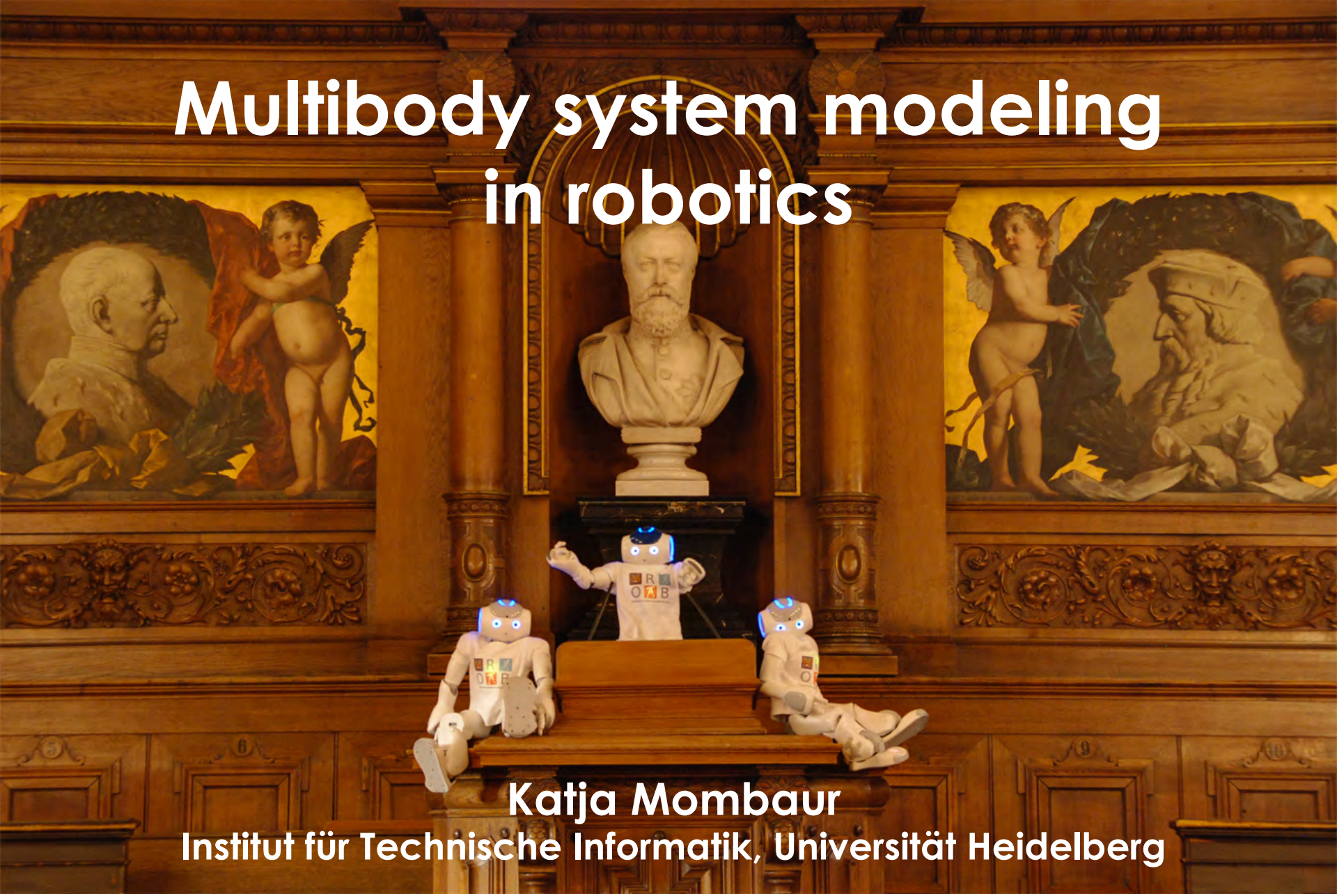
Multibody system modeling in robotics



Katja Mombaur
Institut für Technische Informatik, Universität Heidelberg

Robotics 2 - 6 May 2019

Exam planning

- **The grade for the class is determined by the grade of a 2 hour exam** at the end of the semester
  - First exam (erster Prüfungsversuch)
    Date: Monday, July 29, 2019 11-13
    Registration via MÜSLI until July 20, 2019
  - Second exam (zweiter Prüfungsversuch) - in case of failure or sickness during the first exam
    Date will be announced in time (most probably in last week of vacation period)

- **Admission to the exam** will be decided based on a successful participation in the computing exercises (details see slides of first lecture, April 29)

Terms that you should be familiar with after last lecture

- Modeling
- State variables
- Control variables
- Model parameters
- Different types of process models (deterministic/ stochastic, continuous / discrete in time /states)
- Models of differential equations (ODE, DAE)
- Index of a DAE
- Boundary conditions:
  - Start / end point constraints
  - Coupled / decoupled b. c.
  - Periodic b. c.
- Bounds (inequality constraints)
- Multi-phase models
- Discontinuities
- Objective function (cost function):
  - Lagrange type
  - Mayer type
  - Bolza type

What you will learn in this lecture

- How do robot and human models fit into the general context of process modeling ?
- What do the different components of the dynamic process model look like for robots (and humans)?
- What do the equations of motion look like for robots (and humans)?
- What choices of coordinates do you have for modeling robots (and humans)?

- In addition, you will be reminded of some fundamentals of kinematics and dynamics of mechanical systems (from the Robotics 1 course)

Process model with explicit differential equation(s)

- First order system of Ordinary differential equation (ODE):

$$\dot{x} = f(t, x(t),$$

How can we set up equations of motions for robots?

- Time: $t \in [t_0, t_f]$

Example: Robot

- States: $x(\cdot) : [t_0,$ velocities)
- Controls $u(\cdot) : [t_0,$ ues)
- Parameters $p \in \mathbb{R}^{n_p}$ nertia, length etc.)

How do we select state and control variables and parameters for robot models?

How do we select good coordinates for position and velocity description ?

Differential algebraic equations (DAE)

- System of ordinary differential equations is augmented by n_z algebraic equations

$g_i(\cdot)$ and algebraic variables $z(t) : [t_0, t_f] \mapsto \mathbb{R}^{n_z}$

$$\dot{x}(t) = f(t, x(t), z(t), u(t), p)$$ ← Differential equations

lgebraic equations

In which cases do we get ODEs and when do we get DAE models in robotics? Related to the choice of coordinates (minimal or redundant)

- The derivative of s also zero:

$$0 = \frac{\mathrm{d}g(t, x(t$$ $$+ \frac{\partial g}{\partial z}\frac{\mathrm{d}z}{\mathrm{d}t} + \frac{\partial g}{\partial u}\frac{\mathrm{d}u}{\mathrm{d}t}$$

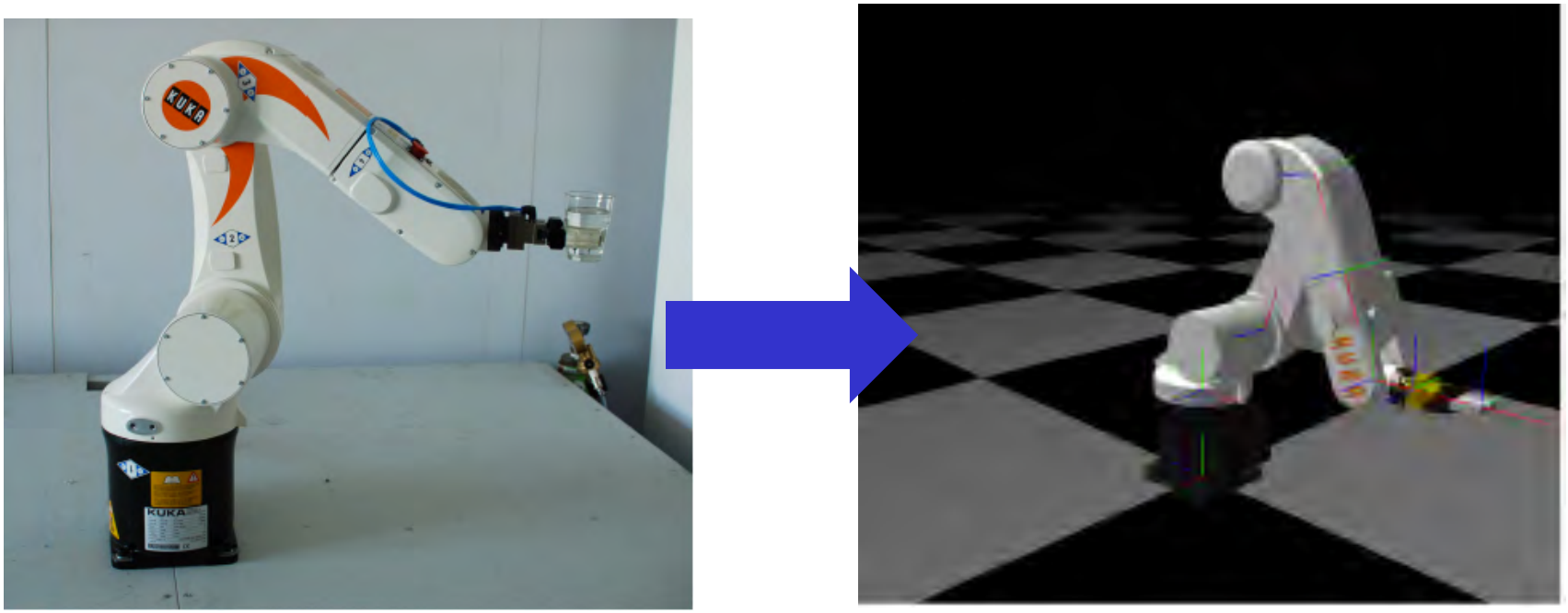
- If $\frac{\partial g}{\partial z} \in \mathbb{R}^{n_z \times n_z}$ can be inverted, it holds:

$$\frac{\mathrm{d}z}{\mathrm{d}t} = -\frac{\partial g}{\partial z}^{-1}\left(\frac{\partial g}{\partial t} + \frac{\partial g}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial g}{\partial u}\frac{\mathrm{d}u}{\mathrm{d}t}\right)$$

➡ Transformation of whole system to ODE is possible

Modeling of our KUKA robot

- Which characteristics of the real robot should the model include?



Modeling of our KUKA robot

- Which characteristics of the real robot should the model include?

Possibilities of motion:
6 degrees of freedom (DOF)

Geometry of the segments

Mass distribution
of the segments

Generation of
motion by motors:
Actuator bounds
upper and lower
limits

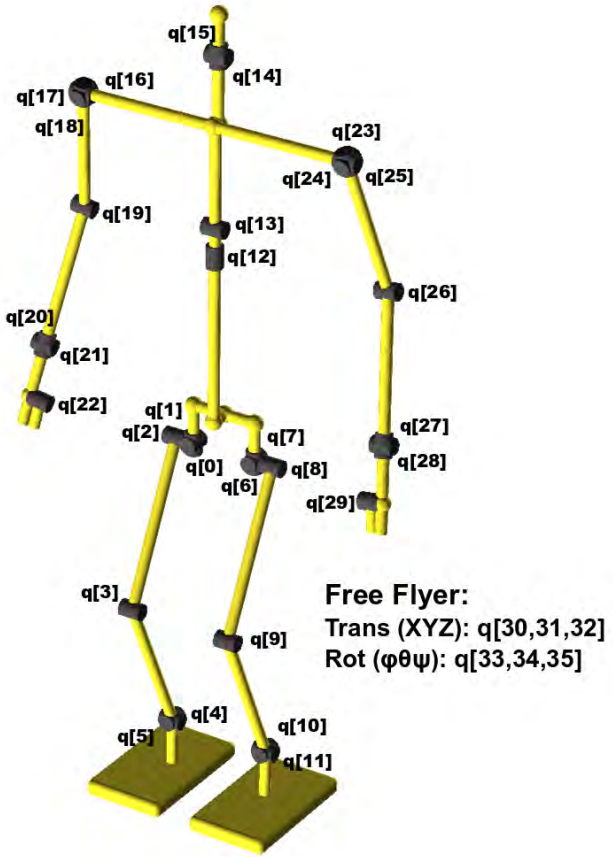
Joint angle bounds
(upper and lower limits)

Load limits

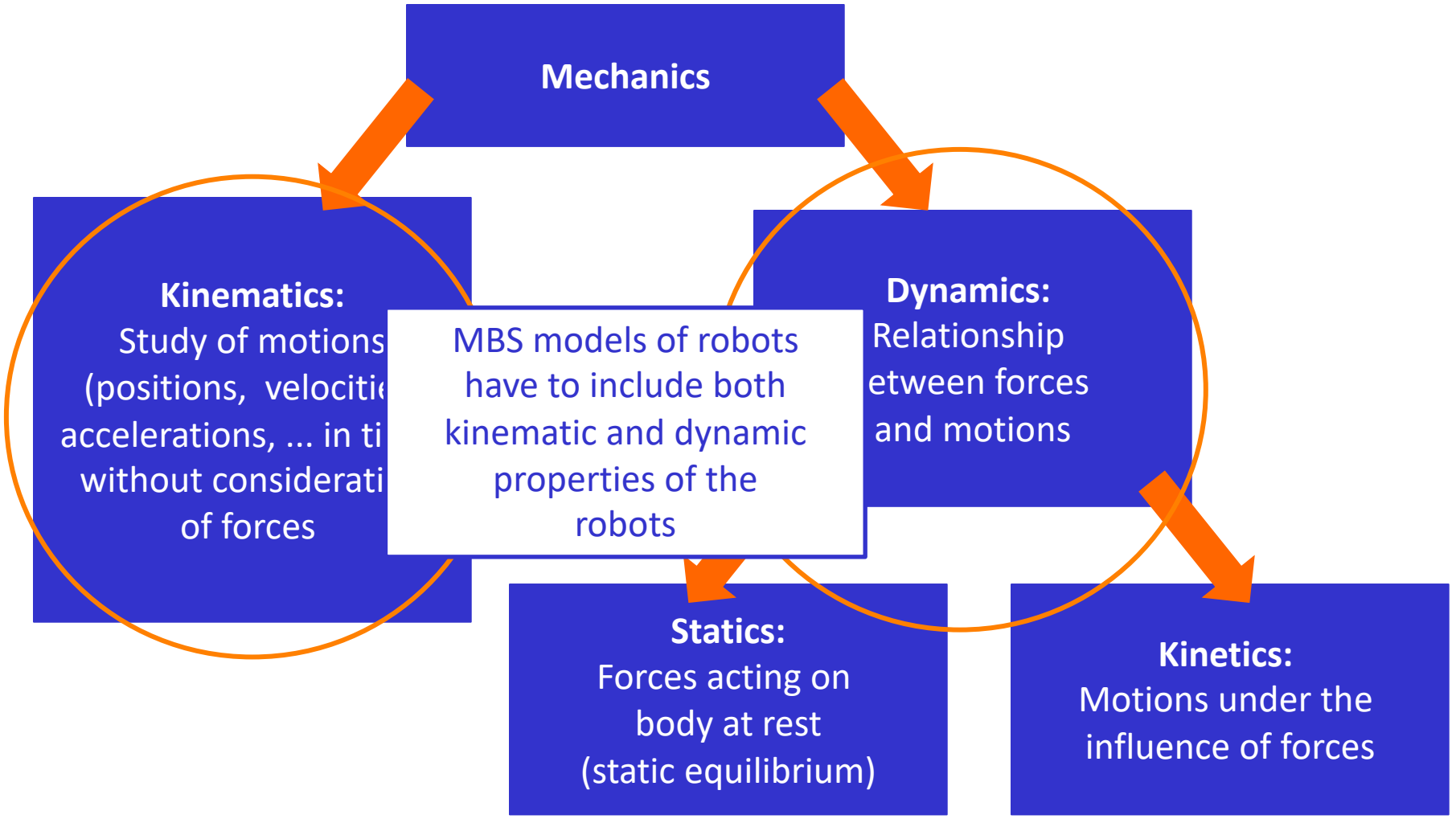
Angular velocity bounds
(upper and lower limits)

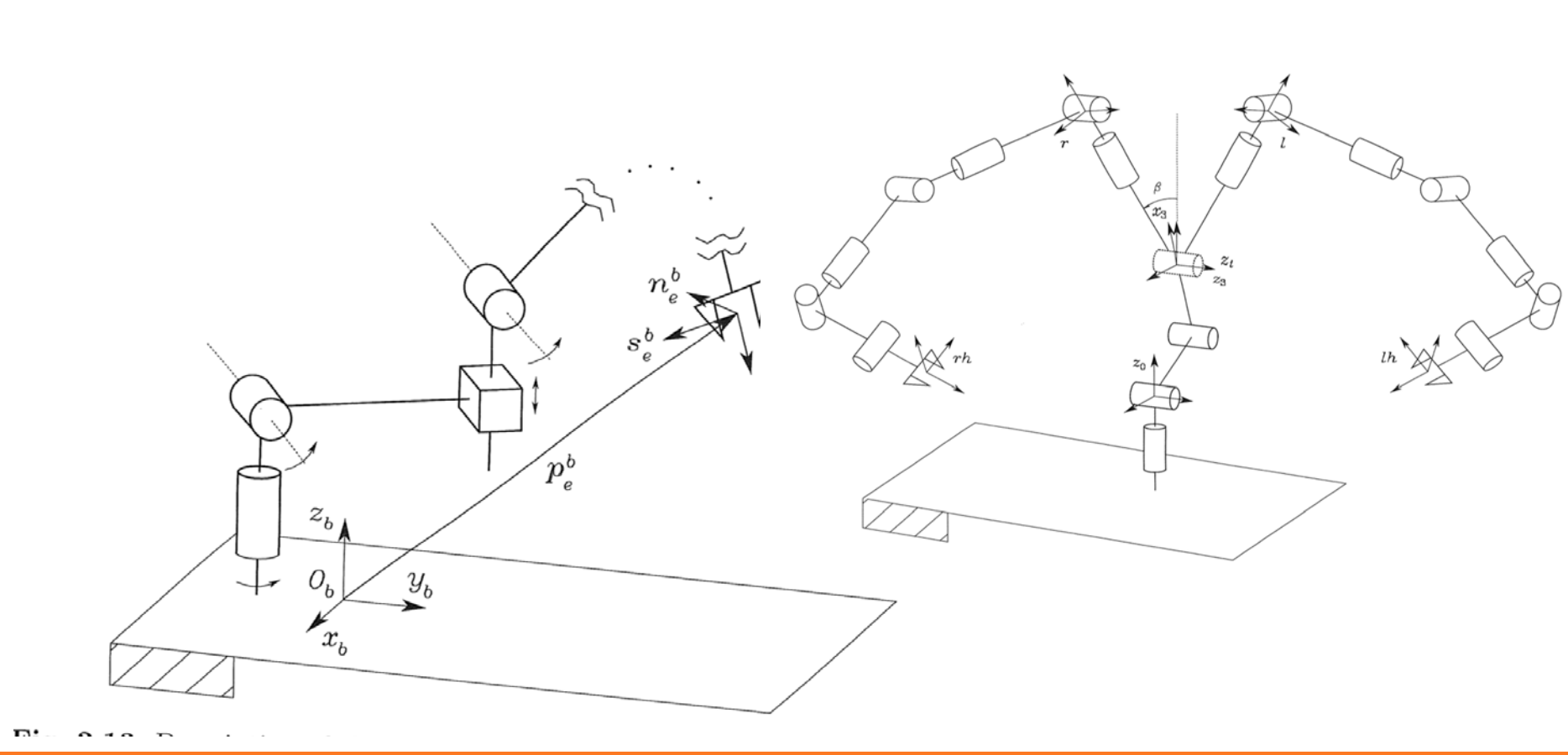
Robots are decsribed as rigid multibody systems

- Multibody systems consist of multiple rigid segments that are connected by joints



Overview –Mechanics

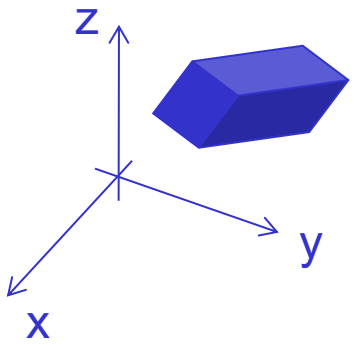


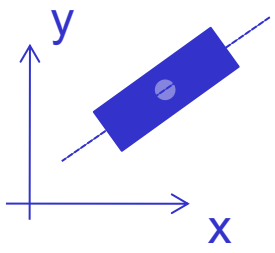


Multibody system kinematics

Reminder: Degrees of freedom (DoF) of multibody systems

- Recall: The **position of an (extended) body** is determined by its (translational) **position** and **orientation**. It has 6 **degrees of freedom** in space and 3 in the plane

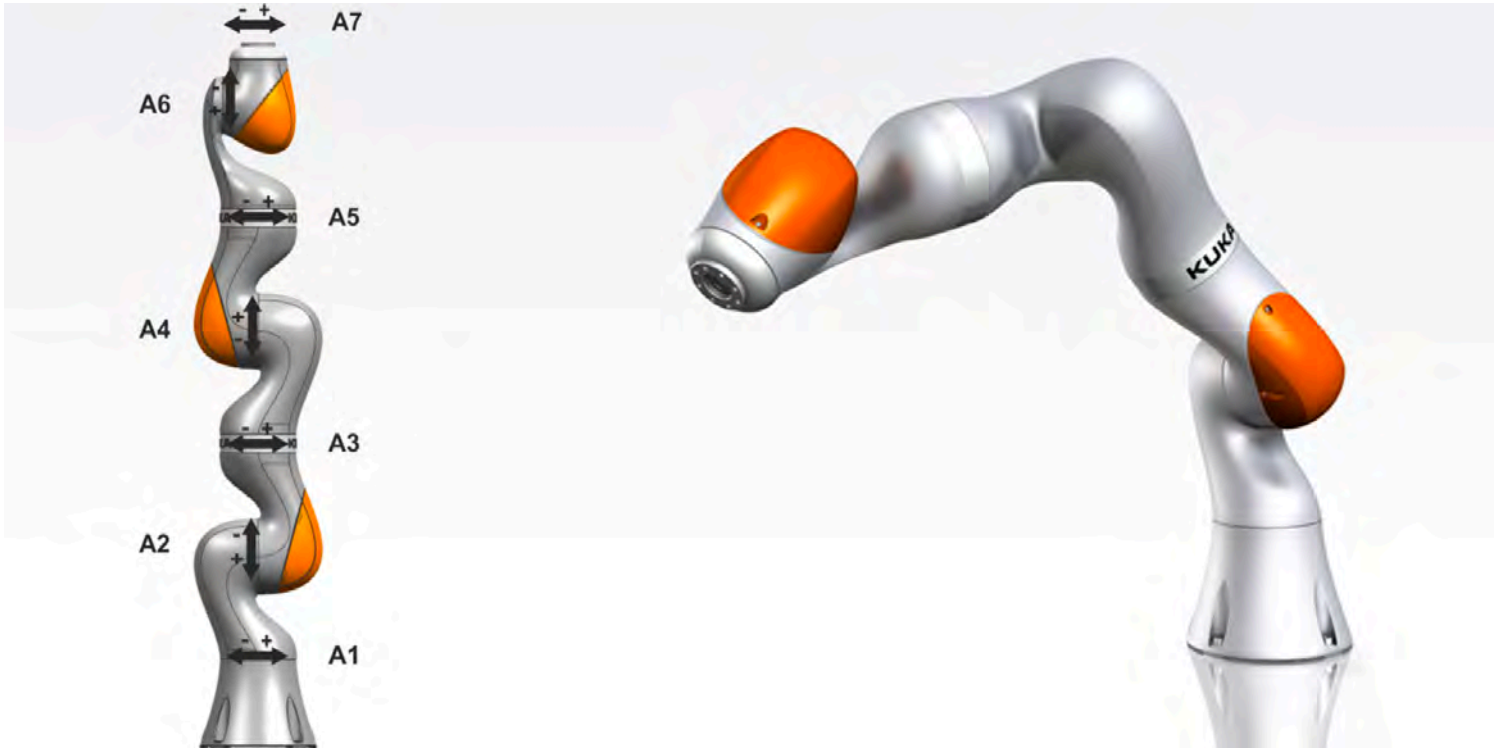




- In rigid body systems, segments are connected by joints which couple their motions together (precise way of coupling depends on the specific joint type)
- Two different ways of calculating the number of DoF in a multibody system
  - Count joint DoF from beginning to end (+ free floating base if applicable)
  - Start with n free bodies and subtract DoF dur to joint constraints

Degrees of freedom of a robot manipulator

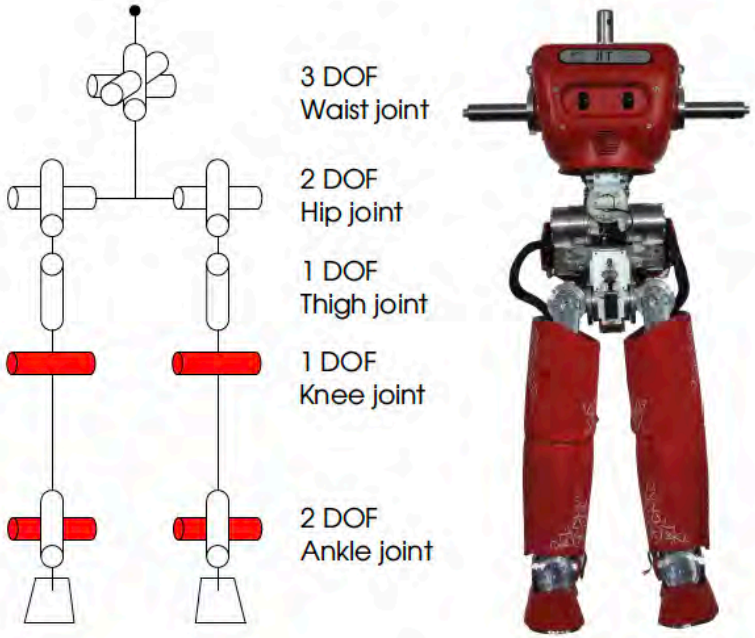
- In industrial robotics robotics, it is very straightforward: count degrees of freedom (DoF) along the kinematic chain
- KUKA Lighweight robot : 7 DoF



Degrees of freedom of a humanoid robot

In humanoid robots, it gets more complex. We have to distinguish

- Internal degrees of freedom / degrees of freedom of joints
- Total (maximum) degrees of freedom including the external degrees of freedom / "free flyer" (6 DOF in 3D)
- "Actual" degrees of freedom depending on the particular situation (contacts etc.)



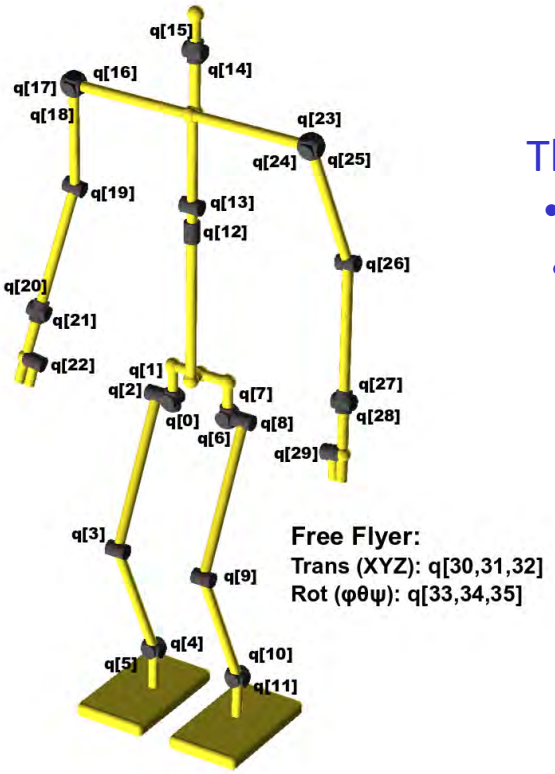
The HeiCub robot has

- 15 internal DOF
- 21 in total max

Degrees of freedom of a humanoid robot

In humanoid robots, it gets more complex. We have to distinguish

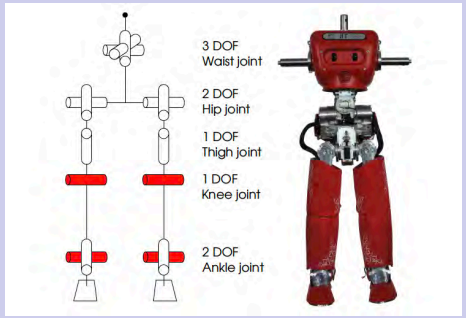
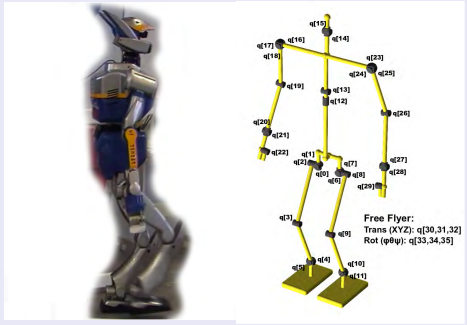
- Internal degrees of freedom / degrees of freedom of joints
- Total (maximum) degrees of freedom including the external degrees of freedom / "free flyer" (6 DOF in 3D)
- "Actual" degrees of freedom depending on the particular situation (contacts etc.)



The HRP-2 robot has

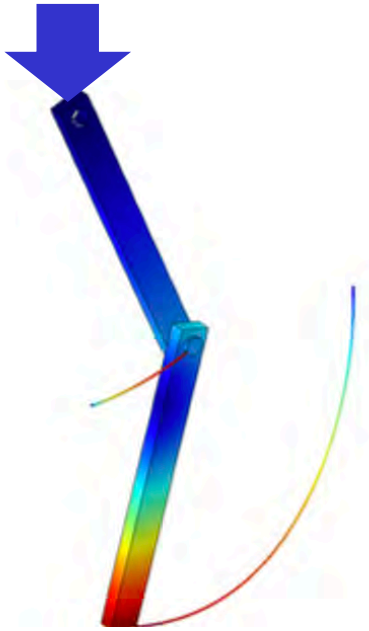
- 30 internal DOF
- 36 in total max

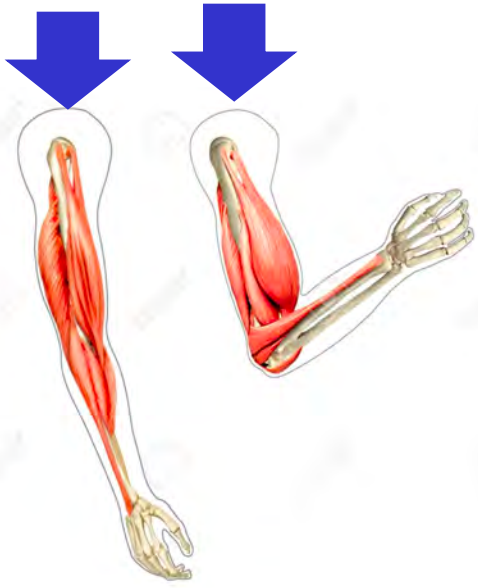
Actual degrees of freedom depending on contacts

	Contact with 1 foot on flat floor	Contact with 2 feet on flat floor
	15	9
	30	24

There are different kinematic structures: 1. (open) kinematic chain

Examples: (double / triple /...) pendulum, industrial robot



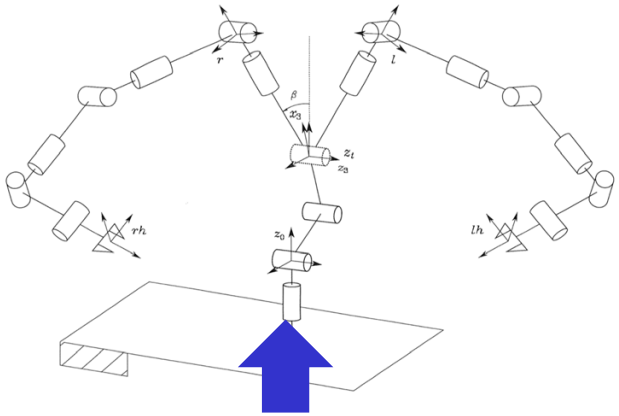


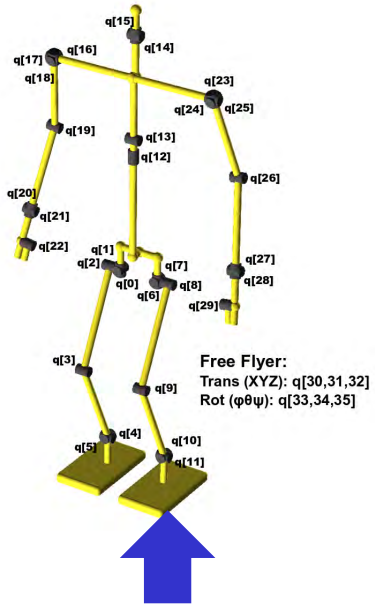
The chain has got a start and end point; there is only one series of joints from start to end

There are different kinematic structures: 2. kinematic tree

Examples:

- **Upper body of a humanoid**
- **walking human / robot with ONE foot on the ground**

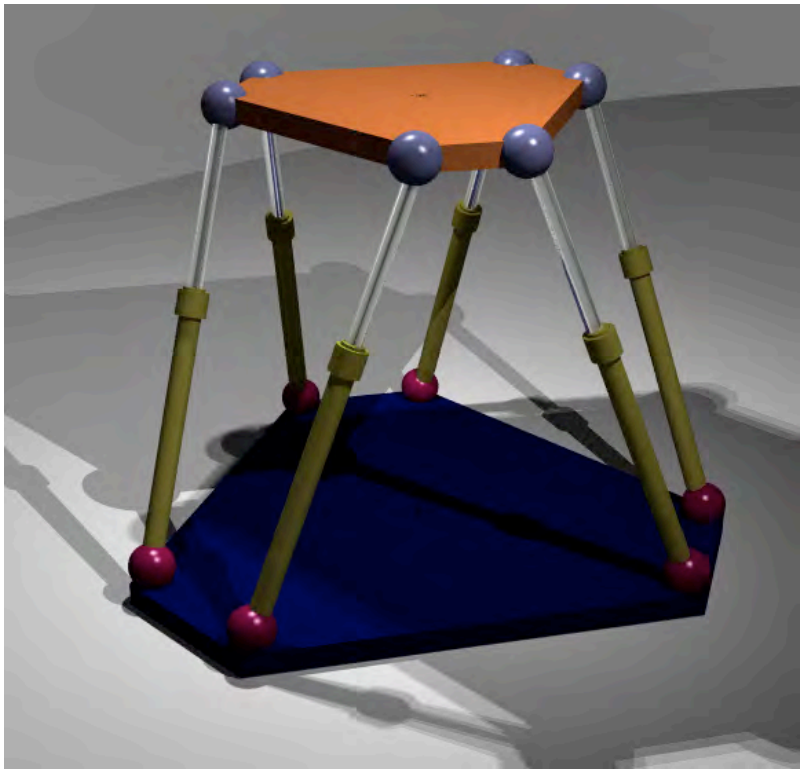




Here several end points exist (branching), but only one root

There are different kinematic structures: 3. Kinematic loops (closed kinematic chain)

Examples

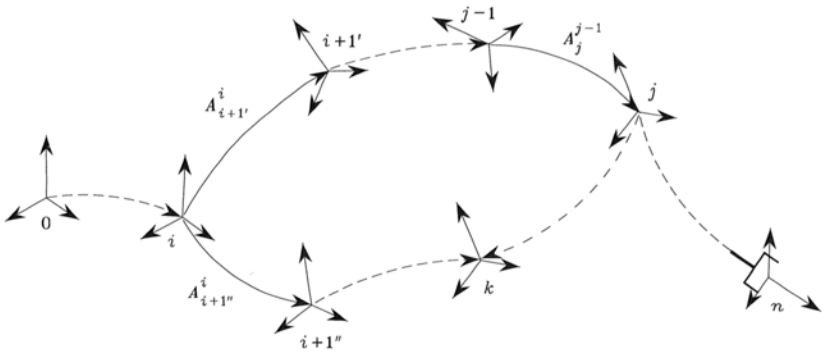


Stewart platform or other robots with parallel kinematics

Walking human / ~~robot~~ with TWO feet on the ground

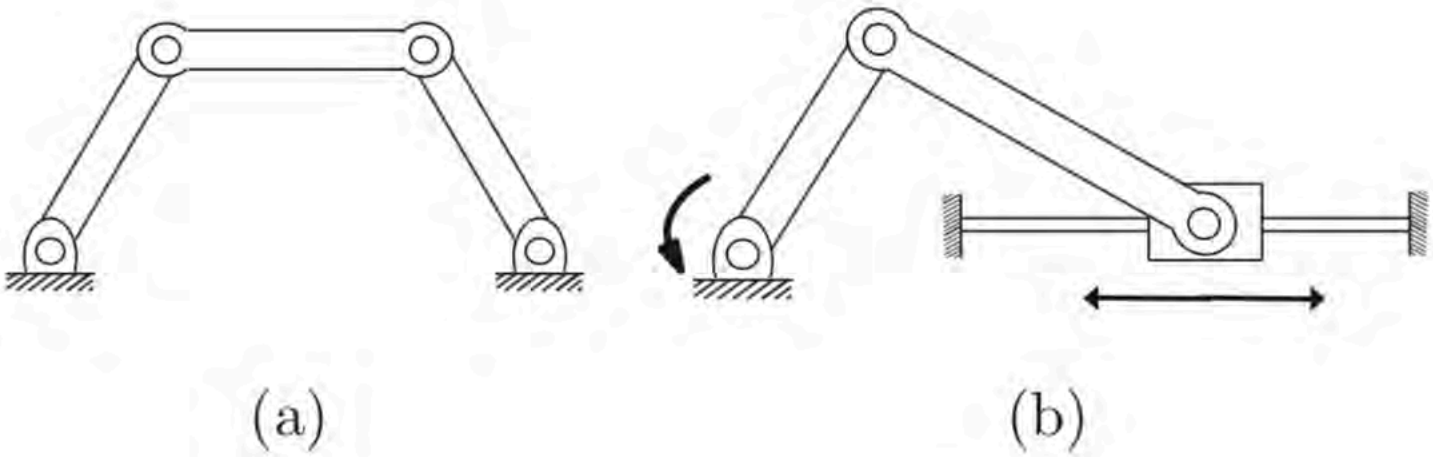
How is redundancy of coordinates linked to the kinematic structure of the system

- In principle all kinematics structures can be formulated by either redundant or non-redundant coordinates – according to taste, algorithm used etc.
- In the case of kinematic loops a formulation with redundant coordinates may come natural:



  - Loop closure is formulated by constraints
  - Constraints as well as all redundant coordinates of open chain / tree can be kept
  - Alternatively constraints could be used for coordinate elimination

How many degrees of freedom do these bodies with closed kinematic loops have?



(a) Four-bar linkage

(b) Slider–crank mechanism.

(a) DOF = 1

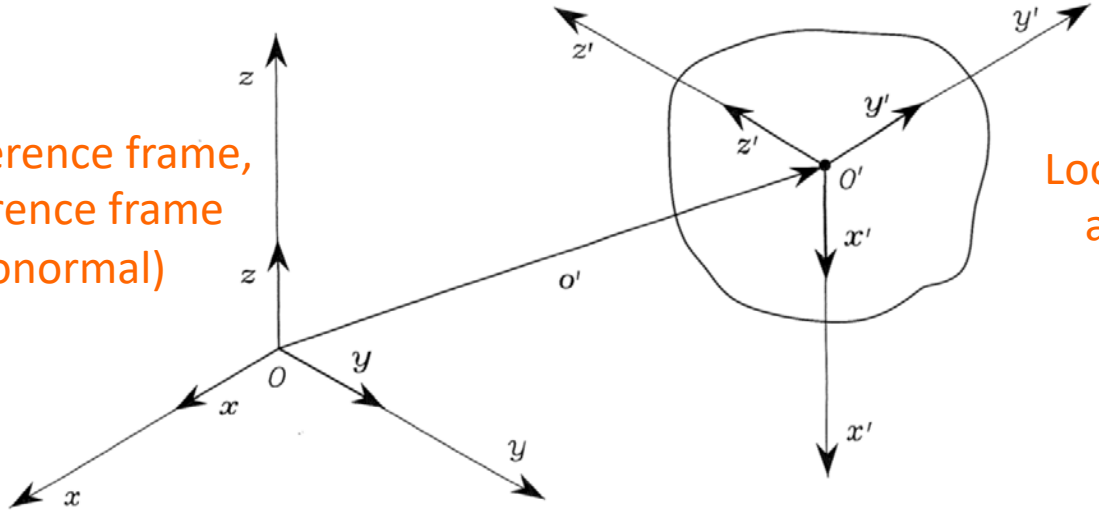
(b) DOF = 1

What would possible minimal and redundant coordinates be?

Katja Mombaur

Reminder: Description of a rigid body in space

Global reference frame, base reference frame (orthonormal)



Local reference frame attached to body (orthonormal)

- A rigid body is completely described in space by
  - its position $\boldsymbol{o}' = o'_x \boldsymbol{x} + o'_y \boldsymbol{y} + o'_z \boldsymbol{z}, \quad \boldsymbol{o}' = \begin{bmatrix} o'_x \\ o'_y \\ o'_z \end{bmatrix}$

    $\boldsymbol{o}' \in \mathbb{R}^3$

    o' is a bound vector
  - its orientation with respect to a global reference frame (expressed by the orientation of the local frame attached to the body)

$$\boldsymbol{x}' = x'_x \boldsymbol{x} + x'_y \boldsymbol{y} + x'_z \boldsymbol{z}$$
$$\boldsymbol{y}' = y'_x \boldsymbol{x} + y'_y \boldsymbol{y} + y'_z \boldsymbol{z}$$
$$\boldsymbol{z}' = z'_x \boldsymbol{x} + z'_y \boldsymbol{y} + z'_z \boldsymbol{z}.$$

Rotation matrix

- x‘, y‘ and z‘ are unit vectors that are orthogonal to each other (orthonormal basis)

$$\boldsymbol{x}'^T\boldsymbol{x}' = 1 \qquad \boldsymbol{y}'^T\boldsymbol{y}' = 1 \qquad \boldsymbol{z}'^T\boldsymbol{z}' = 1$$

$$\boldsymbol{x}'^T\boldsymbol{y}' = 0 \qquad \boldsymbol{y}'^T\boldsymbol{z}' = 0 \qquad \boldsymbol{z}'^T\boldsymbol{x}' = 0.$$

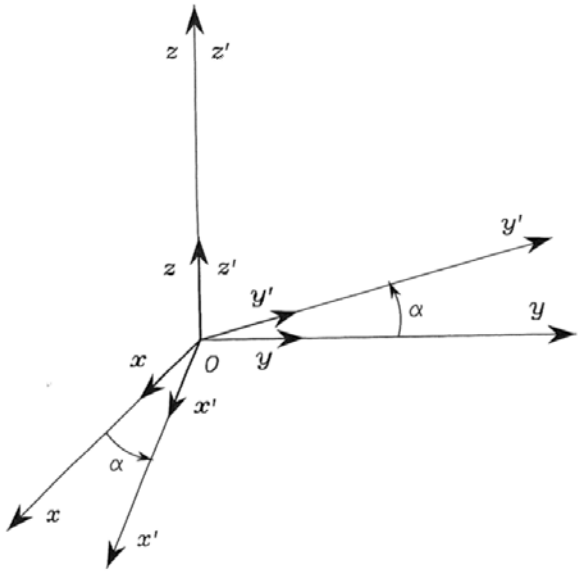
- The 3 vectors can be combined to a rotational matrix

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{x}' & \boldsymbol{y}' & \boldsymbol{z}' \end{bmatrix} = \begin{bmatrix} x'_x & y'_x & z'_x \\ x'_y & y'_y & z'_y \\ x'_z & y'_z & z'_z \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}'^T\boldsymbol{x} & \boldsymbol{y}'^T\boldsymbol{x} & \boldsymbol{z}'^T\boldsymbol{x} \\ \boldsymbol{x}'^T\boldsymbol{y} & \boldsymbol{y}'^T\boldsymbol{y} & \boldsymbol{z}'^T\boldsymbol{y} \\ \boldsymbol{x}'^T\boldsymbol{z} & \boldsymbol{y}'^T\boldsymbol{z} & \boldsymbol{z}'^T\boldsymbol{z} \end{bmatrix}$$

- R is an orthogonal matrix $\boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}_3 \qquad \boldsymbol{R}^T = \boldsymbol{R}^{-1}$
- The rotation matrices belong to the special orthonormal group SO(3) of the real 3x3 matrices with orthonormal columns and determinant = 1.
  - For planar rotations SO(2) (2x2 matrix)

Elementary rotations in 3D

- Frame O‘ can be generated from frame O by an elementary rotation α about the z axis



- This is expressed by the rotation matrix

$$\boldsymbol{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Corresponding rotation matrices for rotations β about the y axis and γ about the x axis:

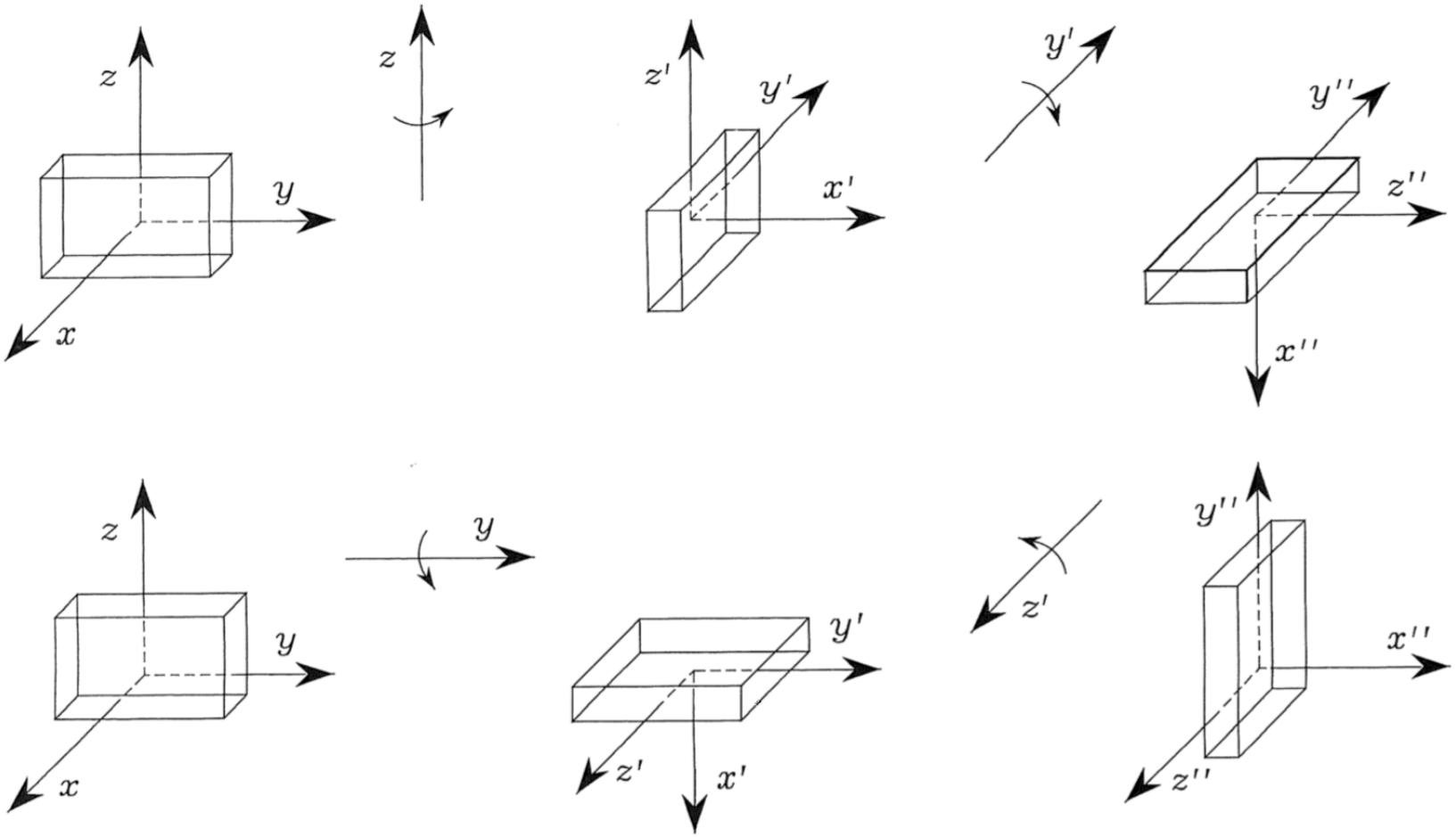
$$\boldsymbol{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \qquad \boldsymbol{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

- These rotation matrices obviously have the following property:

$$\boldsymbol{R}_k(-\vartheta) = \boldsymbol{R}_k^T(\vartheta) \qquad k = x, y, z$$

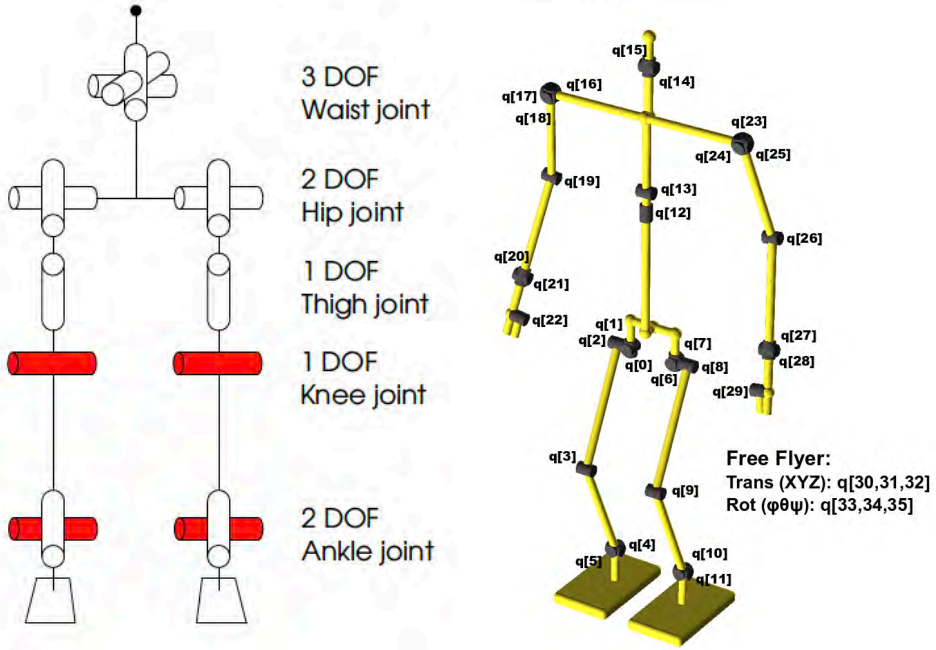
Successive rotations (about local frames)

The order of elementary rotations is important! (matrix multiplication is not commutative)



Robot and human modeling

- There are joints in robots with multiple degrees of freedom



- Sometimes there is a mechanical offset, sometimes there isn't (depending on mechanical joint type)
- When analyzing (or providing) simulation data of motions, it is important to know the order of rotations in each joint!

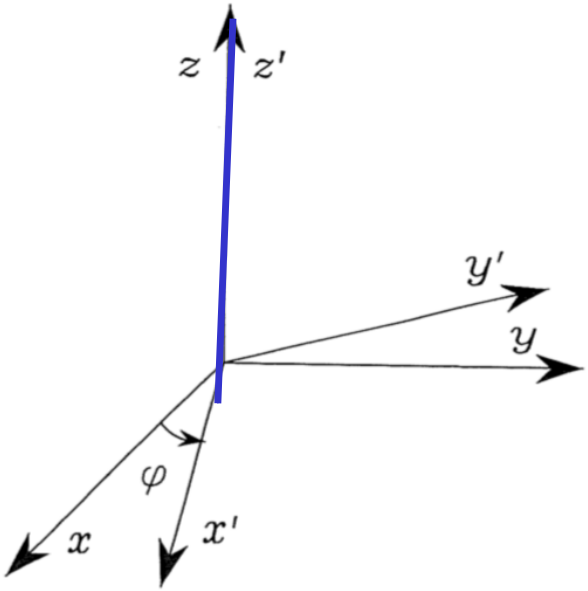
A possible coordinate choice: Euler angles

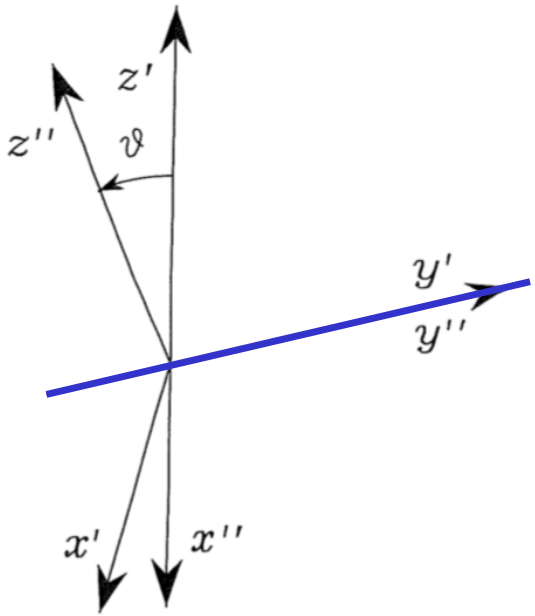
- A minimal description of the orientation of a body with respect to another one can be obtained by three angles:

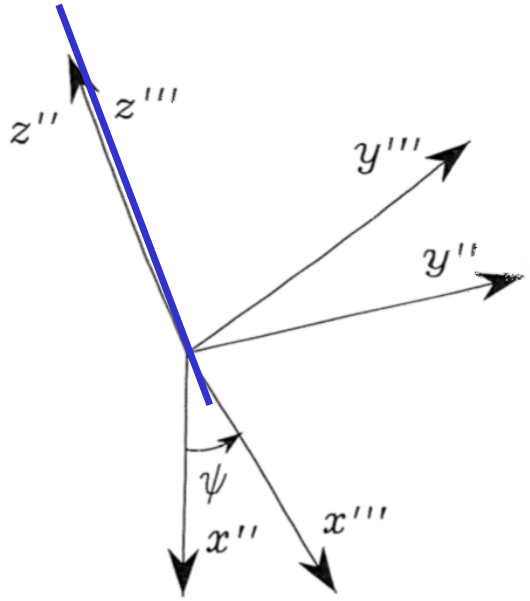
$$\Phi = [\phi \quad \theta \quad \Psi]$$

- a general rotation can be achieved by performing a sequence of three elementary rotations about coordinate axes where the axes of two subsequent rotations are not allowed to be parallel
- Due to this condition, only 12 theoretical possibilities of of 27 remain. These are the Euler angles.
  - *z-x-z, x-y-x, y-z-y, z-y-z, x-z-x, y-x-y*
    x-y-z, y-z-x, z-x-y, x-z-y, z-y-x, y-x-z
- Rotations can be performed about axes attached to the body and fixed axes

ZYZ – Euler angles







$$\boldsymbol{R}(\boldsymbol{\phi}) = \boldsymbol{R}_z(\varphi)\boldsymbol{R}_{y'}(\vartheta)\boldsymbol{R}_{z''}(\psi)$$

$$= \begin{bmatrix} c_\varphi c_\vartheta c_\psi - s_\varphi s_\psi & -c_\varphi c_\vartheta s_\psi - s_\varphi c_\psi & c_\varphi s_\vartheta \\ s_\varphi c_\vartheta c_\psi + c_\varphi s_\psi & -s_\varphi c_\vartheta s_\psi + c_\varphi c_\psi & s_\varphi s_\vartheta \\ -s_\vartheta c_\psi & s_\vartheta s_\psi & c_\vartheta \end{bmatrix}$$

c: cos
s: sin

ZYZ Euler angles – Inverse problem

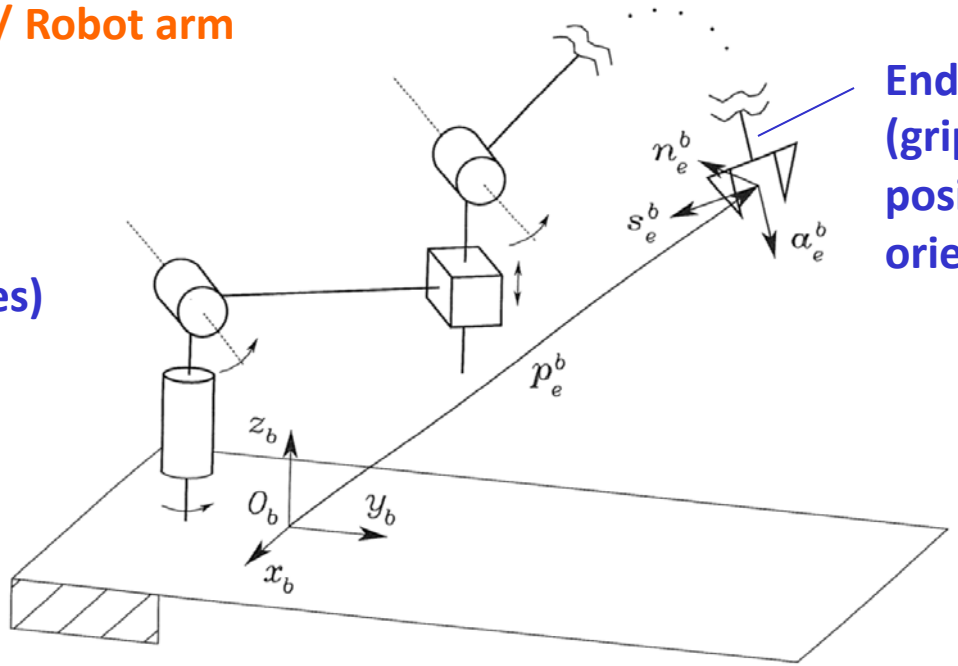
- Often one wants to solve the inverse problem – i.e. determine the Euler angles that generate a given rotation matrix:
- This can be done by comparison with R from previous slide with the given R
- Solving it for the thee angles is usually possible except for singularities
  - check gimbal lock problem
  - occurs in the example if second rotation is zero
  - For other choices of Euler angles, singularities appear in different angles and at different values

Consequences for the choice of angles in robot and human modeling

- Consider the range of motion in the different directions and choose the Euler angles such that the
  - E.g. Euler angles for total orientation of diver in space

Reminder: Forward kinematics vs. Inverse kinematics

Manipulator arm / Robot arm



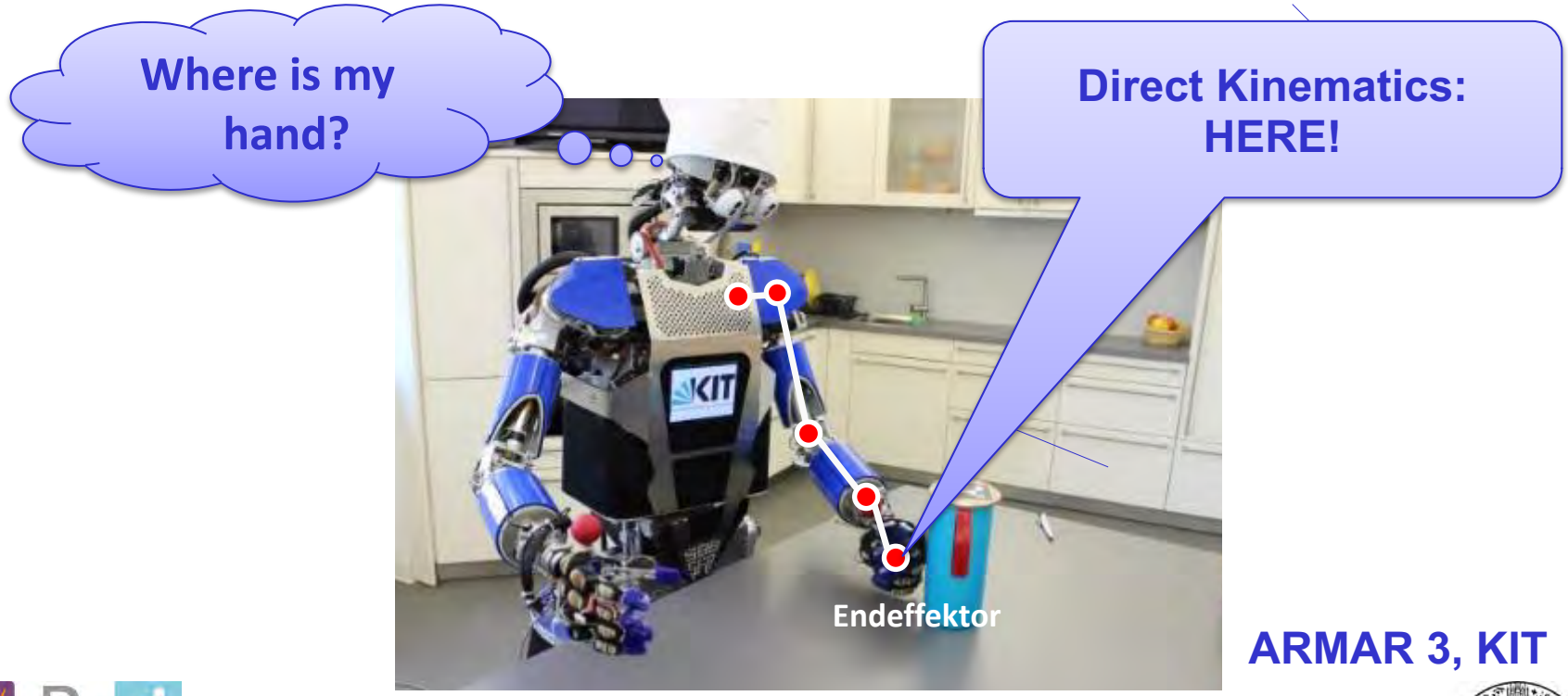
End effector (gripper, tool) position and orientation

Joint variables (here joint angles)

- **Forward kinematics (direct kinematics):**
  Which end effector position and orientation do we obtain for a given set of joint angles (coordinates)?
- **Inverse kinematics:**
  Which joint angles are required for a given end effector position and orientation?

Forward kinematics

- **Direct** kinematic problem (Forward kinematics)
  - **Determine the position (+ orientation) of the end effector** if the joint angles of the robot are given



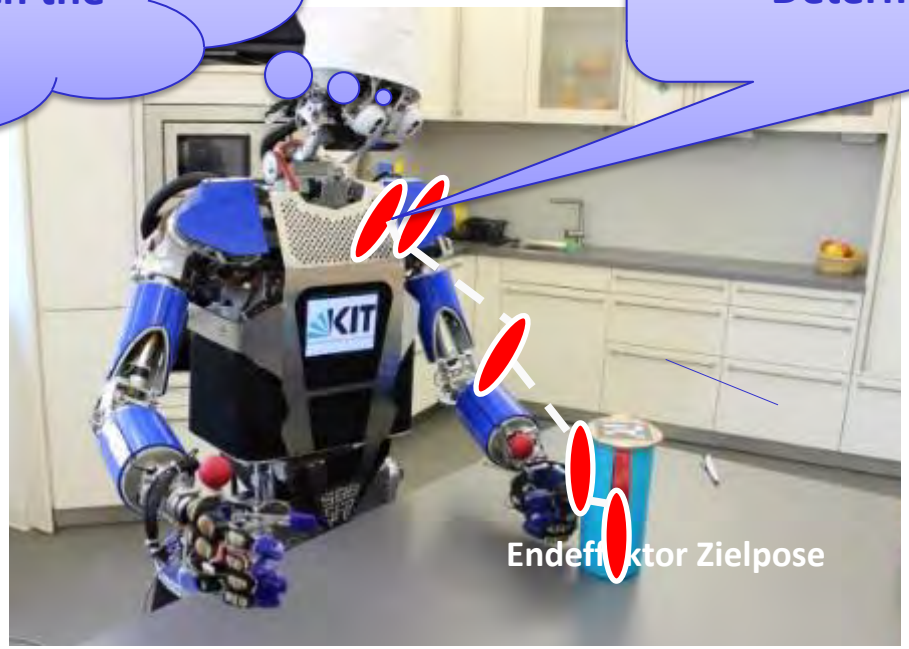
ARMAR 3, KIT

Inverse Kinematics

- **Inverse** kinematic problem (Backward kinematics)
  - **Determine joint angles** that result in a desired position (and orientation) of the end effector

Which joint angles are needed to reach the cup?

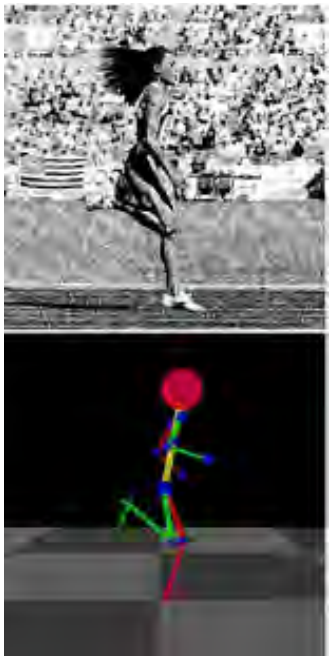
Inverse Kinematics: Determines joint angles

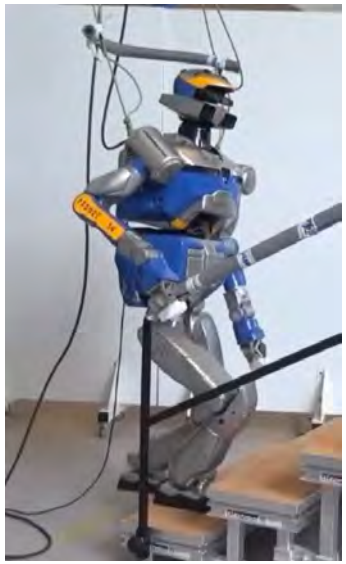


ARMAR 3, KIT

Example for inverse kinematics problem to be solved in the context of humanoid robots and humans

- At which configuration of the robot/human (e.g. joint angle configuration, depending on the choice of coordinates) does the foot enter in contact with with ground?
- At which configuration does the robot touch the handrail ?





Inverse kinematics conditions in multibody system

- Can appear in phase switching conditions

$$s(q(\tau_s), v(\tau_s), p) = 0.$$

- Can also appear in the context of equality constraints that are valid an entire phase (see later)

$$g_{pos} = g(q(t), p) = 0$$
$$g_{vel} = G(q(t), p) \cdot \dot{q}(t) = 0.$$

- Can also appear in inequality constraints of all types, e.g. collision avoidance etc

Inverse kinematics problem

- **Determine required joint angles for a desired end effector position and orientation**

- **Inverse kinematics problem is more difficult than the forward kinemattics problem**:
  - A closed (i.e. analytic) solution can not always be found (but might only be computable numerically
  - It is possible that multiple solutions exist
  - For redundant kinematic structures, an infinite number of solutions may exist
  - It is also possible that no solution exists (if required configuration is outside the work space)

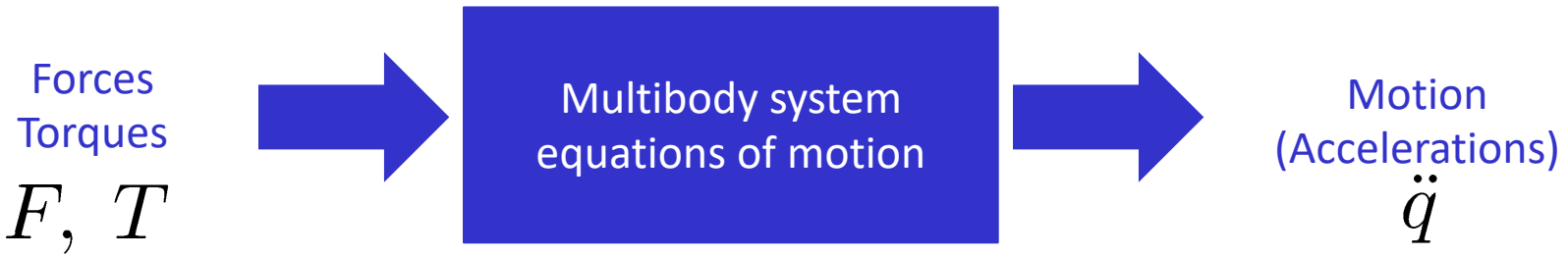
In the context of optimization / optimal control, a potential redundancy in inverse kinematics is usually resolved since in addition an objective function is optimized

The infeasibility obviously continues to exists also in the optimization context

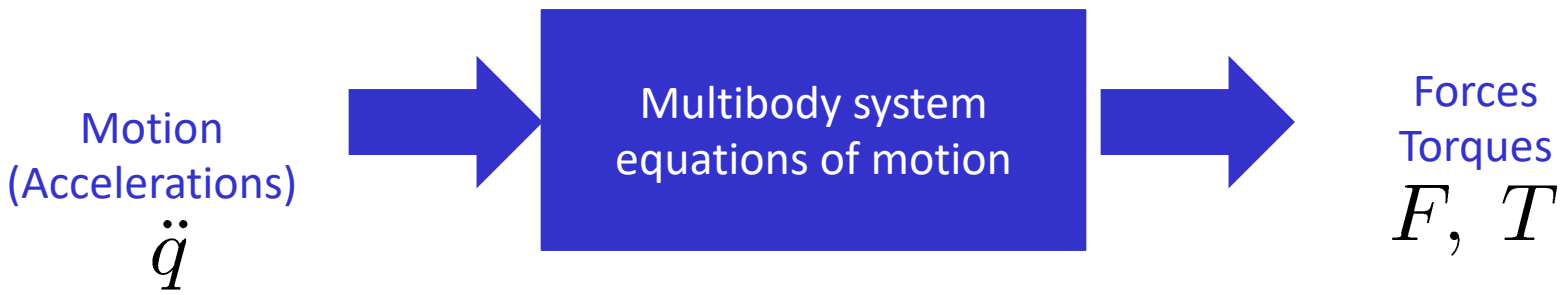
Multibody system dynamics with minimal and redundant coordinates

Different tasks in dynamics

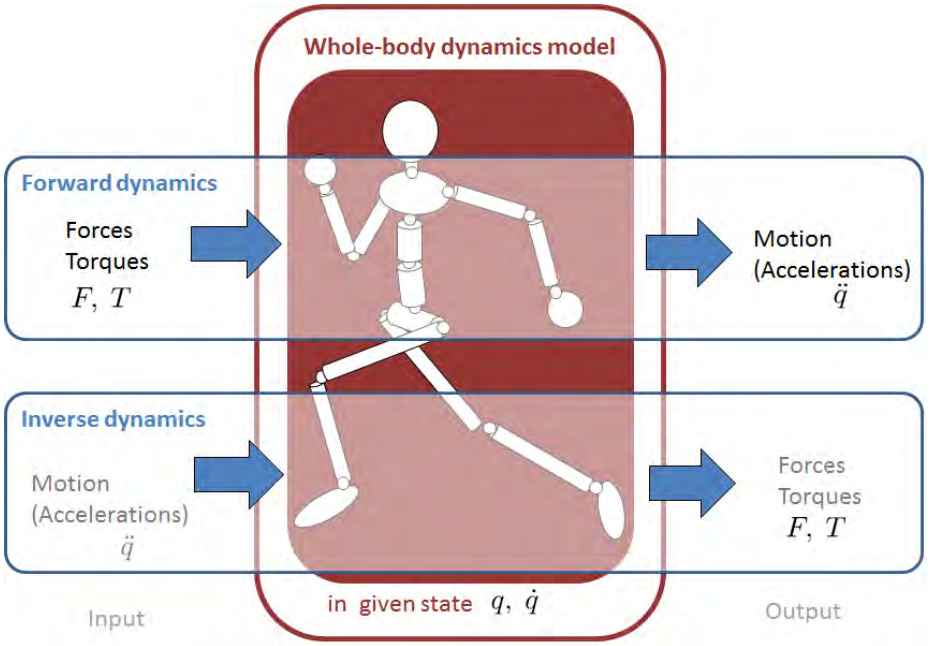
- **System in a given state** $q,\ \dot{q}$
- **Forward dynamics**



- **Inverse Dynamics**



Same model is used, but in different directions



- In simulation, the input side must always be known, and then the output side is determined
- In optimal unknown inputs and outputs optimizing a cost function are determined at the same time
- The direction in which the model is used, determines the choice of state and control variables

Reminder: Equations of motion for rigid body systems

- Newton‘s law (translations)

$$F = ma \qquad F = \sum_{i=1}^{n_T} F_i$$

- Newton‘s law (rotations)

$$T = \Theta\dot{\omega} \qquad T = \sum_{i=1}^{n_T} T_i$$

Two fundamentally different methods for setting up equations of motion of multibody systems

Lagrange equations / Euler-Lagrange

- Is based on a study of the total energy of the entire multibody system
- Is conceptually very simple
- Is getting very complex in practice already for slightly complicated systems

Newton-Euler / Balance of forces

- Is based on a cosideration of the individual bodies of the multibody systems
- The segments are virtualy separated from each other, and Newton‘s equations are set up individually for each body (free body diagrams)
- Recursive computations are possible; efficient

Lagrange equations of the 2nd kind (Euler-Lagrange method)

- **Lagrange function:** $\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathcal{T}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \mathcal{U}(\boldsymbol{q})$

  Kinetic Energy — Potential Energie

- **(Euler-)Lagrange equation** for every component (generalized coordinate = minimal)

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = \xi_i \qquad i = 1, \ldots, n$$

ξ_i is the generalized force corresponding to coordinate q_i

- **(Euler-)Lagrange equations for all coordinates:**

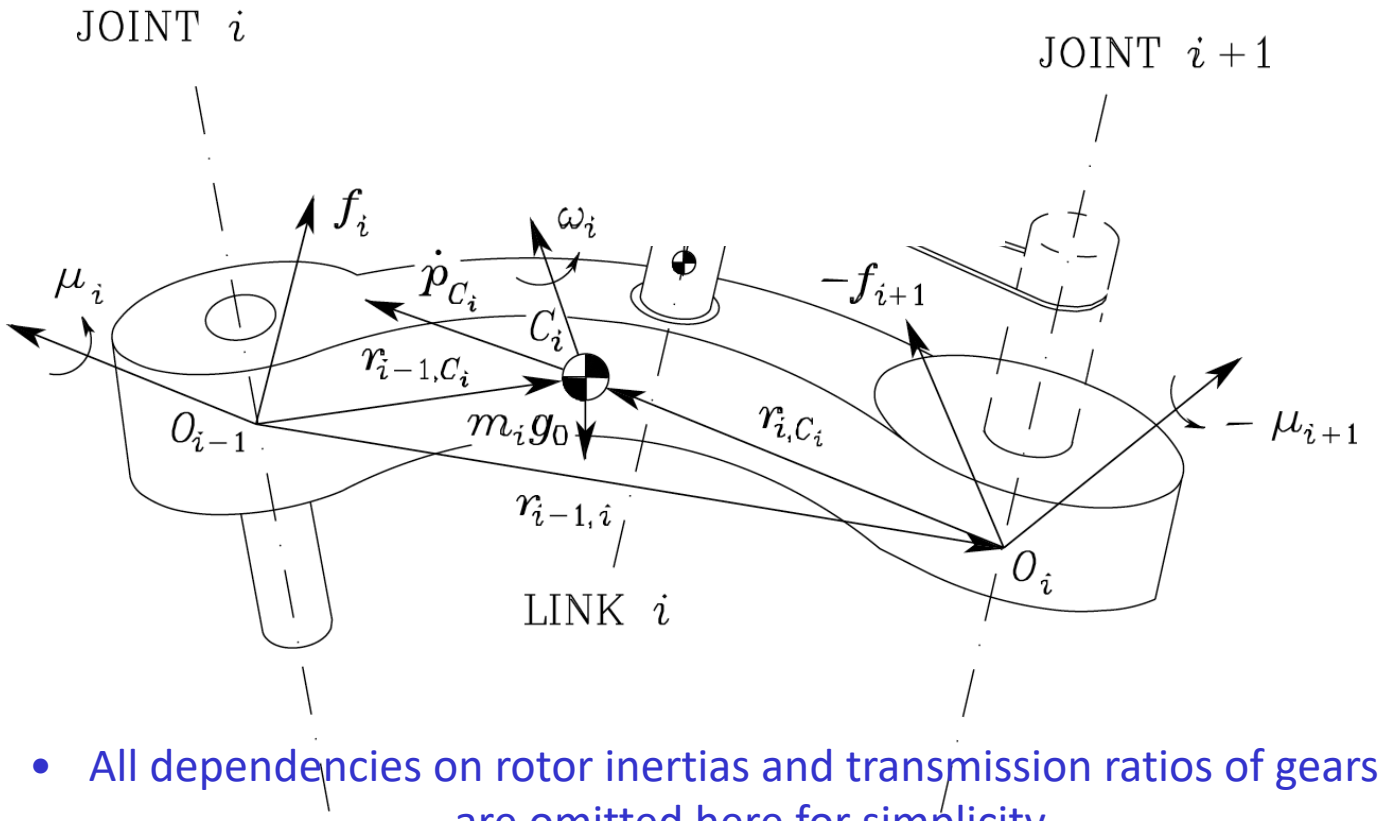
$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}}\right)^T - \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{q}}\right)^T = \boldsymbol{\xi}$$

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\xi}$$

We will discuss the version for redundant coordinates shortly

A general scheme of Newton-Euler

- Cutting free body I formulating the balance equations



- All dependencies on rotor inertias and transmission ratios of gears etc. are omitted here for simplicity

Balance equations (equations of motion of segments)

- Translation (Newton)

$$\boldsymbol{f}_i - \boldsymbol{f}_{i+1} + m_i \boldsymbol{g}_0 = m_i \ddot{\boldsymbol{p}}_{C_i}$$

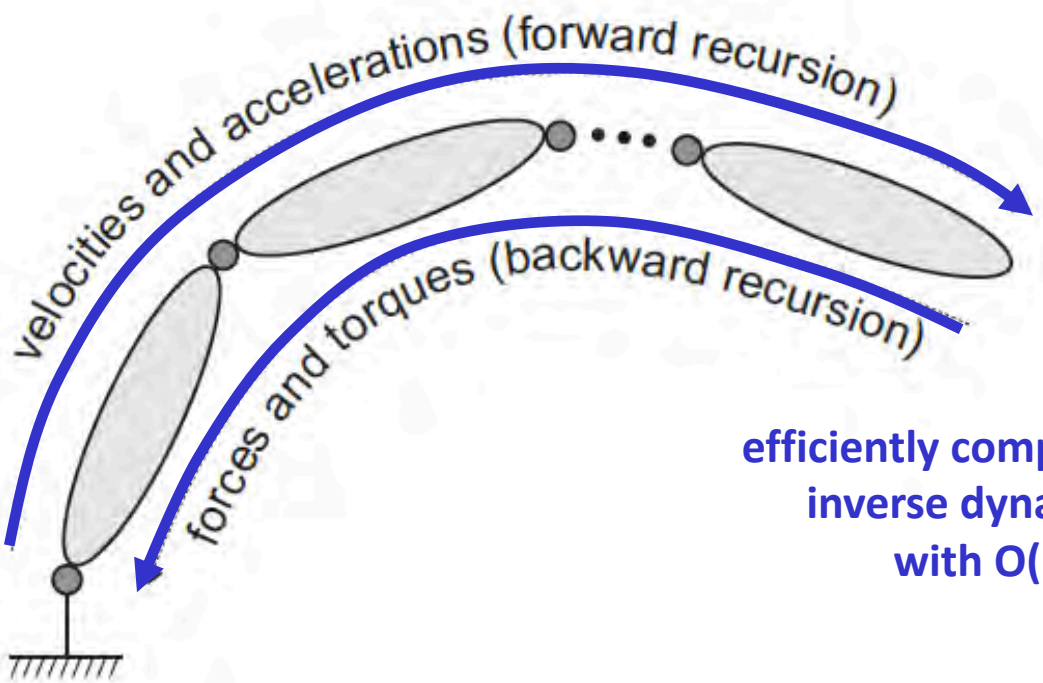
- Rotation (Euler)

$$\boldsymbol{\mu}_i + \boldsymbol{f}_i \times \boldsymbol{r}_{i-1,C_i} - \boldsymbol{\mu}_{i+1} - \boldsymbol{f}_{i+1} \times \boldsymbol{r}_{i,C_i} = \frac{d}{dt}(\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i)$$

with

$$\begin{aligned}
\frac{d}{dt}(\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i) &= \dot{\boldsymbol{R}}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \dot{\boldsymbol{R}}_i^T \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \dot{\boldsymbol{\omega}}_i \\
&= \boldsymbol{S}(\boldsymbol{\omega}_i) \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \boldsymbol{S}^T(\boldsymbol{\omega}_i) \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \dot{\boldsymbol{\omega}}_i \\
&= \bar{\boldsymbol{I}}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i)
\end{aligned}$$

Newton-Euler recursions



efficiently computes the inverse dynamics with O(n)

2 recursions

- **Forward recursion:** Determines positions, velocities and accelerations of all individual segments (iterating outwards from segment to segment) for given generalized coordinates and their derivatives
- **Backward recursion:** Determines generalized forces as a result of the external forces and inertial forces (iterating inwards)

Lagrange equations of the 1st kind (Euler-Lagrange method for redundant coordinates)

- Description of the system with redundant coordinates q (dim of q > # DOF), which are coupled by constraints:

$$g(q) = 0$$

- Lagrange equations of the 1st kind :

$$\frac{d}{dt}\left(\frac{\partial L(q,\dot{q})}{\partial \dot{q}}\right) - \frac{\partial L(q,\dot{q})}{\partial q} + (G(q)^T \lambda)^T = \xi(t,q,\dot{q})$$

Additional term that considers the forces produced by the constraints (constraints forces)
G(q) is Jacobian of g(q)

- Additionally, the constraints must be satisfied:

$$g(q) = 0$$

Equations of motion for redundant coordinates

- The equations of motion are not any more described by an ODE, as in the case of minimal coordinates, but by a DAE
- For mechanical systems, this DAE is

$$\begin{aligned} M(q(t),p)\cdot a &= f(q(t),v(t),u(t),p) - G^T(q(t),p)\lambda \\ g_{pos}(q(t),p) &= 0 \end{aligned}$$

  - Which index does this DAE have?

Mechanical DAE (last slide) has got index 3

- Can be transformed into index 1 DAE by index reduction (differentiating twice with respect to time), also called descriptor form

Differential part

$$\begin{aligned} \dot{q}(t) &= v(t) \\ \dot{v}(t) &= a(t) \end{aligned}$$

Algebraic part

$$\begin{pmatrix} M(q(t),p) & G^T(q(t),p) \\ G(q(t),p) & 0 \end{pmatrix} \begin{pmatrix} a \\ \lambda \end{pmatrix} = \begin{pmatrix} f(q(t),v(t),u(t),p) \\ \gamma(q(t),v(t),p) \end{pmatrix}$$

Invariants

$$\begin{aligned} g_{pos} &= g(q(t),p) = 0 \\ g_{vel} &= G(q(t),p) \cdot \dot{q}(t) = 0. \end{aligned}$$

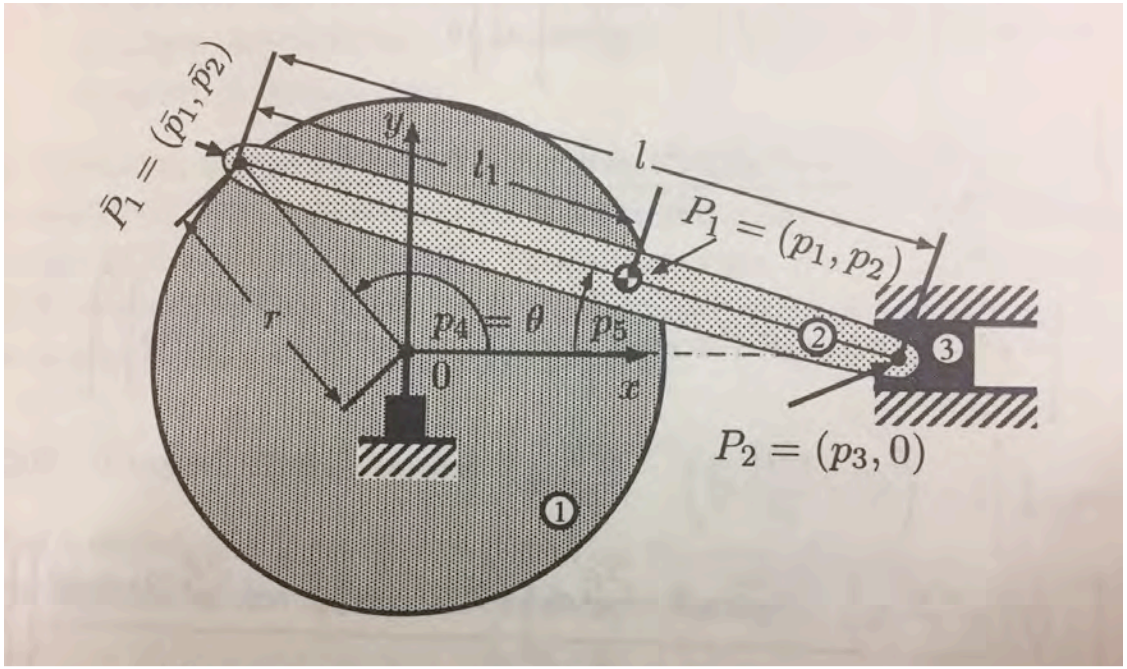
- One further differentiation of the linear system of equations (the algebraic part) would transform the whole system into an ODE (which justifies the statement that the original system was an index 3 DAE)
- Can also be treated in the above form by solving the linear system and inserting the solution (in particular the a) into the differential part

Reasons to use descriptor form

- Why would one want to use more coordinates / variables than necessary?
- This may result in simpler and more intuitive equations of motions despite (see example on next slide)
- In the context of changing contacts (such as in walking and running or many other types of robot motions), it may be desirable to keep the same set of coordinates / differential variables throughout the motion and only add / remove constraint equations and the corresponding terms

Example: Planar slider crank mechanism

- comparison of models in minimal coordinates and redundant coordinates (example from Alishenas / Haug, taken from PhD thesis R. v. Schwerin) similar to slider crank presented before



1 DOF

- Minimal coordinates: 1 variable, e.g. angle θ
- Redundant coordinates: use 5 variables shown in picture)

Minimal coordinate model

$$\left\{ J_1 + J_2 \left[\frac{r^2 c_\theta^2}{l^2 - r^2 s_\theta^2} \right] + m_2 r^2 \left[\frac{1 - l_1^2}{l^2} \right] c_\theta^2 \right.$$

$$\left. + \quad m_2 \left[r + \left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} c_\theta \right] s_\theta^2 \right\} \ddot{\theta}$$

$$= \left\{ J_1 + J_2 \left[\frac{-(l^2 - r^2 s_\theta^2) 2 r^2 c_\theta s_\theta \dot{\theta} + r^2 c_\theta^2 2 r^2 c_\theta s_\theta \dot{\theta}}{(l^2 - r^2 s_\theta^2)^2} \right] \right.$$

$$+ m_2 \left[-r^2 \frac{1 - l_1^2}{l^2} 2 c_\theta s_\theta \dot{\theta} + 2 c_\theta s_\theta \dot{\theta} \left[r + \left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} c_\theta \right]^2 \right]$$

$$+ 2 m_2 s_\theta^2 \left[r + \left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} c_\theta \right] \tag{1.5.1}$$

$$\left. \times \left[\frac{-\left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} s_\theta \dot{\theta} - \frac{r^2 l_1^2}{l^2} c_\theta \frac{1}{2} \left(l^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{3}{2}} - \frac{r^2 l_1^2}{l^2} (2 s_\theta c_\theta) \dot{\theta}}{l_1^2 - \frac{r^2}{l^2} s_\theta^2} \right] \right\} \dot{\theta}$$

$$+ \left\{ J_1 + J_2 \left[\frac{-(l^2 - r^2 s_\theta^2) 2 r^2 c_\theta s_\theta + r^2 c_\theta^2 2 r^2 c_\theta s_\theta}{(l^2 - r^2 s_\theta^2)^2} \right] \right.$$

$$+ m_2 \left[-r^2 \frac{1 - l_1^2}{l^2} 2 c_\theta s_\theta + 2 c_\theta s_\theta \left[r + \left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} c_\theta \right]^2 \right]$$

$$+ 2 m_2 s_\theta^2 \left[r + \left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} c_\theta \right]$$

$$\left. \times \left[\frac{-\left(l_1^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{1}{2}} \frac{r^2 l_1^2}{l^2} s_\theta - \frac{r^2 l_1^2}{l^2} c_\theta \frac{1}{2} \left(l^2 - \frac{r^2 l_1^2}{l^2} s_\theta^2 \right)^{-\frac{3}{2}} - \frac{r^2 l_1^2}{l^2} 2 s_\theta c_\theta}{l_1^2 - \frac{r^2}{l^2} s_\theta^2} \right] \right\} \frac{\dot{\theta}^2}{2}$$

$$+ n_1 - m_2 g r \left(1 - \frac{l_1}{l} \right) c_\theta$$

where

$$s_\theta := \sin\theta \qquad \text{and} \qquad c_\theta := \cos\theta \,.$$

Thus, even this little

Model with redundant coordinates

- Part 1

$$\mathcal{A} \cdot \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix} = \begin{pmatrix} 0 \\ -m_2 g \\ 0 \\ n_1 \\ 0 \\ -r c_{p_4}\, v_4^2 - l_1 c_{p_5}\, v_5^2 \\ r s_{p_4}\, v_4^2 - l s_{p_5}\, v_5^2 \\ -(l-l_1) c_{p_5}\, v_5^2 \\ -r c_{p_4}\, v_4^2 - l c_{p_5}\, v_5^2 \end{pmatrix}$$

where

$$s_{p_i} := \sin(p_i) \qquad \text{and} \qquad c_{p_i} := \cos(p_i)\,, \quad i = 4, 5\,.$$

and

$$\mathcal{A} = \left(\begin{array}{ccccc|cccc} m_2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & m_3 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & J_1 & 0 & r s_{p_4} & r c_{p_4} & 0 & r s_{p_4} \\ 0 & 0 & 0 & 0 & J_2 & l_1 s_{p_5} & -l c_{p_5} & -(l-l_1) c_{p_4} & l s_{p_5} \\ \hline 1 & 0 & 0 & r s_{p_4} & l_1 s_{p_5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r c_{p_4} & -l c_{p_5} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -(l-l_1) c_{p_4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & r s_{p_4} & l s_{p_5} & 0 & 0 & 0 & 0 \end{array}\right)$$

is

Model with redundant coordinates

- Part 2
  - Position constraints

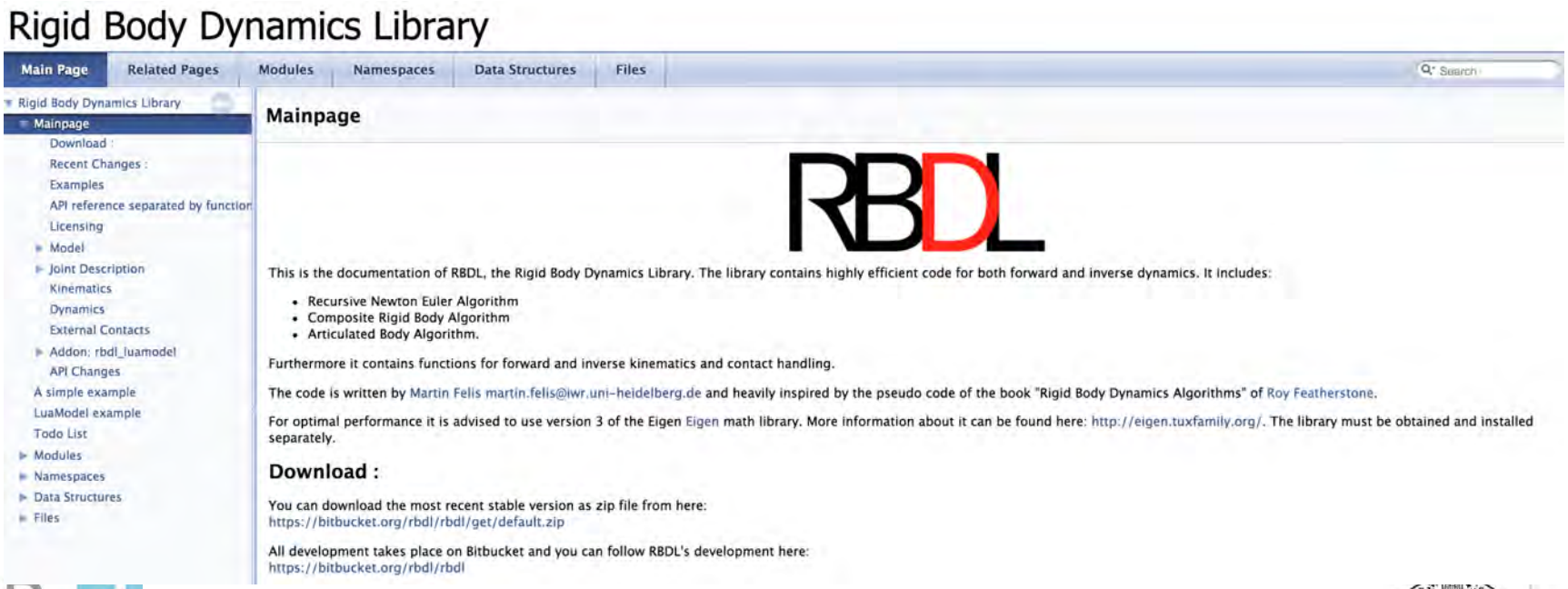
$$
\begin{aligned}
p_1 - r c_{p_4} - l_1 c_{p_5} &= 0 \\
p_2 - (l - l_1) s_{p_5} &= 0
\end{aligned}
\qquad
\begin{aligned}
r s_{p_4} - l s_{p_5} &= 0 \\
p_3 - r c_{p_4} - l c_{p_5} &= 0
\end{aligned}
$$

  - Velocity constraints

$$
\begin{aligned}
v_1 + r s_{p_4}\, v_4 + l_1 s_{p_5}\, v_5 &= 0 \\
v_2 - (l - l_1) c_{p_5}\, v_5 &= 0
\end{aligned}
\qquad
\begin{aligned}
r c_{p_4}\, v_4 - l c_{p_5}\, v_5 &= 0 \\
p_3 + r s_{p_4}\, v_4 + l s_{p_5}\, v_5 &= 0
\end{aligned}
$$

RBDL – Rigid body dynamics library

- You will not have to set up the equations of motion yourself, but you will be using an efficient tool to do so in the exercises
- Author: Martin Felis, ORB
  An efficient tool for setting up forward and inverse dynamic models



Sources

- Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: Robotics - Modelling, Planning and Control (Springer)
- Tamim Asfour, KIT, Course „Robotik I: Einführung in die Robotik“ (2017/18)
- K. Lynch, F. C. Park: Modern Robotics


Thank you very much for your attention!