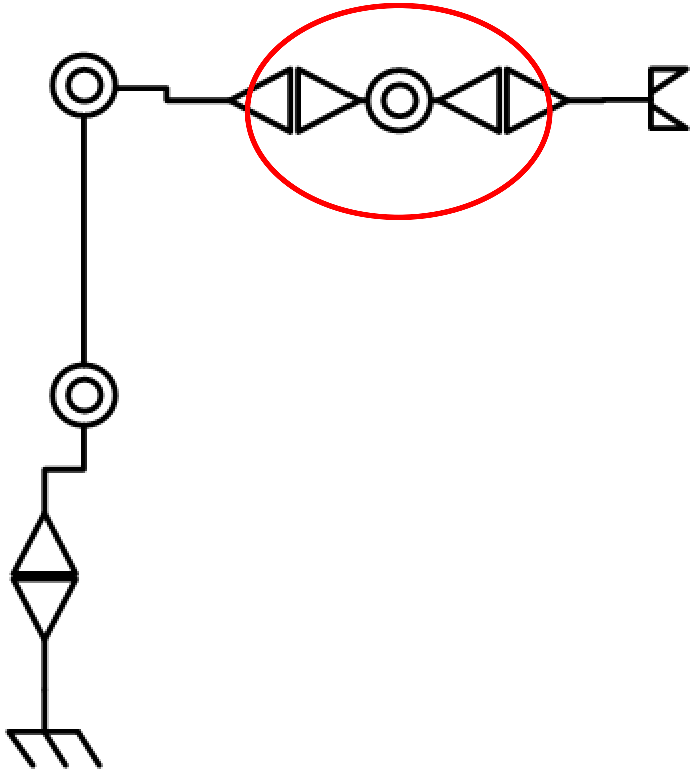
Schematic drawing of the Kuka KR16

Elbow joint (1)



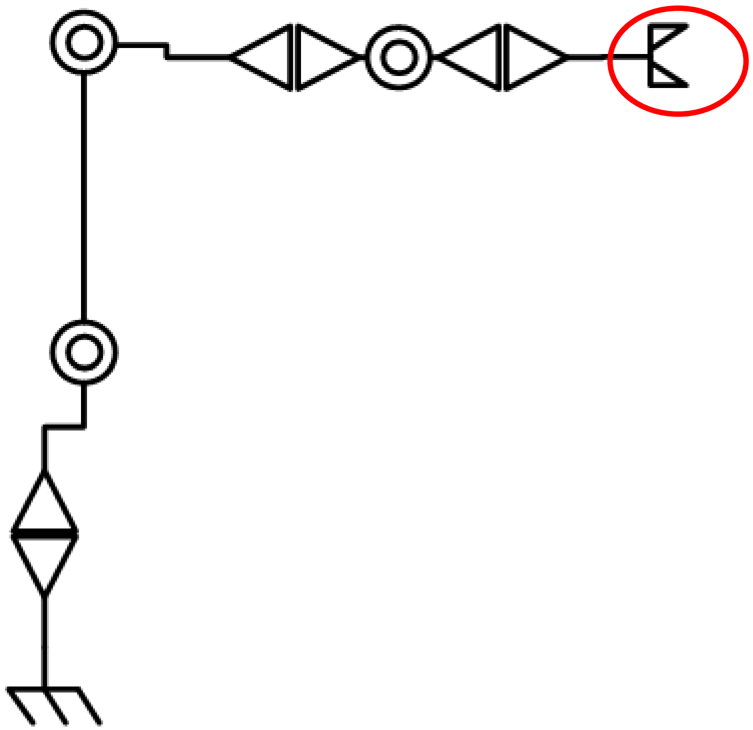
Schematic drawing of the Kuka KR16

Spherical wrist (3)

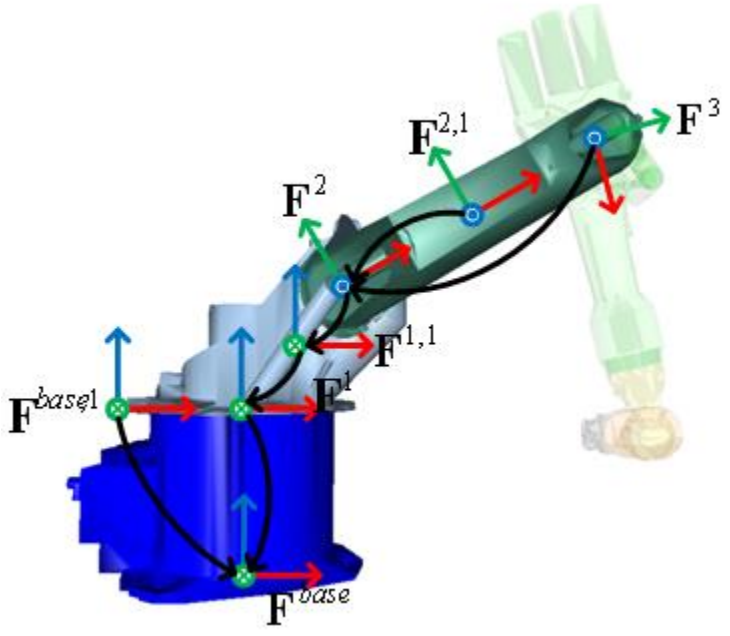


Schematic drawing of the Kuka KR16

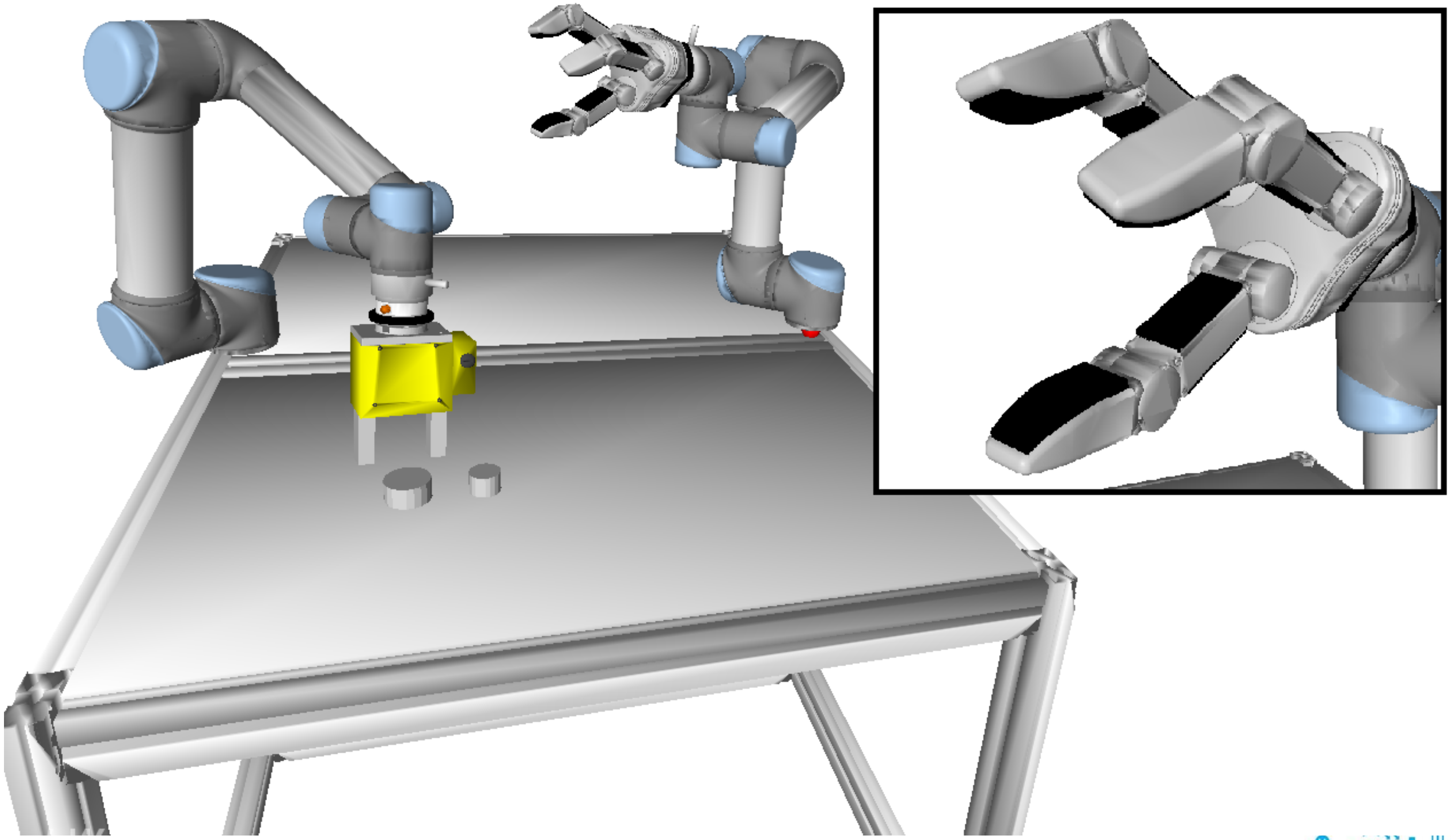
Tool



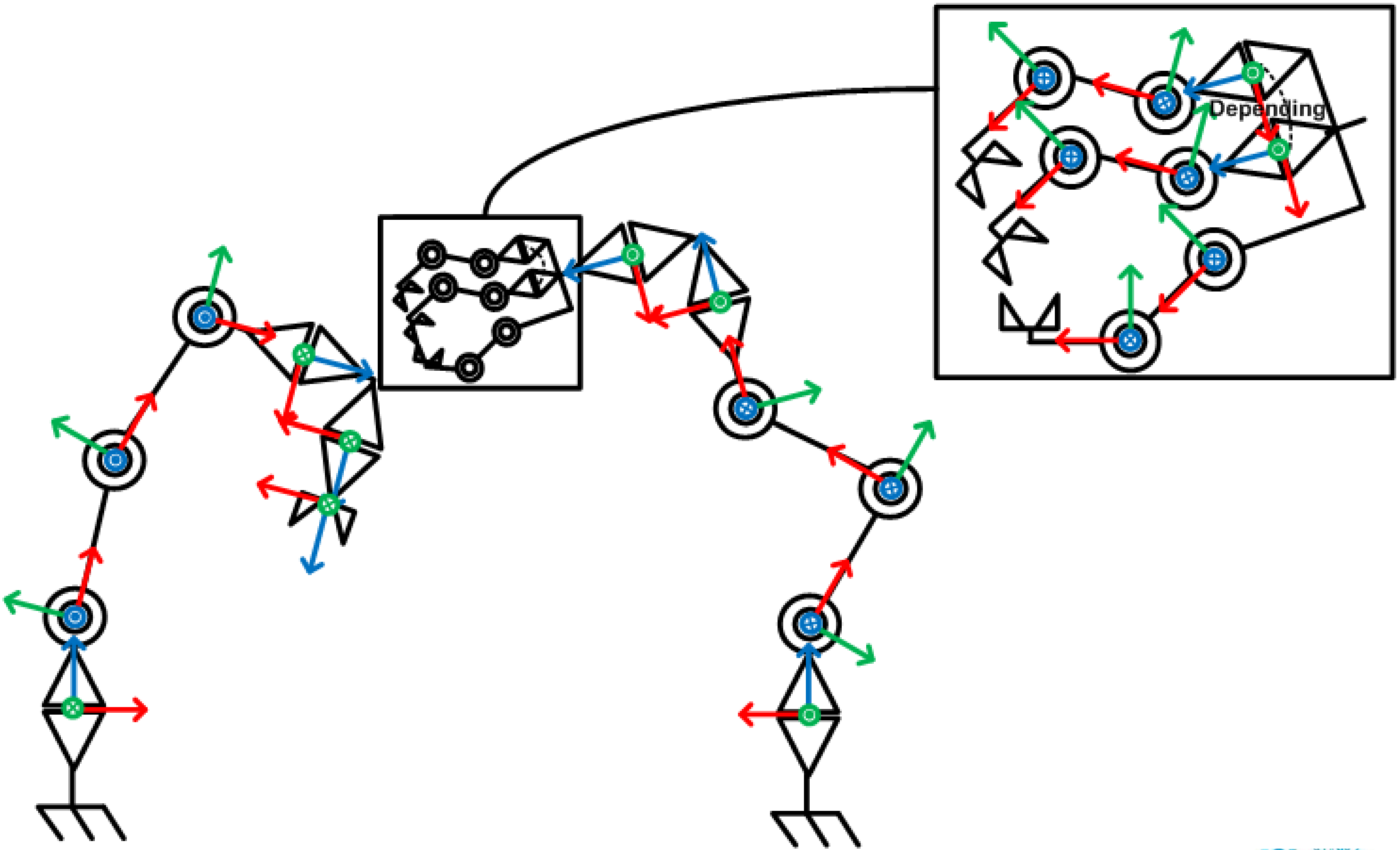
Frames on a robot



A two robot work cell

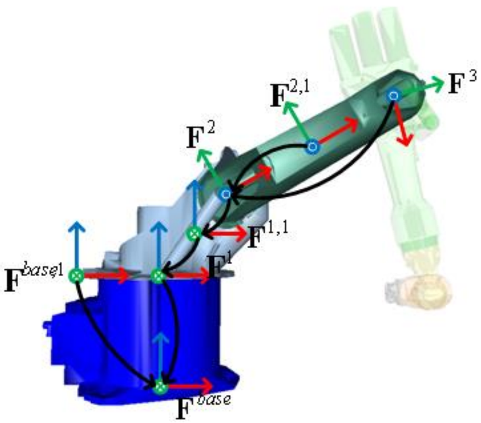


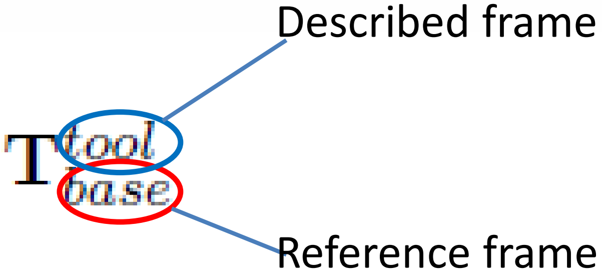
Frames in a work cell



Homogeneous Transformation

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{p} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$





Joint vector $\mathbf{q}$

- The *joint vector* is defined *as the set of all joint variables of all robotic devices* in a scene
- To simplify, we will from now on only consider a single serial robot in a scene.
- The joint vector uniquely determines the position and orientation of all robot links

Tool frame and Forward Kinematics

The *tool frame* is a special frame fixed with respect to the outermost link on the robot.

It defines the position and orientation of the active site on the robot (for simplicity we assume that there is only one active site).

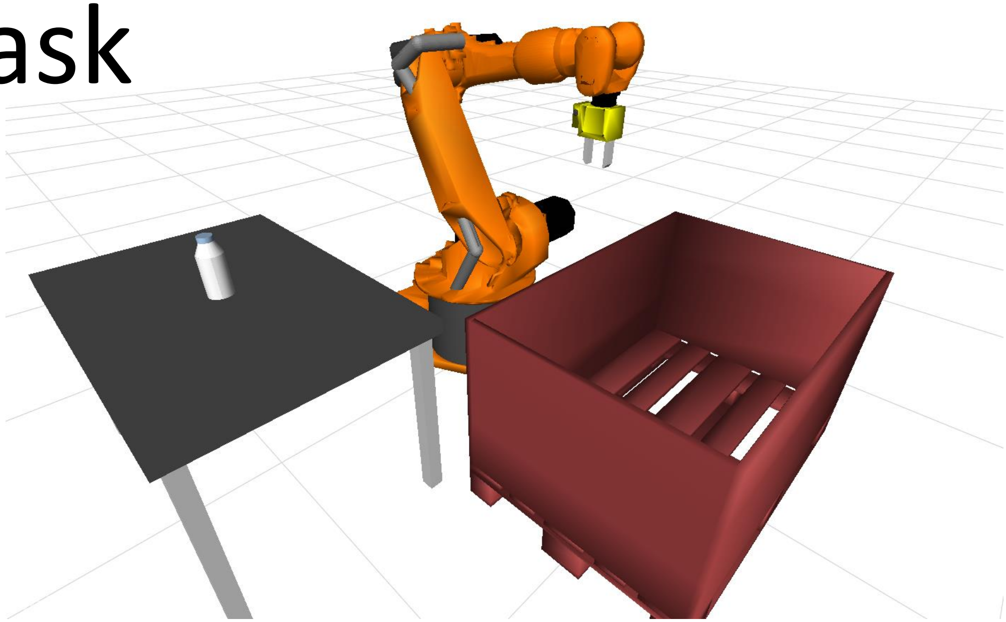
We denote it by the homogeneous transformation : $\mathbf{T}_{base}^{tool}$

Forward kinematics is the task of computing the tool frame as function of the joint vector, denoted as $\mathbf{T}_{base}^{tool}(\mathbf{q})$

The task is accomplished by straightforward use of matrix algebra

$$\mathbf{T}_{base}^{tool} = \mathbf{T}_{base}^{1}\mathbf{T}_{1}^{2}\ldots\mathbf{T}_{n-1}^{n}\mathbf{T}_{n}^{tool}$$

Typical robot task



Move F^{1a} //Collision free move to frame **F**^{1a} where the tool is adjacent to the bottle

Move Linear to frame F^{1b} //Move the tool in a straight line to the bottle

Close Gripper //Pick up the bottle

Move Linear to frame F^{1a} //Move back to frame **F**^{1a}

Move F^{2a} // Colllision free move to frame **F**^{2a} adjacent to the place position at the table

Move Linear to frame F^{2b} //Move the tool in a straight line to the place position

Open Gripper //Release the bottle

Move Linear to frame F^{2a} //Move back to frame **F**^{2a}

Move Home //Move back to the *Home* configuration

Inverse Kinematics

Desired transformation $[\mathbf{T}_{base}^{tool}]_{desired}$

Find **a joint vector** $\mathbf{q}$ so that $\mathbf{T}_{base}^{tool}(\mathbf{q}) = [\mathbf{T}_{base}^{tool}]_{desired}$

Nonlinear root finding problem.
Numerical approach – for any serial robot. Good initial guess required

Find **all joint vectors** $\mathbf{q}$ so that $\mathbf{T}_{base}^{tool}(\mathbf{q}) = [\mathbf{T}_{base}^{tool}]_{desired}$

Analytical approach – for serial robots with special (idealized) kinematics.

The previous 23 slides has illustrated why an industrial robot manipulator arm is just a stupid machine that is easy to use as any other tool…

Challenges in industrial robotics are related to describing and optimizing the process

- Modelling the process
- Use the model to derive a desired robot tool path
- Complete the solution by inverse kinematics

Why work with industrial robotics ?

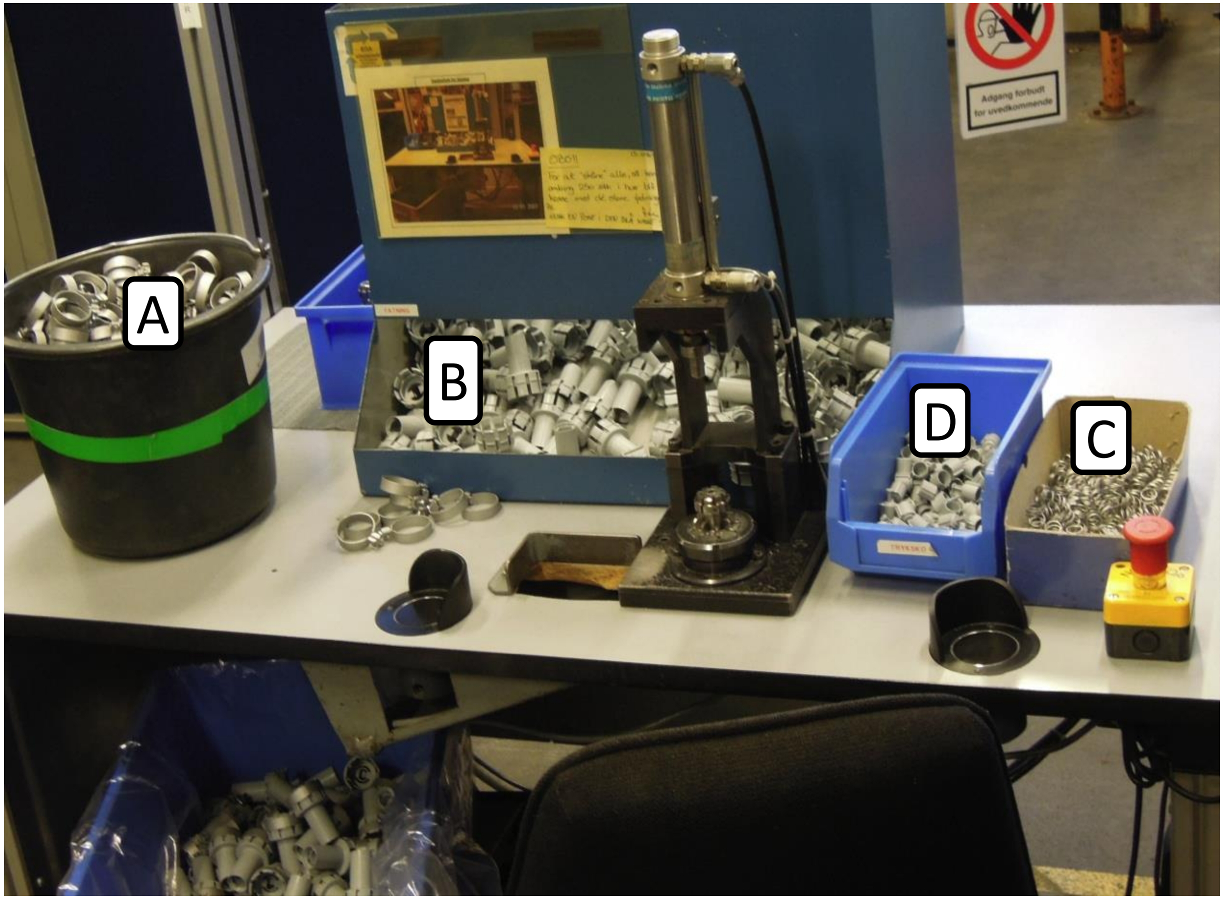
- Human workers
  - Movable so that they can go to where the work in the shopfloor is
  - Fast start-up (hours for new tasks, immediately for shifts between existing tasks)
  - Humans quickly learn how to improve the performance based on experience
  - Easy adaptable to task modifications

- Traditional automated solutions
  - Static big facilities with fences and a lot of integrated machinery
  - Slow start-up (typically in the range of 3-24 months for new tasks)
  - Improvement based on experience is basically non-existing
  - Difficult to adapt to task modifications
  - (Much) too expensive, i.e. too long payback time except for ”long term high runners”

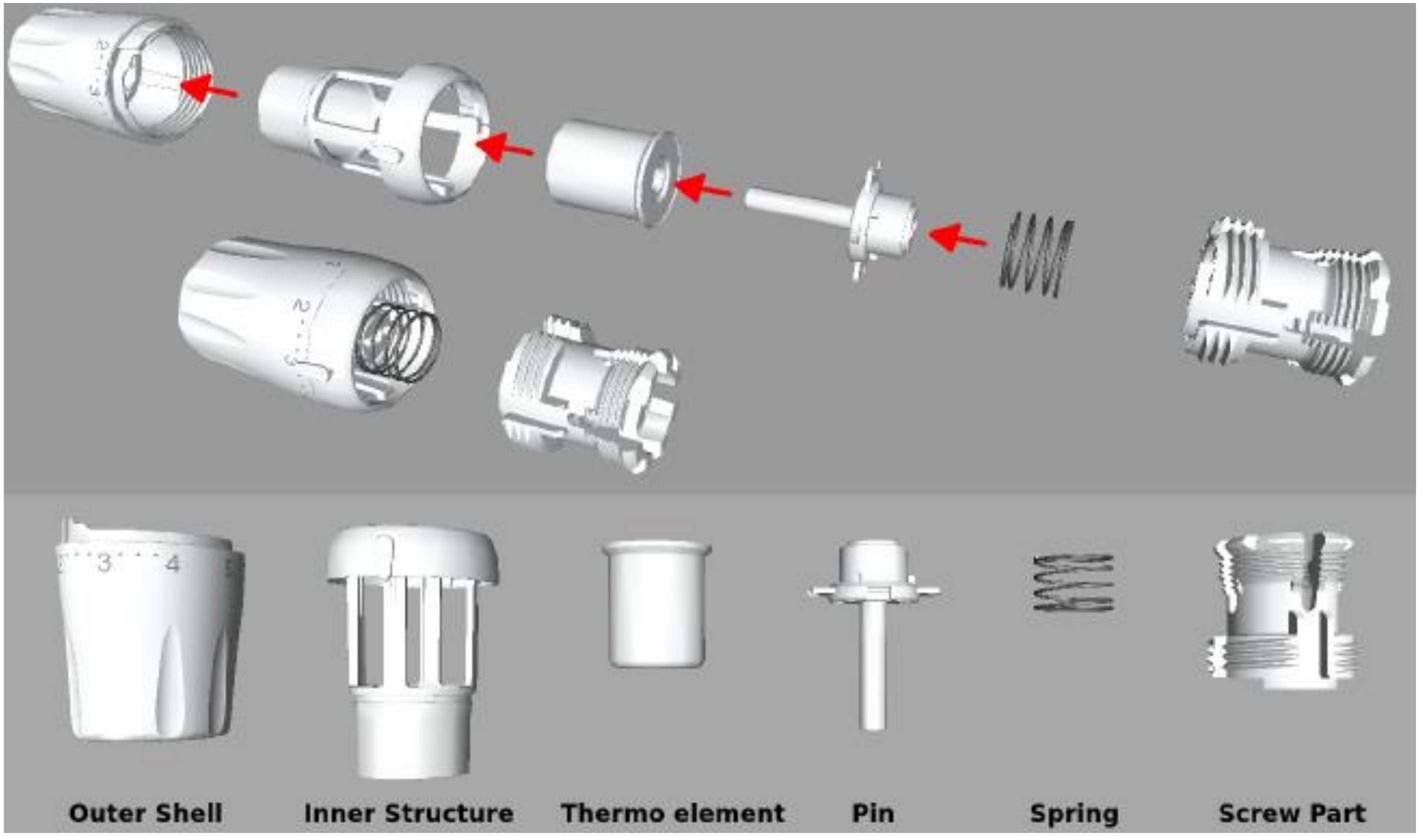
Traditional robot automation
Complex, but repetitive

Traditional robot automation
Non-repetitive, but simple

Trend in robotic automation
Non-repetitive and (somewhat) complex

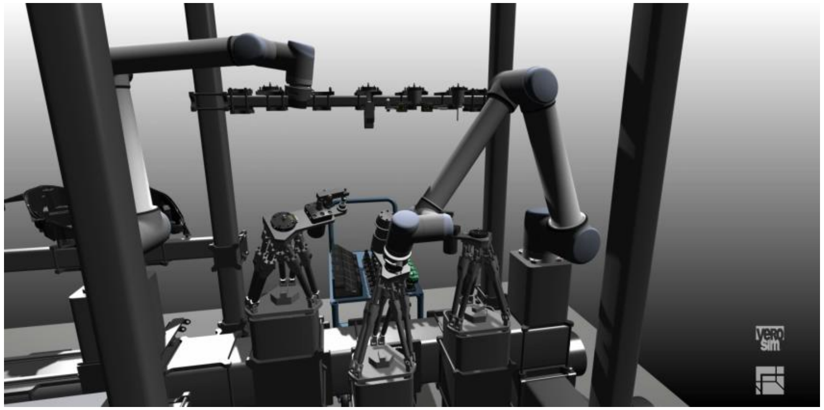


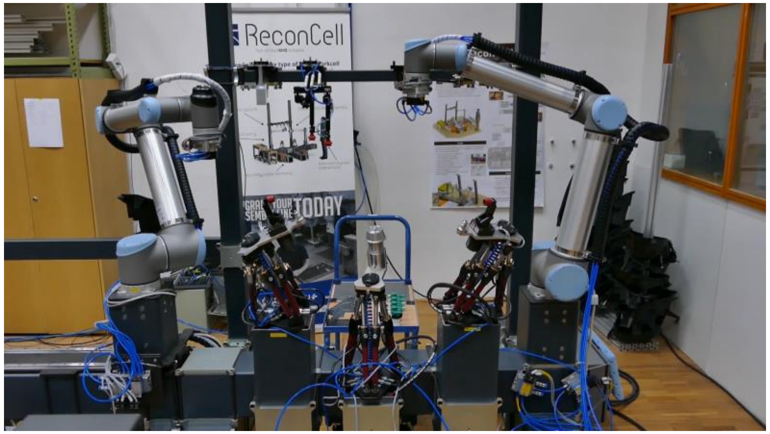
Trend in robotic automation
Robot programming based on digital models



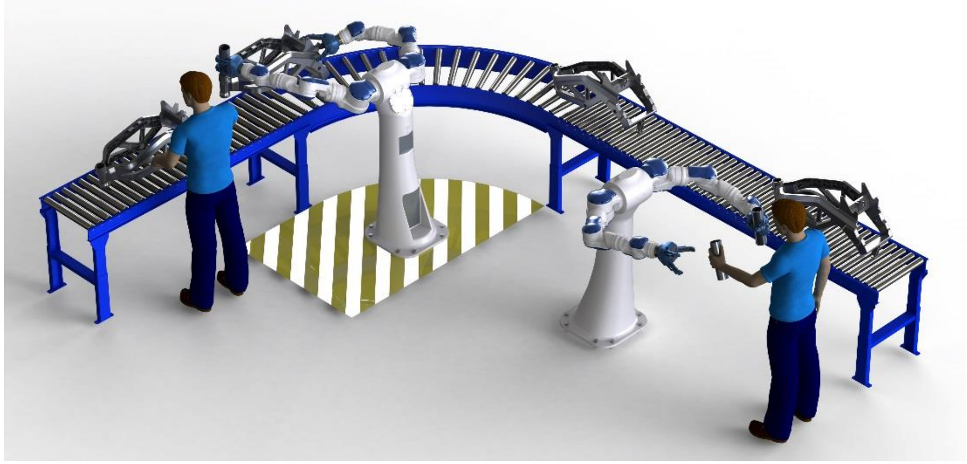
Programming and testing in virtual environments

Example from one of our European projects:
Headlight assembly for automotive

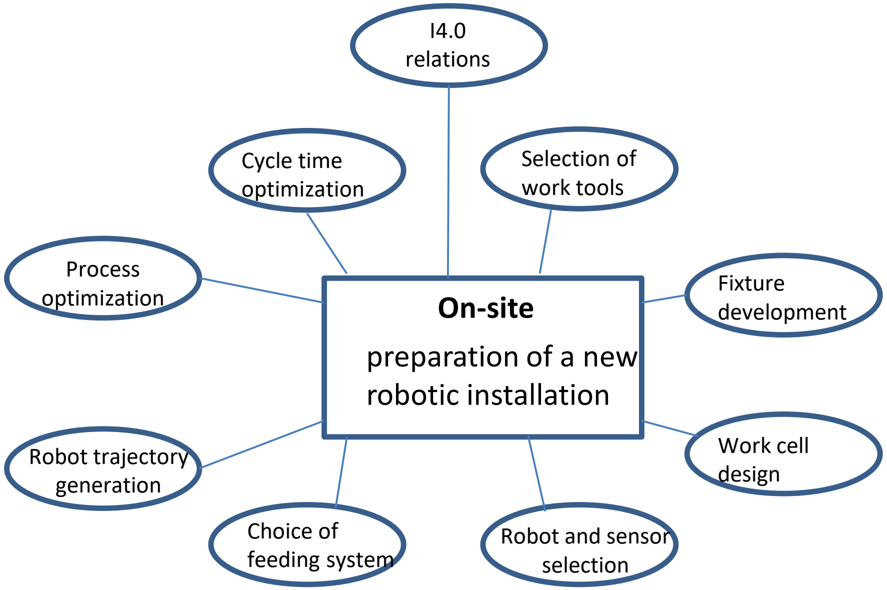




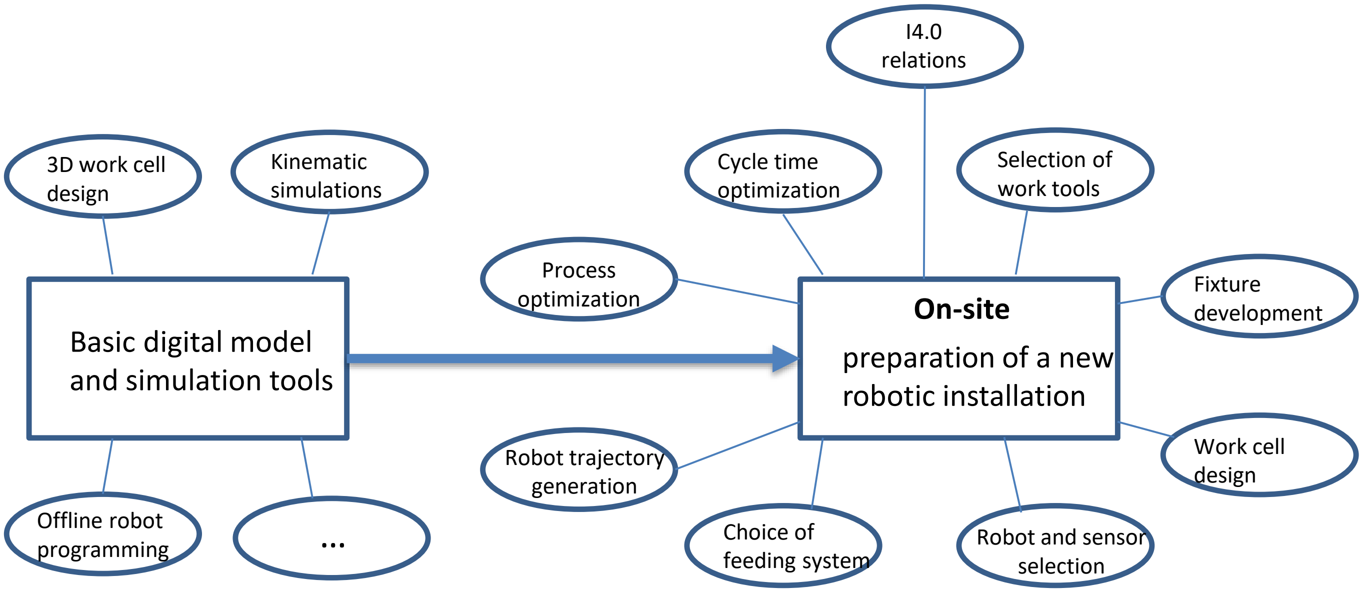
Trend in robotic automation
Collaborative robots



Traditional programming of robot systems



Using just simple digital tools

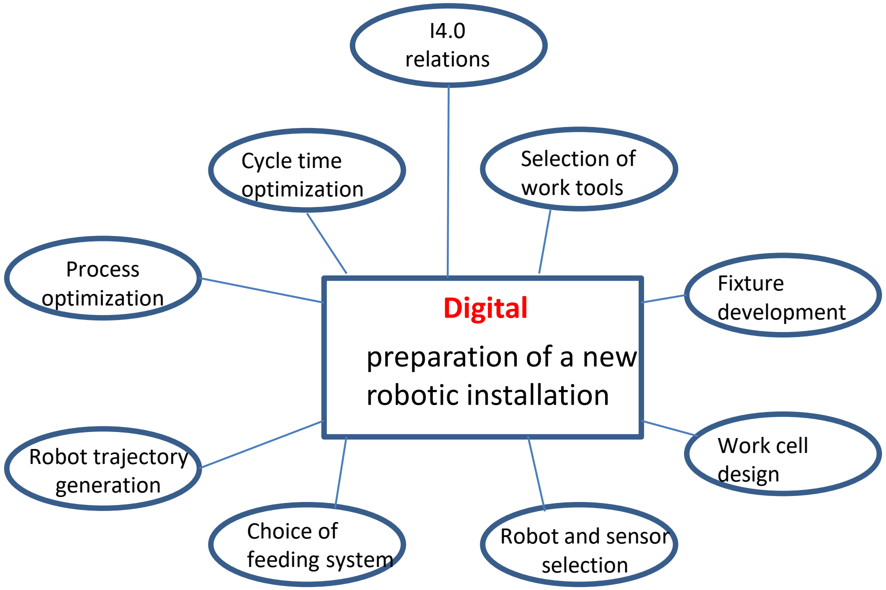


Problem: The robot systems integrator is often unable to offer even these basic digital tools

Problem: 3D CAD models of all relevant parts in the production should be available at the enduser

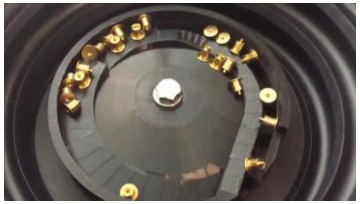
Problem: Unawareness of the advantages

Where we should be !!!

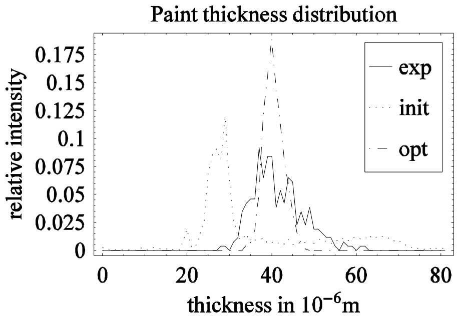


Three examples of digital modeling and simulation:

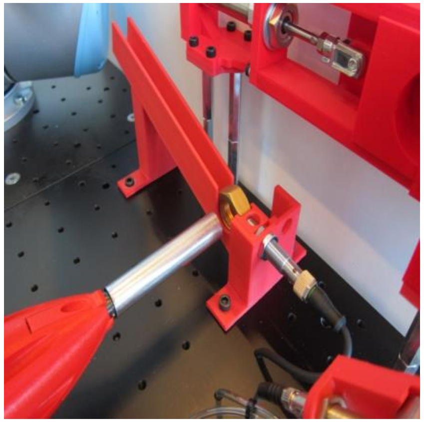
Choice of feeding system



Robot trajectory generation



Process optimization



Example 1: Bowl feeder design

Problem:

Part specific bowl feeders are implemented based on a trial and error procedure

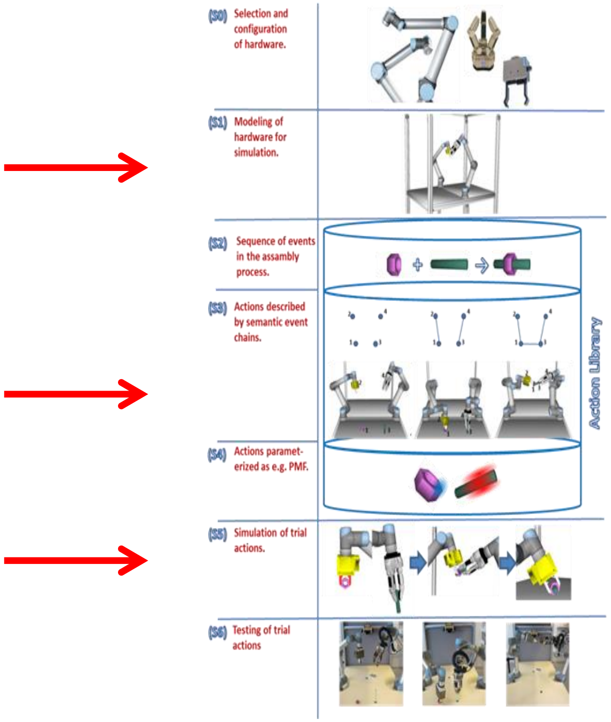
Requires very high expertise

Expensive bottleneck

Solution:

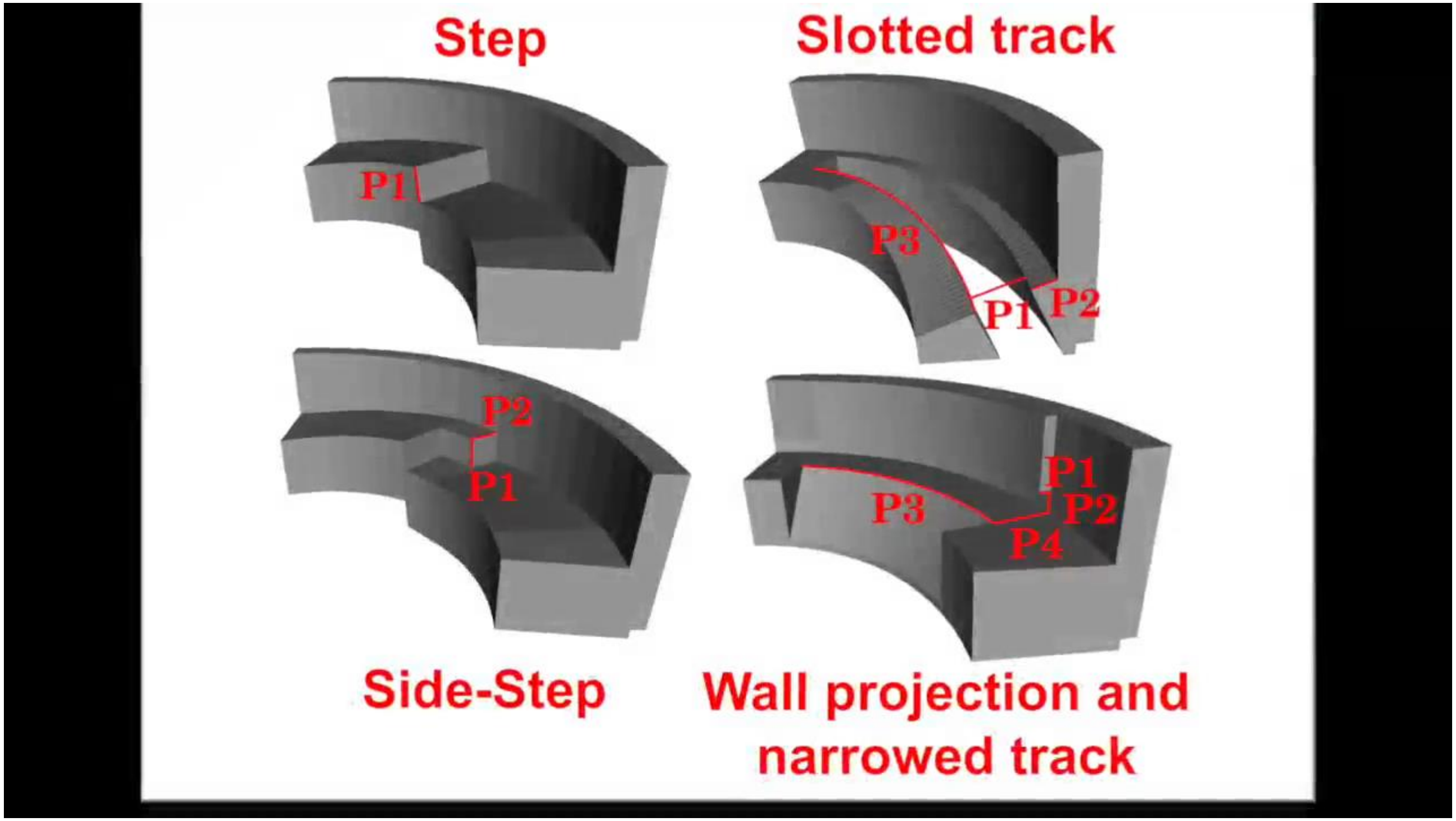
Use computer simulations as a replacement for the trial and error procedure

Reduces price and paves the way for the option of enduser designing these themselves



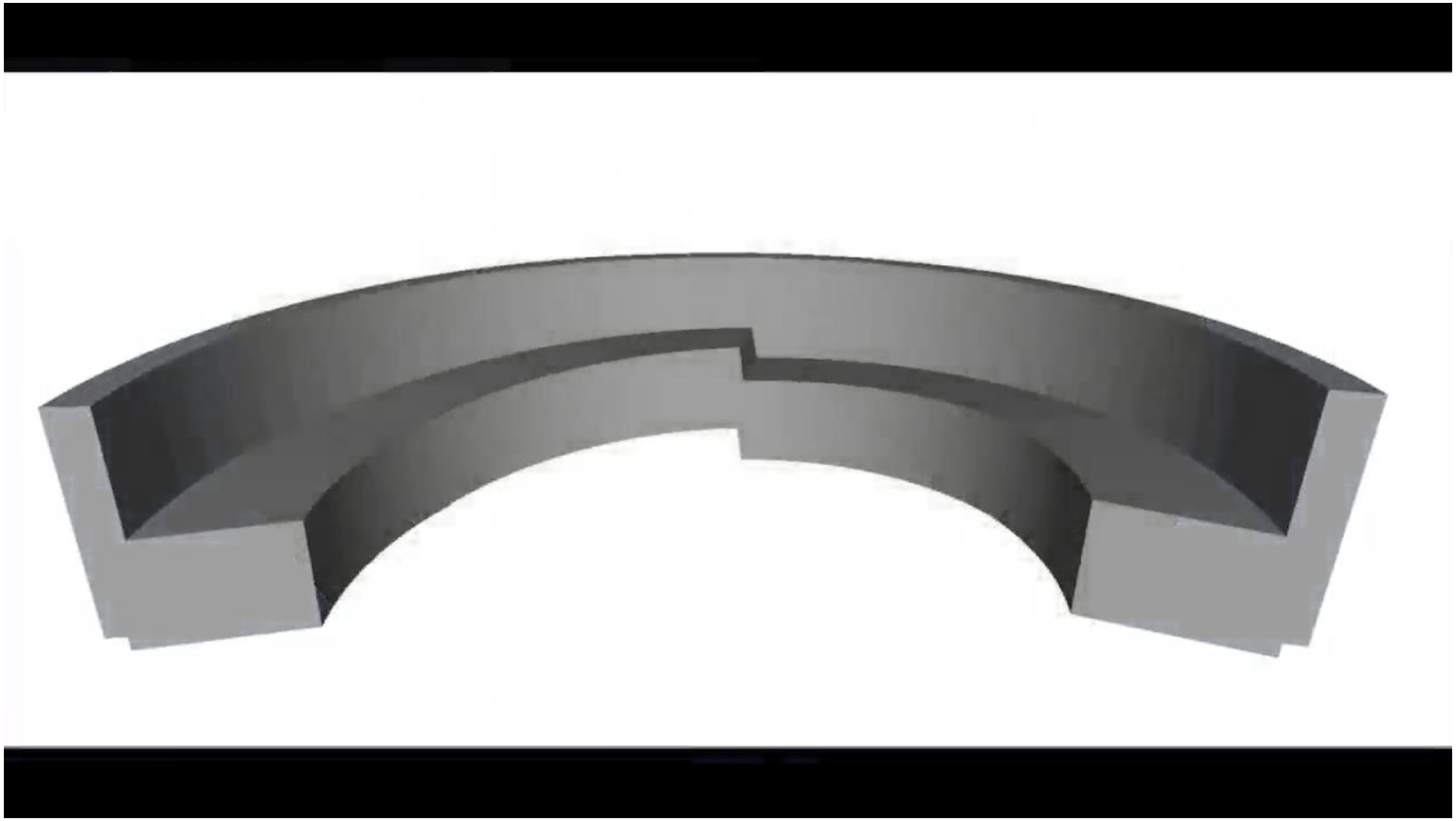
Choice of feeding system

Precise dynamic simulations of vibrational feeders for virtual design and optimization of traps and trap sequences



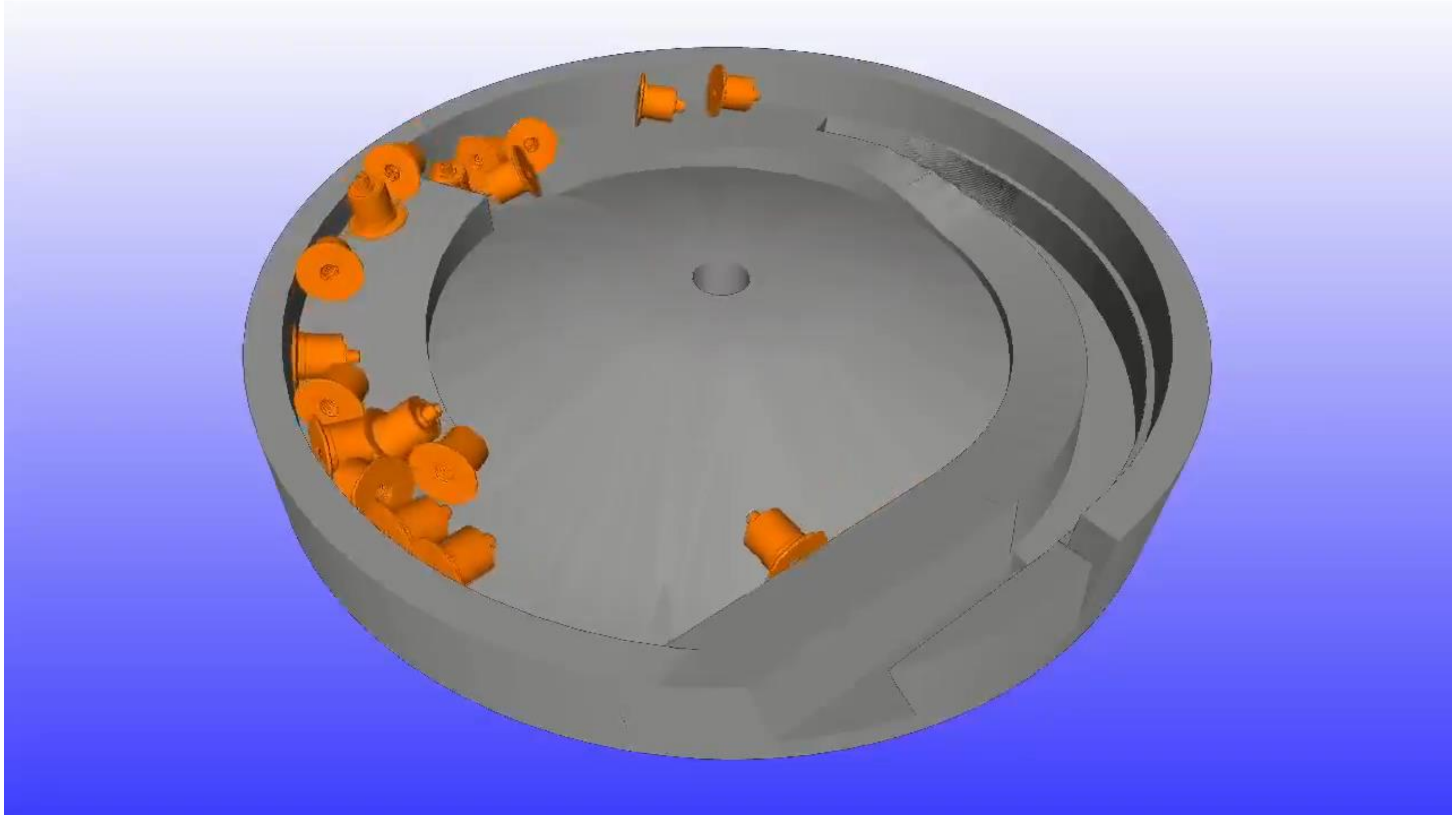
Choice of feeding system

Precise dynamic simulations of vibrational feeders for virtual design and optimization of traps and trap sequences



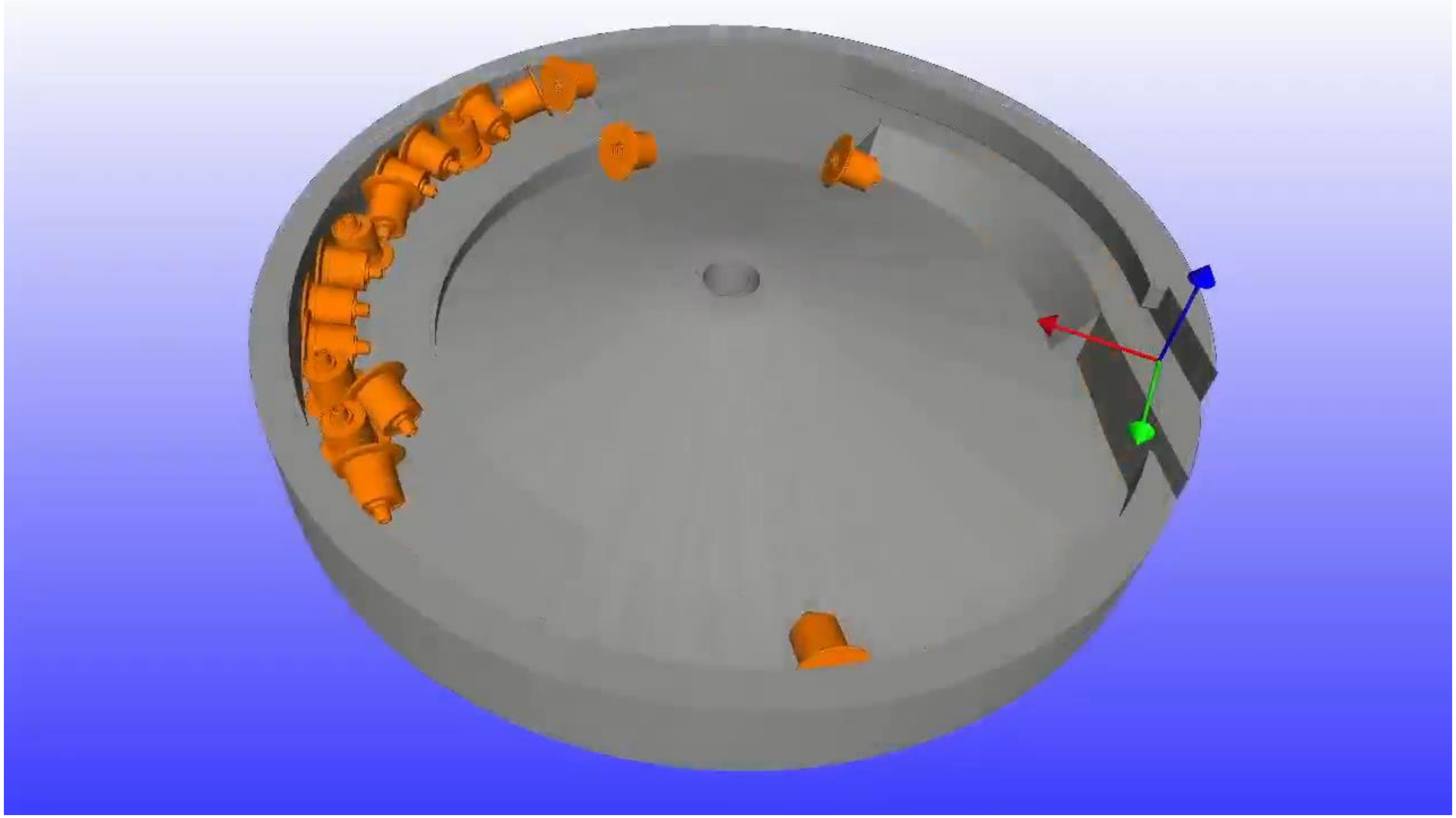
Choice of feeding system

Precise dynamic simulations of vibrational feeders for virtual design and optimization of traps and trap sequences



Choice of feeding system

Precise dynamic simulations of vibrational feeders for virtual design and optimization of traps and trap sequences



Choice of feeding system

Precise dynamic simulations of vibrational feeders for virtual design and optimization of traps and trap sequences

Example 2: Robot Spray Painting

Problem:

Double-curved surface to be painted (one of a kind)

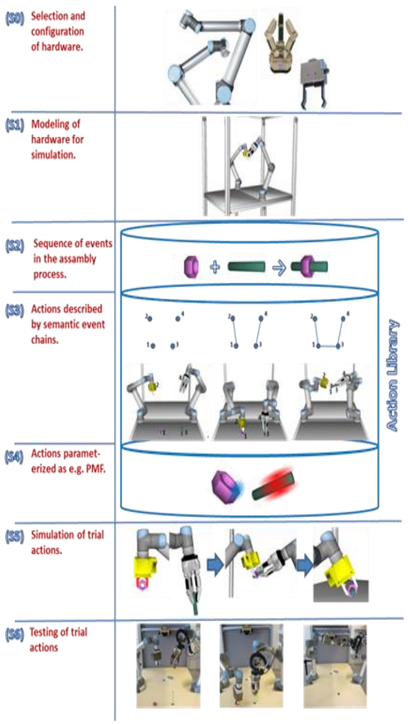
Desired thickness given

Automatical computation of spray nozzle trajectory

Solution:

Calibrate paint flux in spray nozzle

Compute nozzle trajectory by simulation based optimization



Spray cone flux model

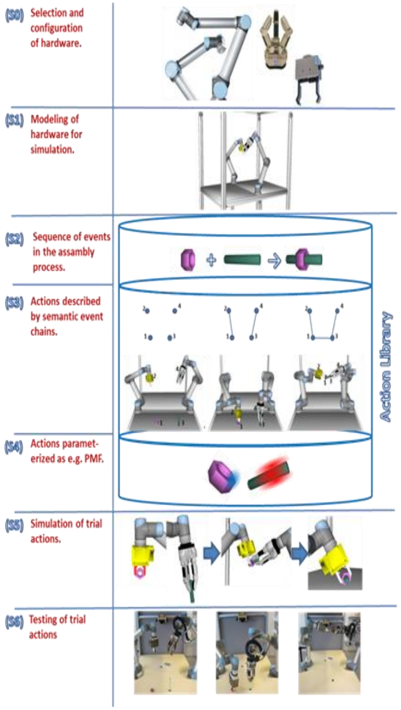
The streamlines radiate from the tool center, which ensures that the paint flux is contained within a cone whose silhouettes are straight lines,

If the cone is intersected by a plane perpendicular to the cone axis, the curves of constant flux in this plane are ellipses,

The divergence of j must be zero, which ensures that the total flux across any surface completely intersecting the cone is a constant, expressing conservation of paint.

$$\mathbf{j}(\mathbf{r}, \mathbf{u}_1 \cdots \mathbf{u}_3) = \alpha Q((\mathbf{r} \cdot \mathbf{u}_3)/(\mathbf{r}^T \mathrm{A} \mathrm{r})^{1/2})\mathbf{r}/(\mathbf{r}^T \mathrm{A} \mathrm{r})^{3/2}$$

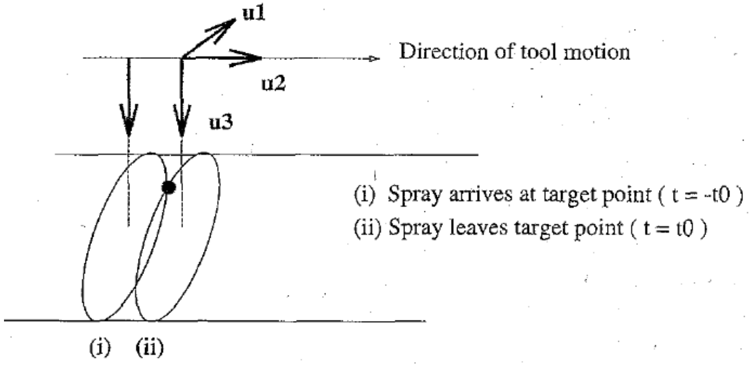
$$\mathbf{A} = \mathbf{u}_1(t)\mathbf{u}_1(t)^T + \alpha^2 \mathbf{u}_2(t)\mathbf{u}_2(t)^T + \mathbf{u}_3(t)\mathbf{u}_3(t)^T$$



Finding the flux parameters of the nozzle by calibration

- Paint flat surface
- Constant distance and speed
- Spray gun perpendicular to surface

$$c(u,v)dA = -\int_{-\infty}^{\infty} \mathbf{j}(\mathbf{r}(u,v) - \mathbf{r}_{TC}(t), \mathbf{u}_1(t), \mathbf{u}_2(t), \mathbf{u}_3(t)) \cdot (\mathbf{r}_u(u,v) \times \mathbf{r}_v(u,v))\, du\, dv\, dt$$



After several shifts of variables and some tedious algebra…

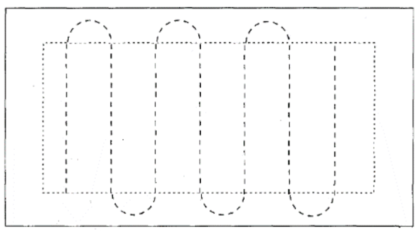
$$C(w) = \int_0^w \frac{q(u)}{\sqrt{w-u}} du.$$

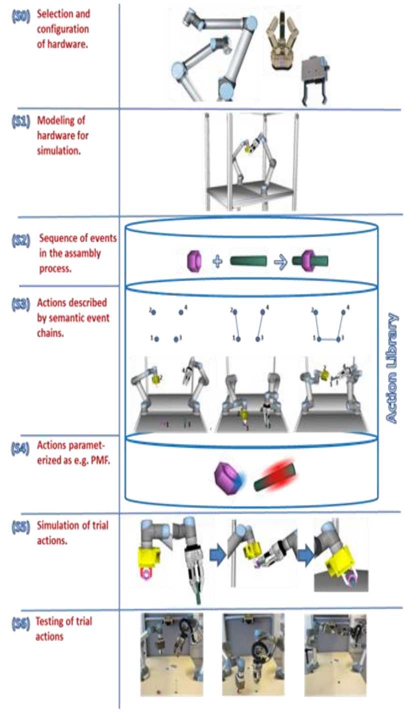
Analytical solution apparently first found by N.H. Abel in 1823:

$$q(u) = \frac{1}{\pi} \int_0^u \frac{C'(\eta) d\eta}{\sqrt{u-\eta}}.$$

Optimizing the nozzle trajectory

Initial guess

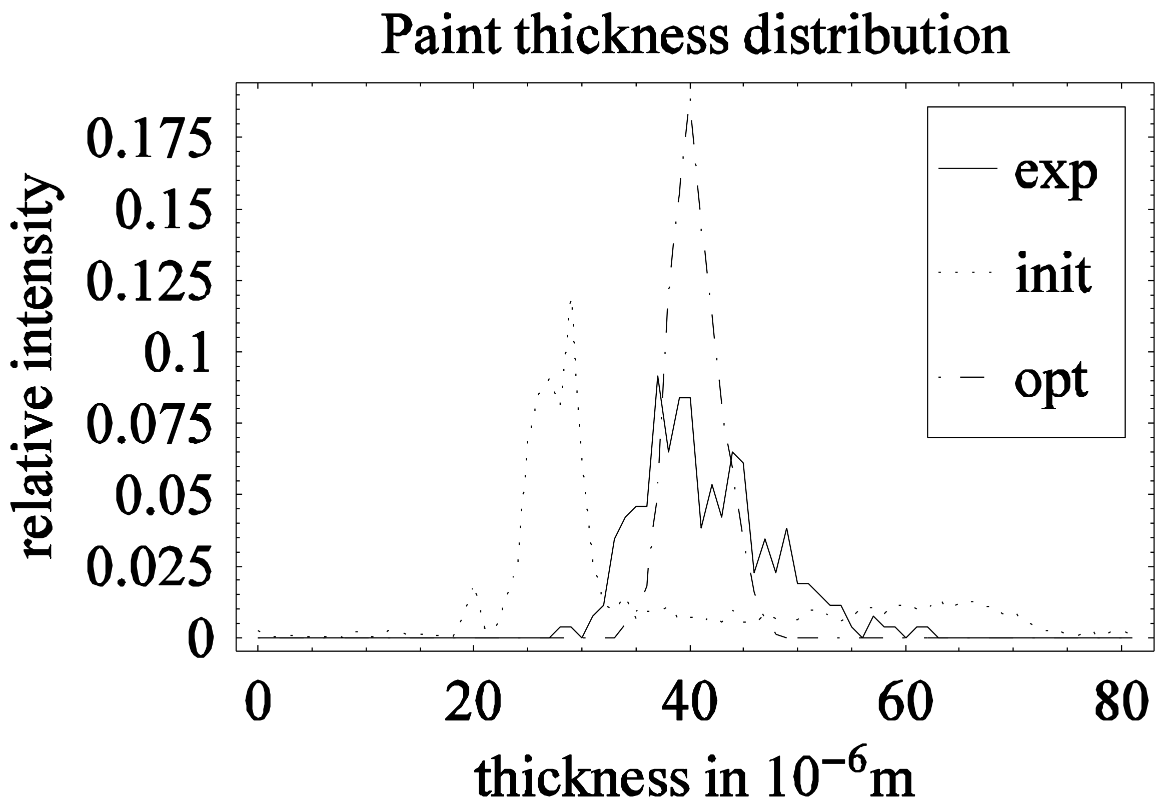




Optimization by simulation (guess trajectory and compute coverage thickness)

Painting a wheelbarrow

Simulation and experimental results



Example 3: Dynamic simulation models for robust robot assembly

Example 3: Dynamic simulation models for robust robot assembly

Problem:

Optimizing assembly operations on site can be difficult because errors are rare.

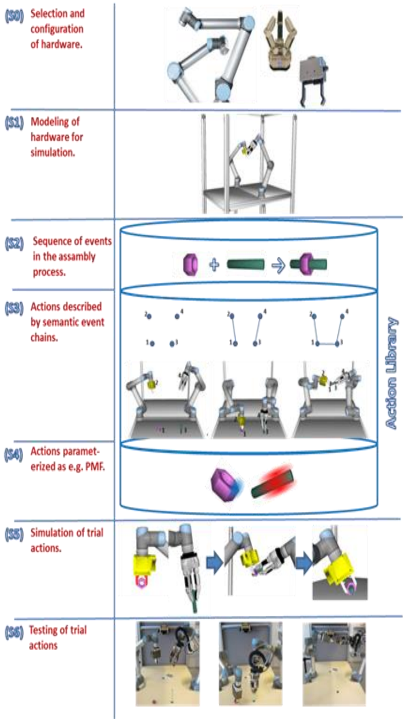
Hence the run-in time is long

Process is poorly understood

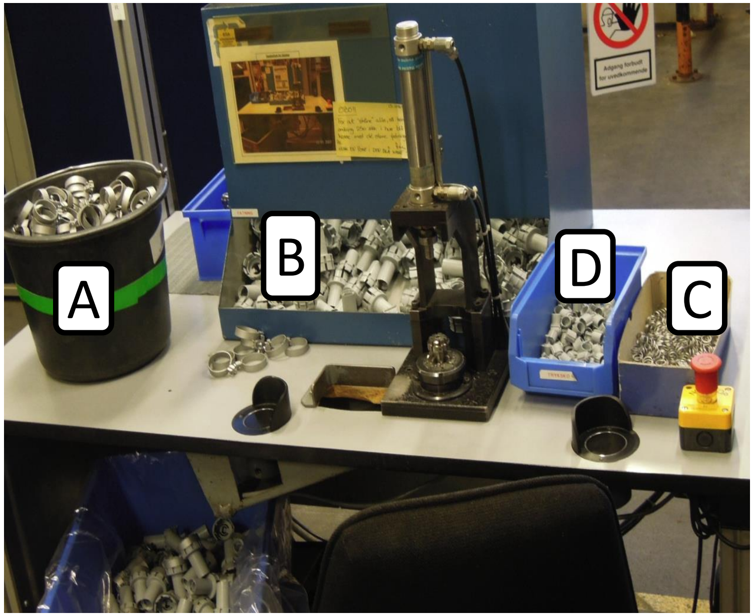
Solution:

Use computer simulations as a supplement for the trial and error procedure

Requires a process model and hence forces process understanding



Usage: Simple movable, reconfigurable, adaptable platforms



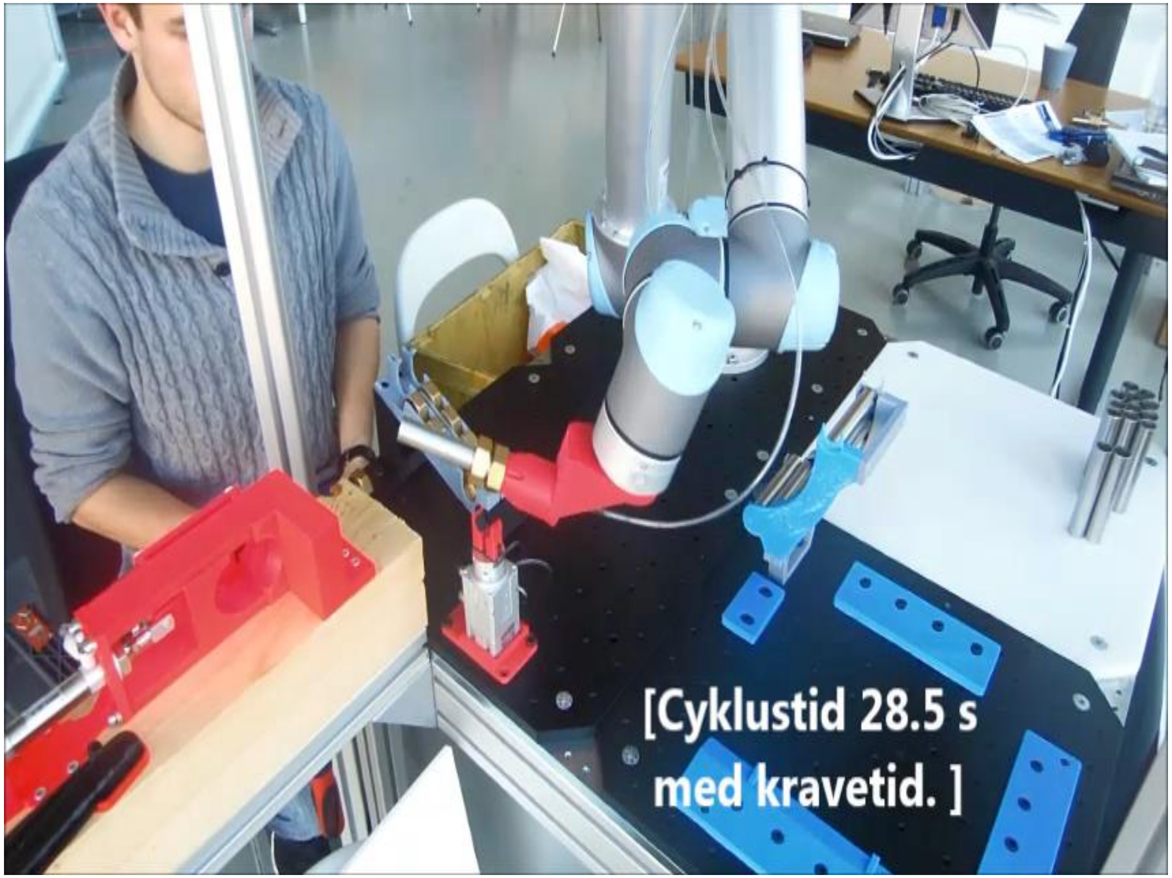
Movable and reconfigurable

Adaptable to randomly located parts

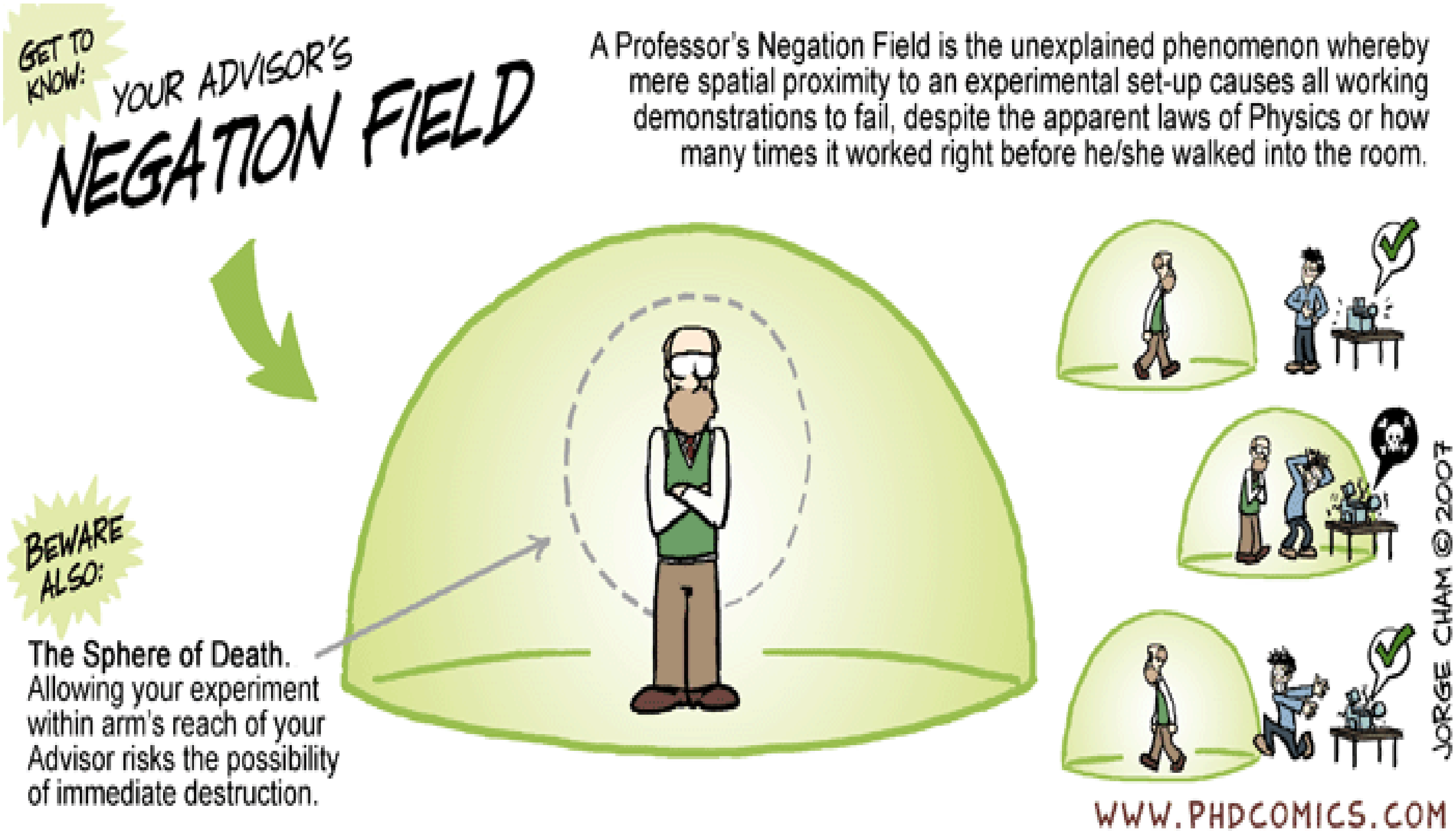
An example of a simple task

- Posed by the company KVM-Conheat
- Task: Place the two union nuts on the pipe as shown
- Several steps in the task, but main here is on getting the nuts onto the pipe
- Pipe and nut feeders used

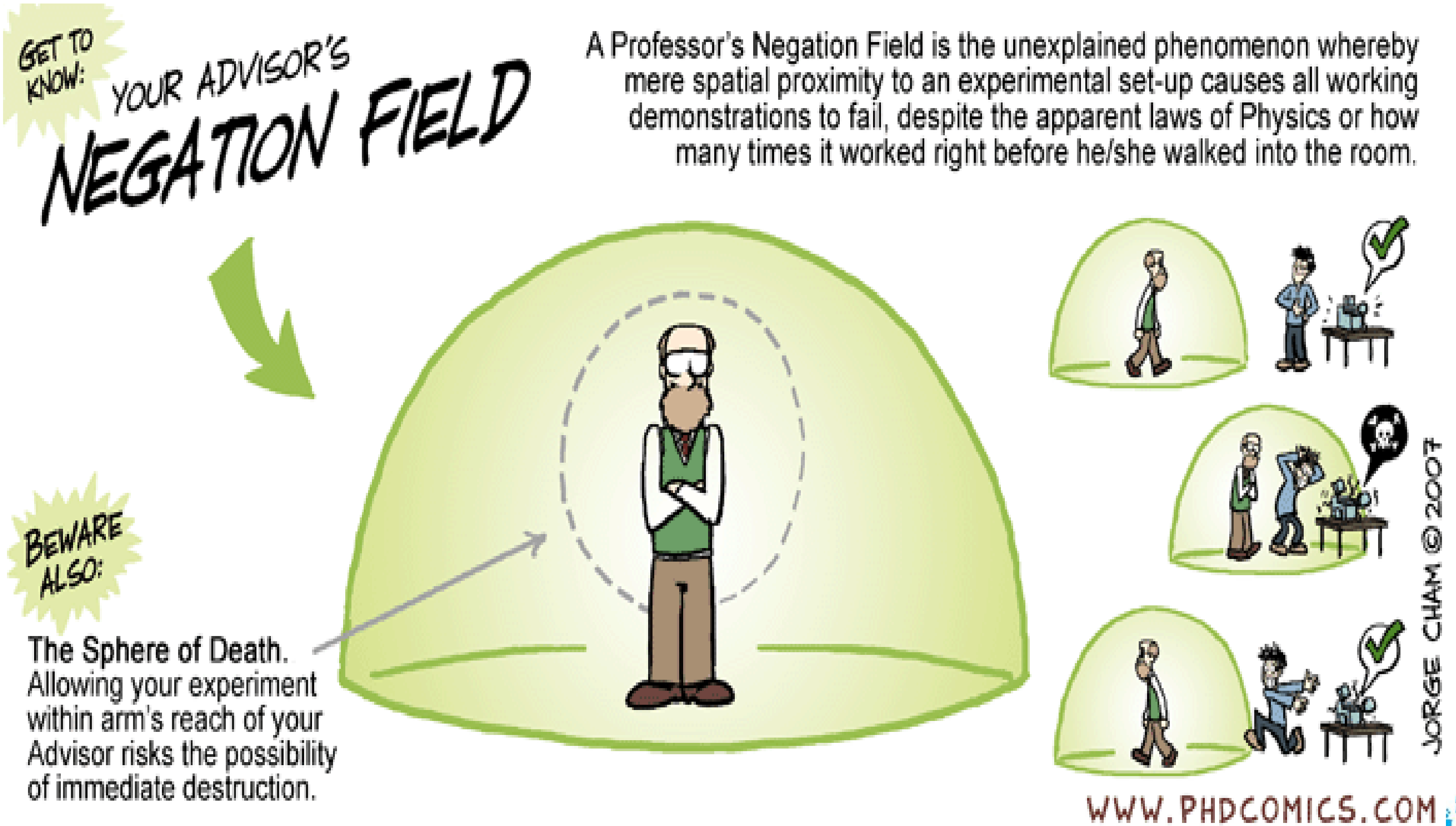
Conventional teach-in programming of the task



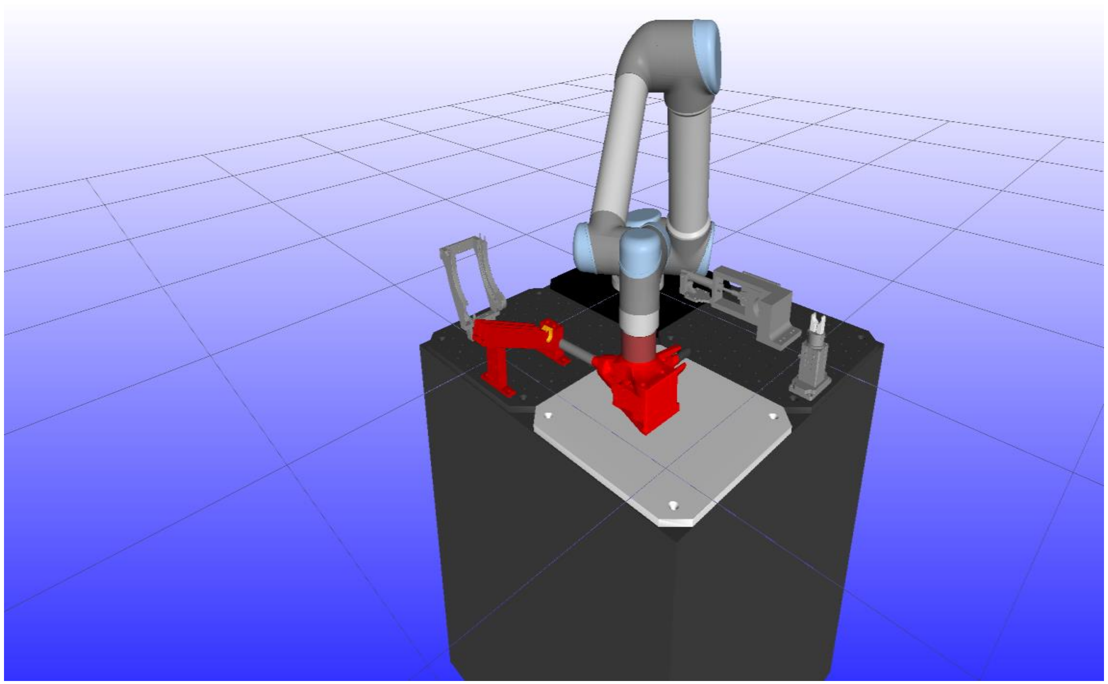
Professor's negation field:

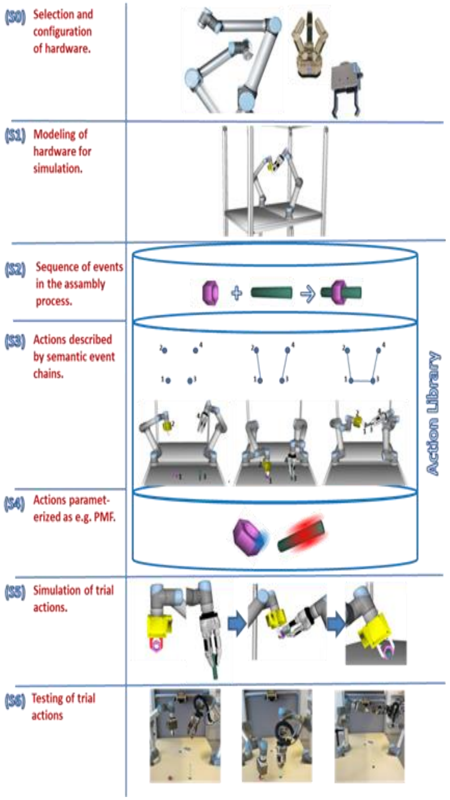


Professor's negation field:
Lack of uncertainty handling

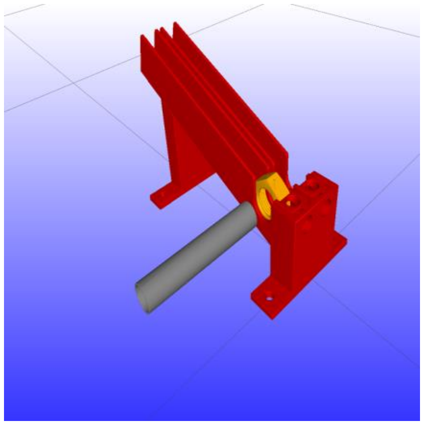


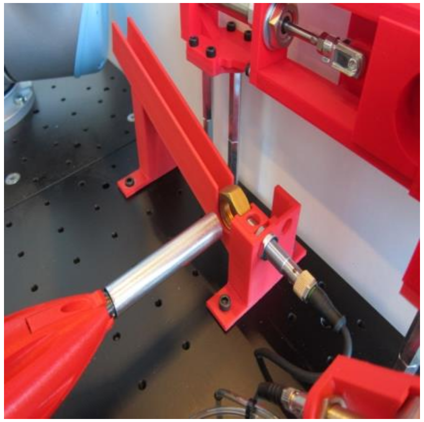
Modeling of hardware for simulation

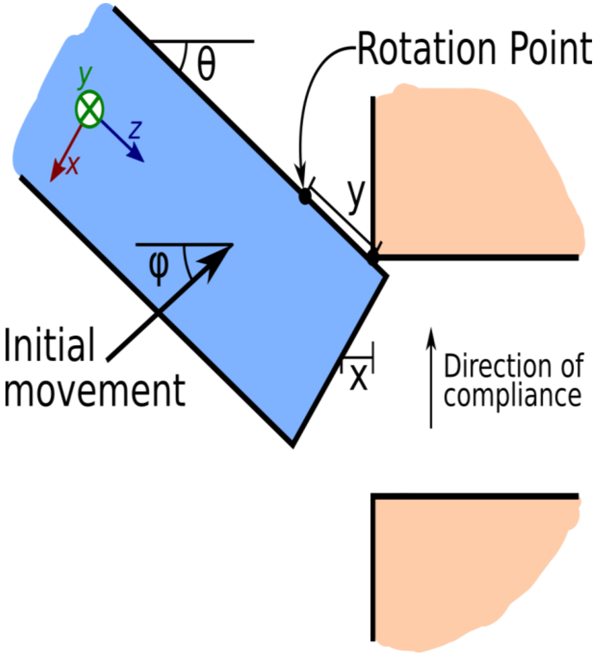


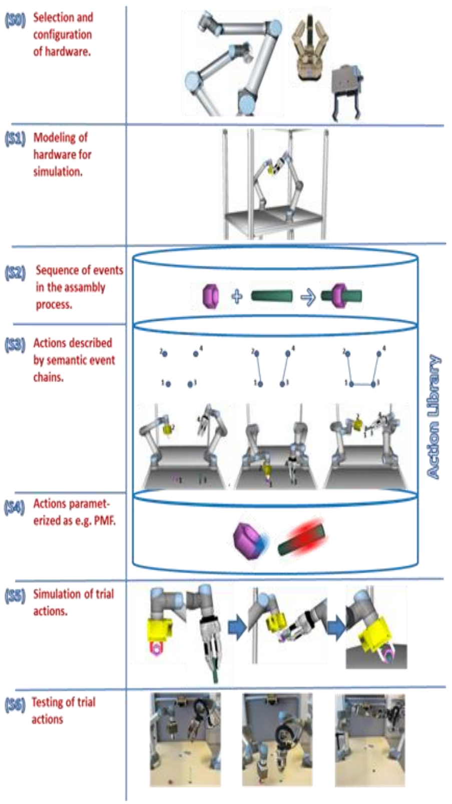


Action parametrization



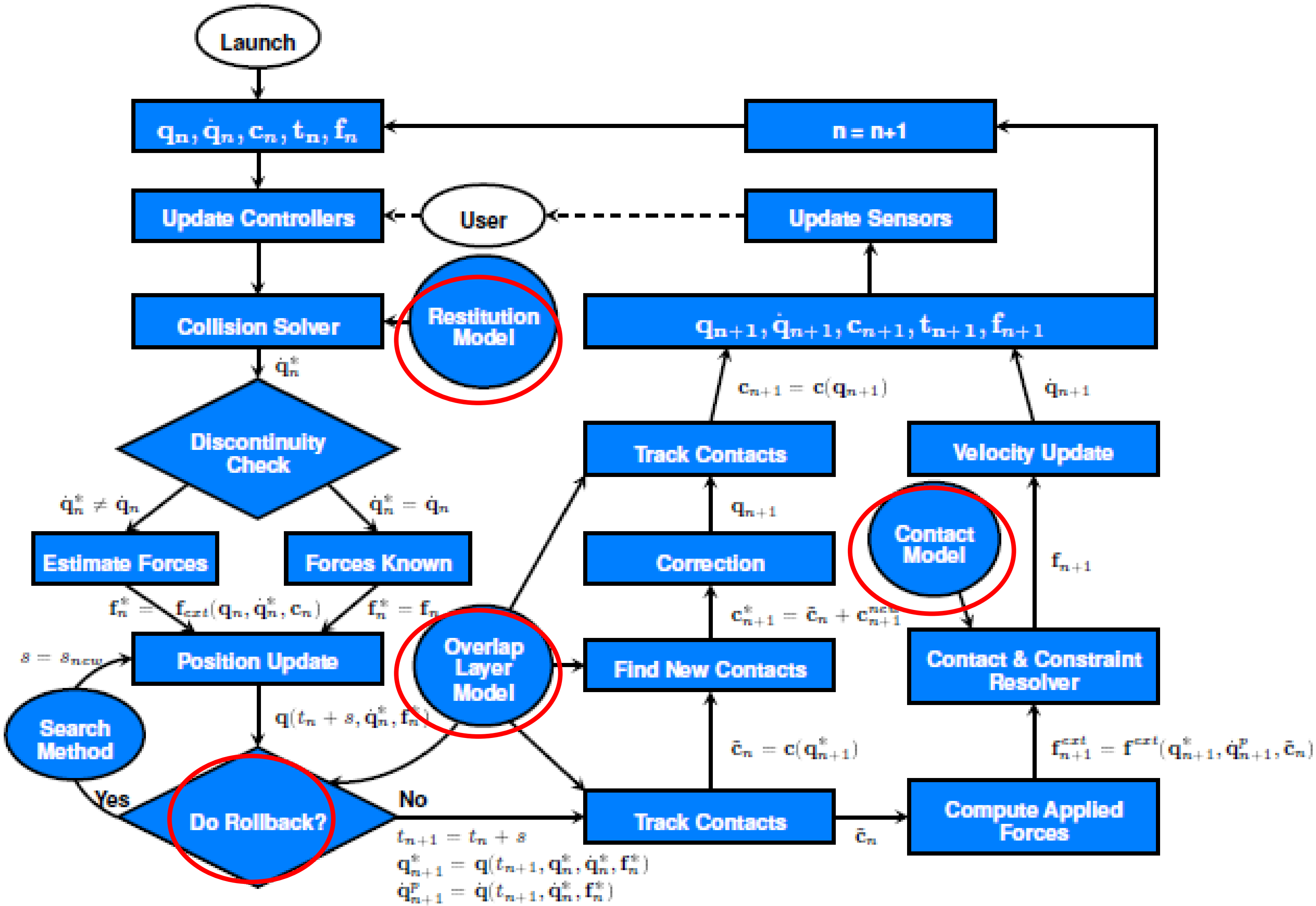




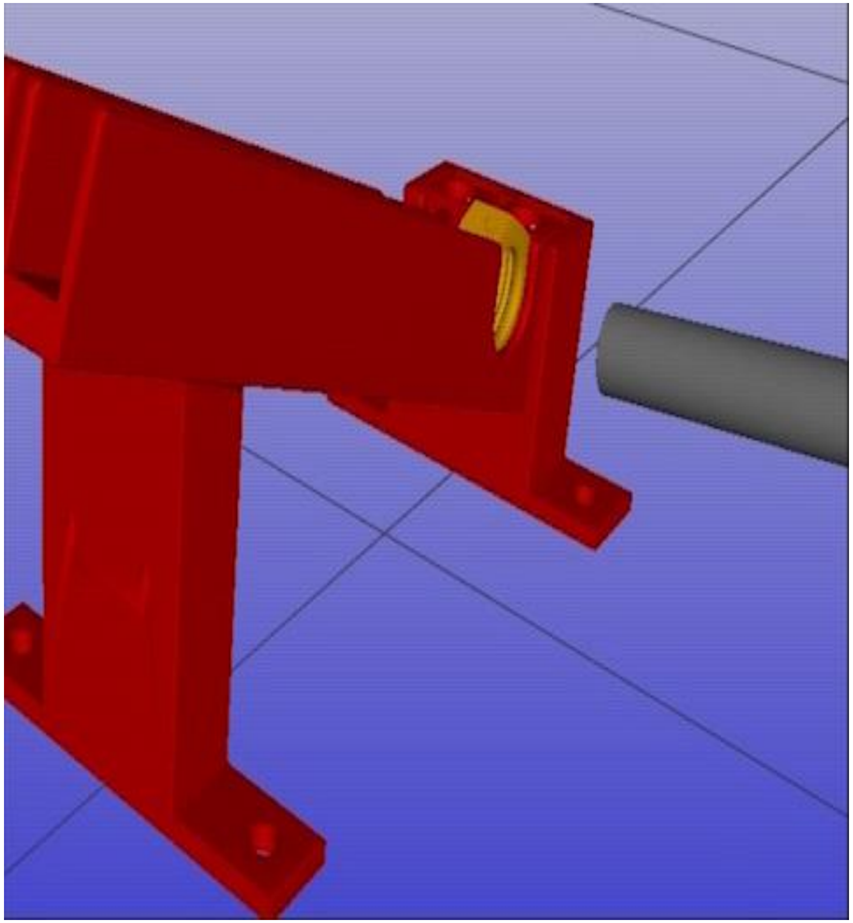


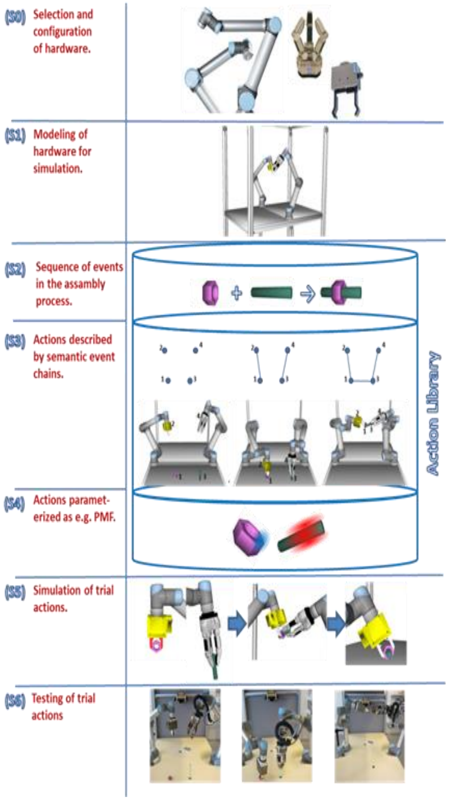
Dynamic simulations

- Newton/Euler rigid body equations of motion
- Bodies modeled as rigid
- Contact models (friction, restitution)
- Existing engines too inaccurate (aimed for computer games)


Launch
$q_n, \dot{q}_n, c_n, t_n, f_n$
n = n+1
Update Controllers
User
Update Sensors
Collision Solver
Restitution Model
$q_{n+1}, \dot{q}_{n+1}, c_{n+1}, t_{n+1}, f_{n+1}$
$\dot{q}_n^*$
Discontinuity Check
$c_{n+1} = c(q_{n+1})$
$\dot{q}_{n+1}$
Track Contacts
Velocity Update
$\dot{q}_n^* \neq \dot{q}_n$
$\dot{q}_n^* = \dot{q}_n$
q_{n+1}
Contact Model
Estimate Forces
Forces Known
Correction
f_{n+1}
$f_n^* = f_{ext}(q_n, \dot{q}_n^*, c_n)$
$f_n^* = f_n$
$c_{n+1}^* = \bar{c}_n + c_{n+1}^{new}$
$s = s_{new}$
Position Update
Overlap Layer Model
Find New Contacts
Contact & Constraint Resolver
Search Method
$q(t_n + s, \dot{q}_n^*, f_n^*)$
$\bar{c}_n = c(q_{n+1}^*)$
$f_{n+1}^{ext} = f^{ext}(q_{n+1}^*, \dot{q}_{n+1}^p, \bar{c}_n)$
Yes
No
Do Rollback?
Track Contacts
Compute Applied Forces
$\bar{c}_n$
$t_{n+1} = t_n + s$
$q_{n+1}^* = q(t_{n+1}, q_n^*, \dot{q}_n^*, f_n^*)$
$\dot{q}_{n+1}^p = \dot{q}(t_{n+1}, \dot{q}_n^*, f_n^*)$

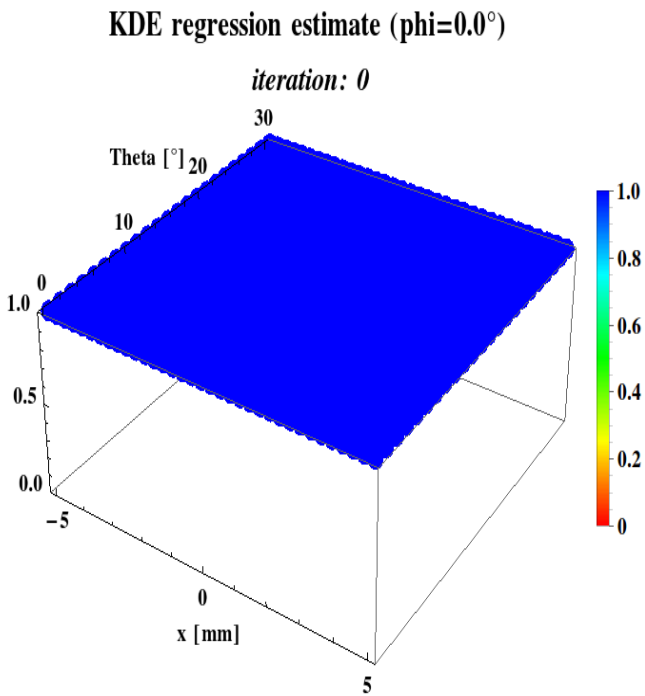
Simulation of trial actions:

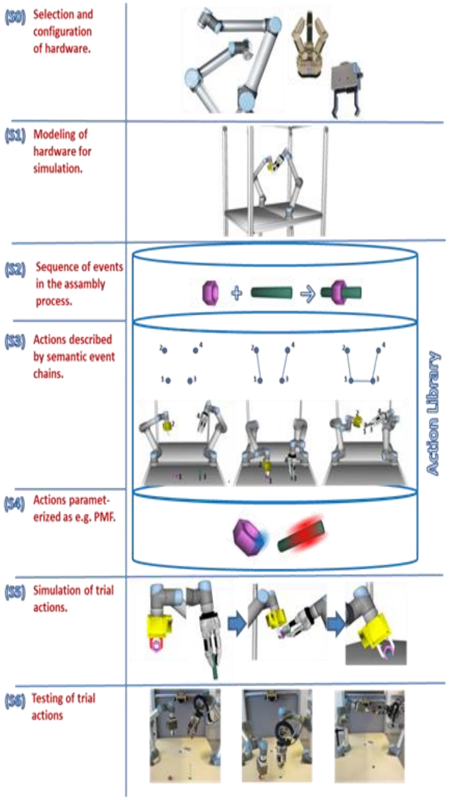




A chosen set of action parameters
A randomly chosen pose perturbation

Learning promising action parameters:

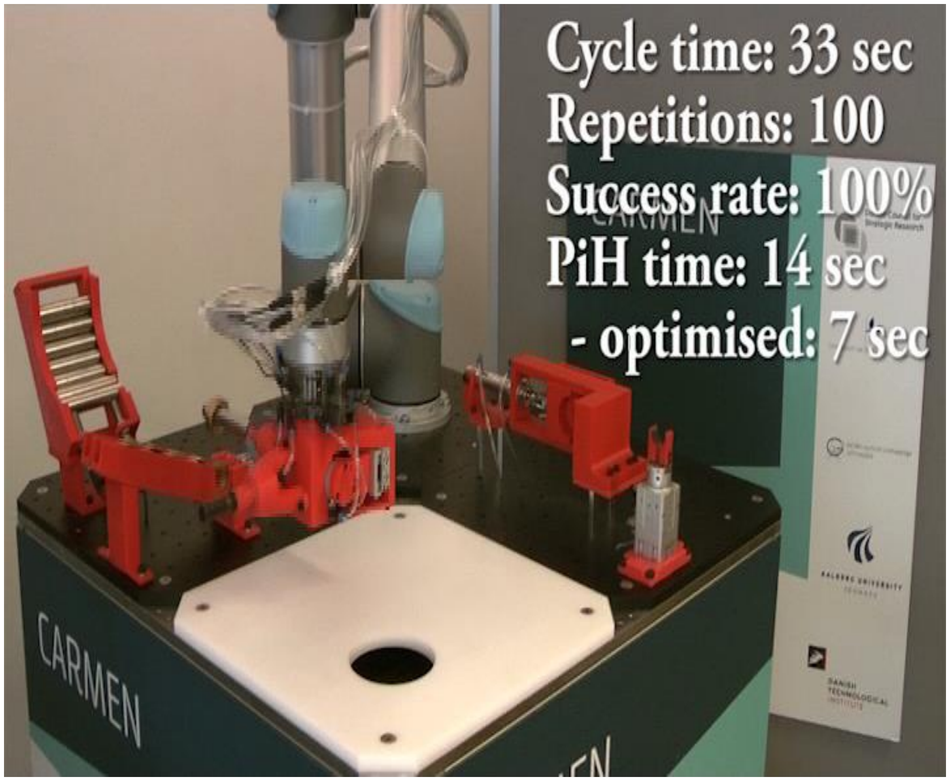


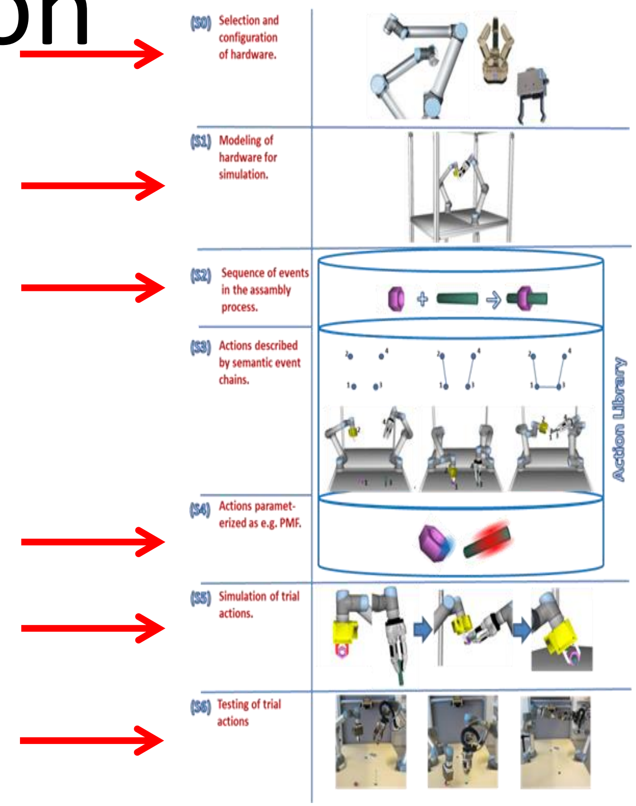


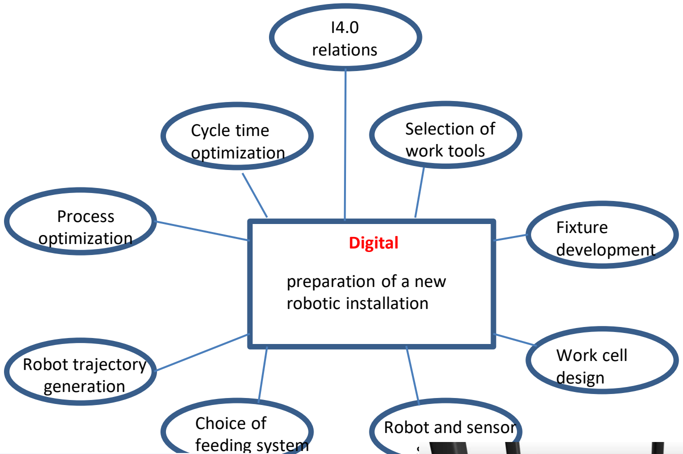
$$\hat{\mu}_H(x) = \frac{\hat{p}_O(x,s)}{\hat{p}_X(x)} = \frac{n^{-1}\sum_{i=1}^{n} K_{H,x_i}(x) O_i}{n^{-1}\sum_{j=1}^{n} K_{H,x_j}(x)}$$

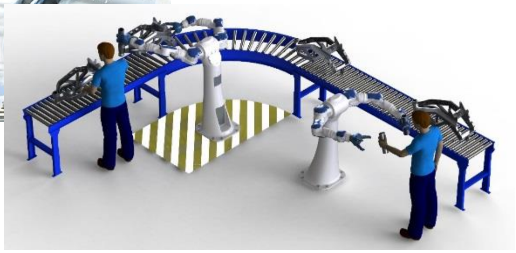
$$CI_{kde} = z \cdot \sqrt{\frac{||K||_2^2 \, \hat{\sigma}_H^2(x)}{n \, |H| \, \hat{f}_H(x)}}$$

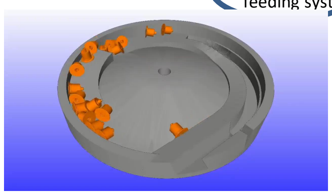
Learned solution

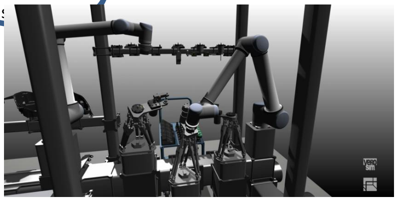


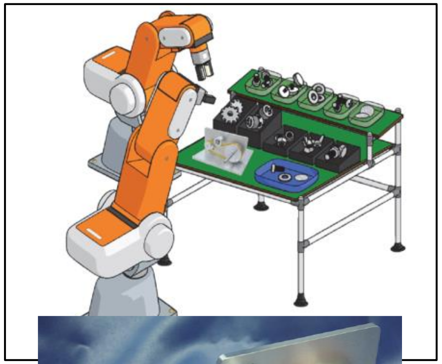


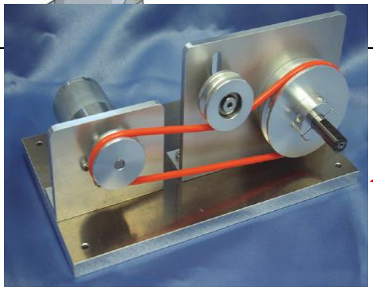

I4.0 relations
Cycle time optimization
Selection of work tools
Process optimization
Digital
preparation of a new robotic installation
Fixture development
Robot trajectory generation
Work cell design
Choice of feeding system
Robot and sensor

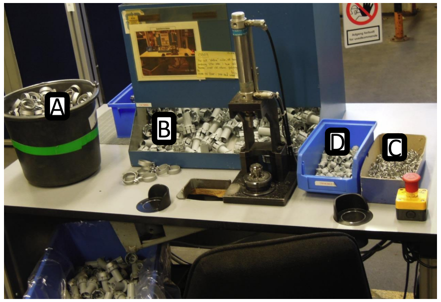


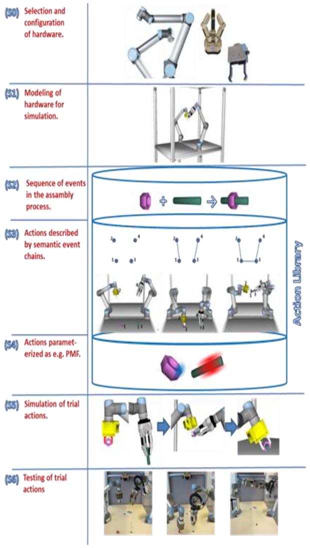









A
B
D
C


(S0) Selection and configuration of hardware.
(S1) Modeling of hardware for simulation.
(S2) Sequence of events in the assembly process.
(S3) Actions described by semantic event chains.
(S4) Actions parameterized as e.g. PMF.
(S5) Simulation of trial actions.
(S6) Testing of trial actions
Action Library