



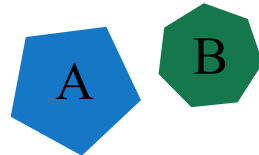
Robotics: Forward kinematics of open chains

Vladimír Petřík

vladimir.petrík@cvut.cz

29.09.2025

Connecting two rigid bodies

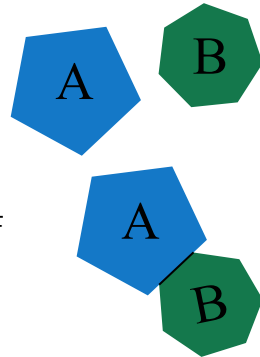


- ▶ How many DoF has system of two planar rigid bodies?



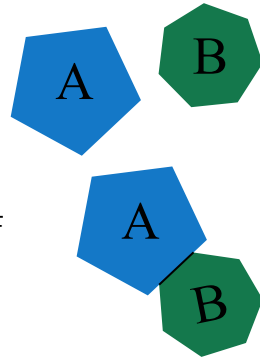
Connecting two rigid bodies

- ▶ How many DoF has system of two planar rigid bodies? $(3 + 3)$ DoF
- ▶ How many DoF if we glue/fix them together?



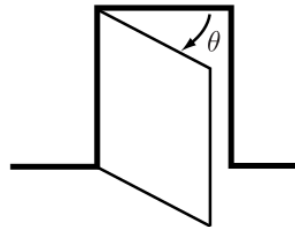
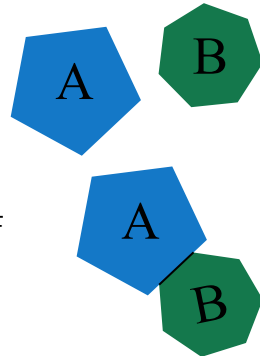
Connecting two rigid bodies

- ▶ How many DoF has system of two planar rigid bodies? $(3 + 3)$ DoF
- ▶ How many DoF if we glue/fix them together? 3 DoF
- ▶ **Fixed joint**
 - ▶ connects two rigid bodies together
 - ▶ removes 3 DoF in planar case and 6 DoF in spatial case



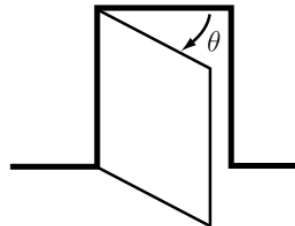
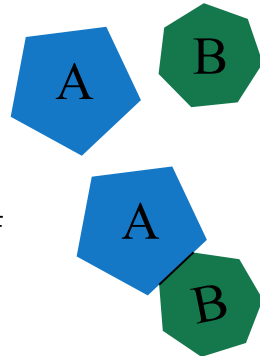
Connecting two rigid bodies

- ▶ How many DoF has system of two planar rigid bodies? $(3 + 3)$ DoF
- ▶ How many DoF if we glue/fix them together? 3 DoF
- ▶ **Fixed joint**
 - ▶ connects two rigid bodies together
 - ▶ removes 3 DoF in planar case and 6 DoF in spatial case
- ▶ How many DoF for door if there is no joint?



Connecting two rigid bodies

- ▶ How many DoF has system of two planar rigid bodies? $(3 + 3)$ DoF
- ▶ How many DoF if we glue/fix them together? 3 DoF
- ▶ **Fixed joint**
 - ▶ connects two rigid bodies together
 - ▶ removes 3 DoF in planar case and 6 DoF in spatial case
- ▶ How many DoF for door if there is no joint?
- ▶ **Revolute joint**
 - ▶ connects two rigid bodies together
 - ▶ has 1 DoF
 - ▶ removes 2 DoF in planar case and 5 DoF in spatial case



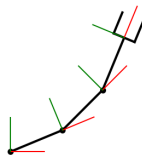
Robotic manipulator

- ▶ Constructed from links (typically rigid bodies)
- ▶ **Two** links are connected by various joints
- ▶ Actuators deliver torque/force to cause link motions
- ▶ End-effector/Gripper is attached to some of the links



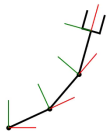
Robotic manipulator

- ▶ Constructed from links (typically rigid bodies)
- ▶ **Two** links are connected by various joints
- ▶ Actuators deliver torque/force to cause link motions
- ▶ End-effector/Gripper is attached to some of the links



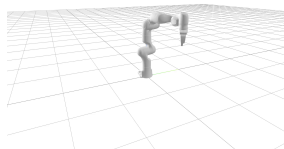
Robotic manipulator

- ▶ Constructed from links (typically rigid bodies)
- ▶ **Two** links are connected by various joints
- ▶ Actuators deliver torque/force to cause link motions
- ▶ End-effector/Gripper is attached to some of the links



Robotic manipulator

- ▶ Constructed from links (typically rigid bodies)
- ▶ **Two** links are connected by various joints
- ▶ Actuators deliver torque/force to cause link motions
- ▶ End-effector/Gripper is attached to some of the links

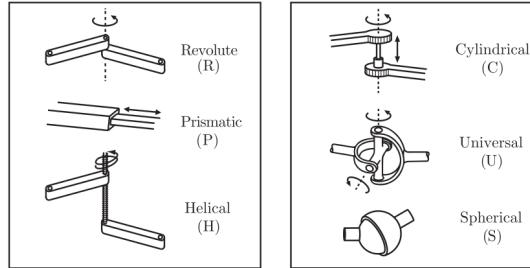


Prismatic joint

- ▶ Also sliding or linear joint
- ▶ Only translation motion in 1 DoF
- ▶ Removes 2 DoF in planar case and 5 DoF in spatial case



Joints types



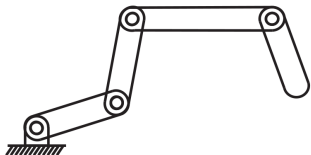
Constraints between two rigid bodies

Joint type	DoF	Planar	Spatial
R	1	2	5
P	1	2	5
H	1	-	5
C	2	-	4
U	2	-	4
S	3	-	3



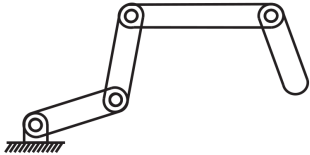
Open/Closed kinematic chain

- ▶ Open kinematics chains: no loops
- ▶ Closed kinematic chains contains loops

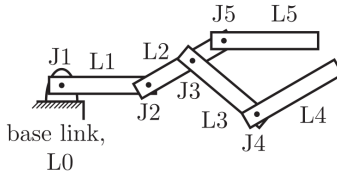


Open/Closed kinematic chain

- ▶ Open kinematics chains: no loops
- ▶ Closed kinematic chains contains loops

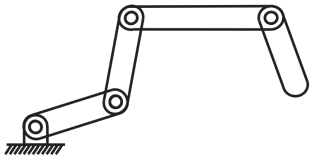


Open - sequential

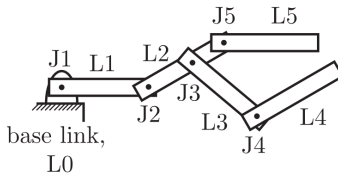


Open/Closed kinematic chain

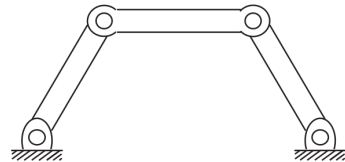
- ▶ Open kinematics chains: no loops
- ▶ Closed kinematic chains contains loops



Open - sequential

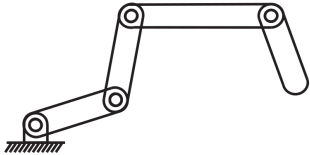


Open - tree

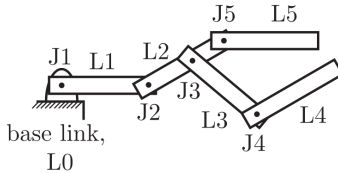


Open/Closed kinematic chain

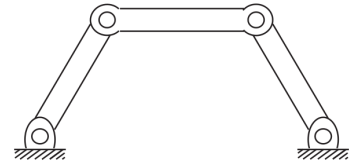
- ▶ Open kinematics chains: no loops
- ▶ Closed kinematic chains contains loops



Open - sequential

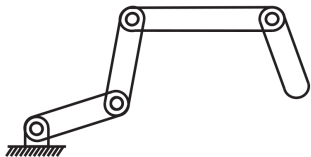


Open - tree

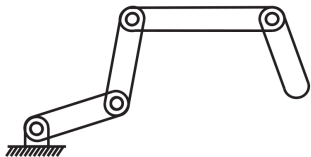


Closed

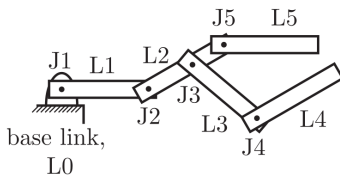
How many DoF?



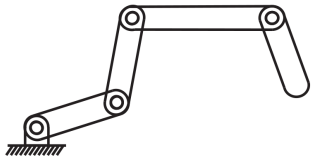
How many DoF?



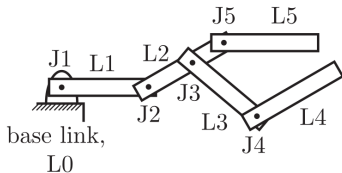
Easy: 4 DoF



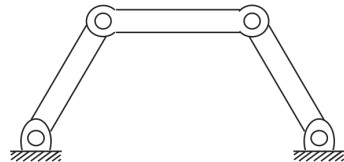
How many DoF?



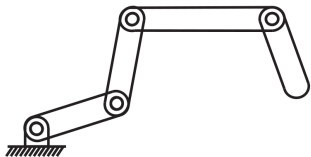
Easy: 4 DoF



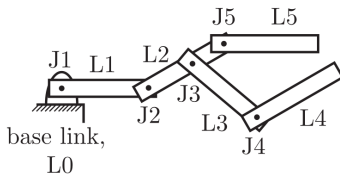
Easy: 5 DoF



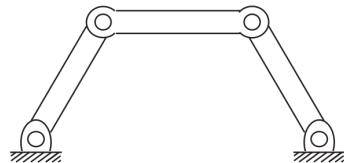
How many DoF?



Easy: 4 DoF



Easy: 5 DoF



More difficult: 1 DoF



Grübler's formula

- ▶ $n_{\text{DoF}} = m(L - 1) - \sum_{i=1}^N c_i$
- ▶ L is number of links including ground
- ▶ N is number joints
- ▶ m is DoF of rigid body (3 for planar, 6 for spatial)
- ▶ c_i number of constraints provided by joint i



Grübler's formula

- ▶ $n_{\text{DoF}} = m(L - 1) - \sum_{i=1}^N c_i = m(L - 1 - N) + \sum_{i=1}^N f_i$
- ▶ L is number of links including ground
- ▶ N is number joints
- ▶ m is DoF of rigid body (3 for planar, 6 for spatial)
- ▶ c_i number of constraints provided by joint i
- ▶ f_i number of freedoms provided by joint i



Grübler's formula

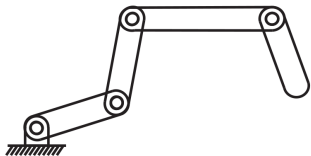
- ▶ $n_{\text{DoF}} = m(L - 1) - \sum_{i=1}^N c_i = m(L - 1 - N) + \sum_{i=1}^N f_i$
- ▶ L is number of links including ground
- ▶ N is number joints
- ▶ m is DoF of rigid body (3 for planar, 6 for spatial)
- ▶ c_i number of constraints provided by joint i
- ▶ f_i number of freedoms provided by joint i
- ▶ $f_i + c_i = m$
- ▶ Works for *generic* cases, fails under certain configurations - when joints constraints are not independent



Applications of Grübler's formula

$$n_{\text{DoF}} = m(L - 1 - N) + \sum_{i=1}^N f_i$$

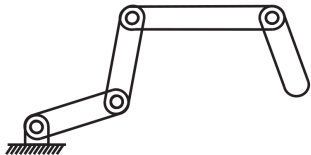
m - body DoF, L - number of links, N - number of joints, f_i - joint DoF



Applications of Grübler's formula

$$n_{\text{DoF}} = m(L - 1 - N) + \sum_{i=1}^N f_i$$

m - body DoF, L - number of links, N - number of joints, f_i - joint DoF

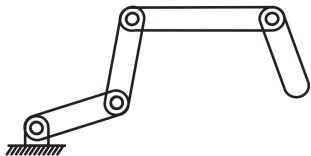


$$3(5 - 1 - 4) + (1 + 1 + 1 + 1) = 4 \text{ DoF}$$

Applications of Grübler's formula

$$n_{\text{DoF}} = m(L - 1 - N) + \sum_{i=1}^N f_i$$

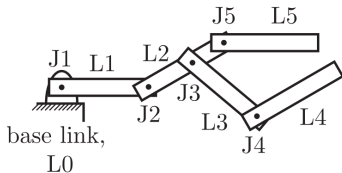
m - body DoF, L - number of links, N - number of joints, f_i - joint DoF



$$3(5 - 1 - 4) + (1 + 1 + 1 + 1) =$$

4 DoF

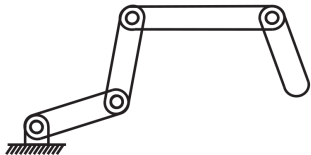
$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 0) = 4 \text{ DoF (if fixed joint)}$$



Applications of Grübler's formula

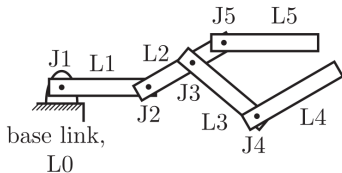
$$n_{\text{DoF}} = m(L - 1 - N) + \sum_{i=1}^N f_i$$

m - body DoF, L - number of links, N - number of joints, f_i - joint DoF

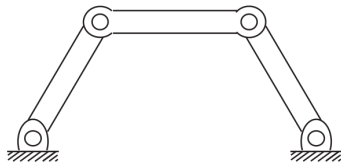


$$3(5 - 1 - 4) + (1 + 1 + 1 + 1) = 4 \text{ DoF}$$

$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 0) = 4 \text{ DoF (if fixed joint)}$$



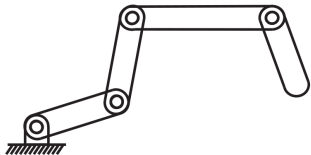
$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 1) = 5 \text{ DoF}$$



Applications of Grübler's formula

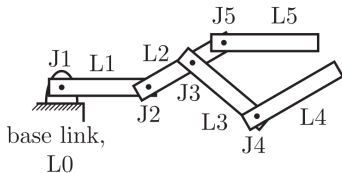
$$n_{\text{DoF}} = m(L - 1 - N) + \sum_{i=1}^N f_i$$

m - body DoF, L - number of links, N - number of joints, f_i - joint DoF

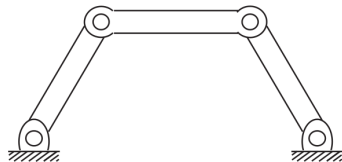


$$3(5 - 1 - 4) + (1 + 1 + 1 + 1) = 4 \text{ DoF}$$

$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 0) = 4 \text{ DoF (if fixed joint)}$$



$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 1) = 5 \text{ DoF}$$



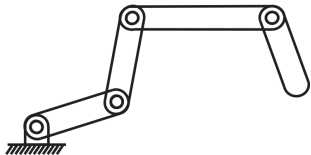
$$3(4 - 1 - 4) + (1 + 1 + 1 + 1) = 1 \text{ DoF}$$



Applications of Grübler's formula

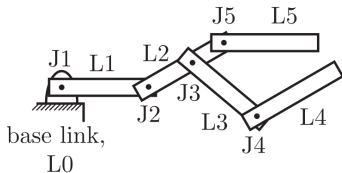
$$n_{\text{DoF}} = m(L - 1 - N) + \sum_{i=1}^N f_i$$

m - body DoF, L - number of links, N - number of joints, f_i - joint DoF

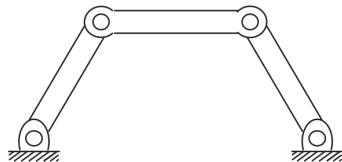


$$3(5 - 1 - 4) + (1 + 1 + 1 + 1) = 4 \text{ DoF}$$

$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 0) = 4 \text{ DoF (if fixed joint)}$$



$$3(6 - 1 - 5) + (1 + 1 + 1 + 1 + 1) = 5 \text{ DoF}$$

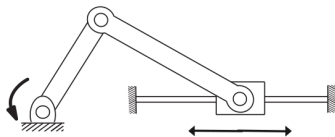


$$3(4 - 1 - 4) + (1 + 1 + 1 + 1) = 1 \text{ DoF}$$

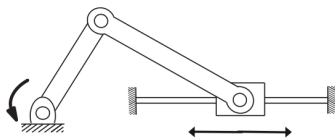
$$3(5 - 1 - 5) + (1 + 1 + 1 + 1 + 0) = 1 \text{ DoF (if two grounds connected by fixed joint)}$$



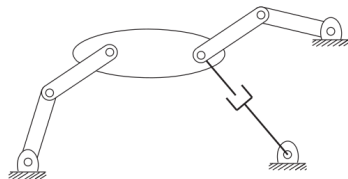
Application of Grübler's formula



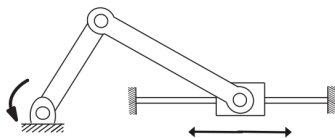
Application of Grübler's formula



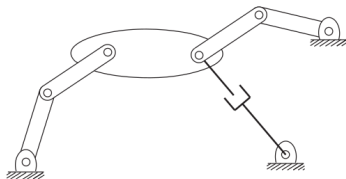
$$3(4-1-4) + (1+1+1+1) = 1 \text{ DoF}$$



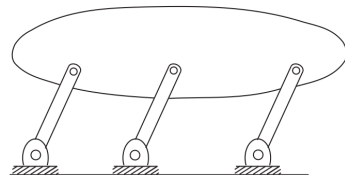
Application of Grübler's formula



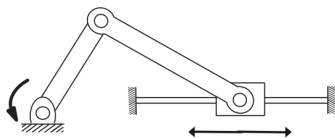
$$3(4 - 1 - 4) + (1 + 1 + 1 + 1) = 1 \text{ DoF}$$



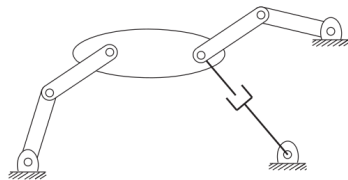
$$3(8 - 1 - 9) + (9) = 3 \text{ DoF}$$



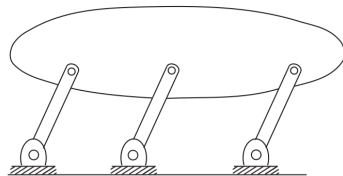
Application of Grübler's formula



$$3(4 - 1 - 4) + (1 + 1 + 1 + 1) = 1 \text{ DoF}$$



$$3(8 - 1 - 9) + (9) = 3 \text{ DoF}$$

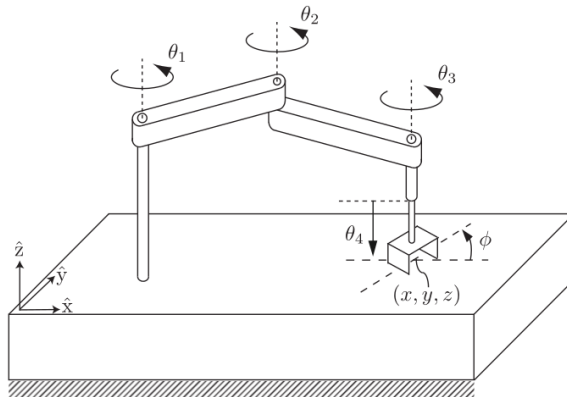


$$3(5 - 1 - 6) + (6) = 0 \text{ DoF}$$

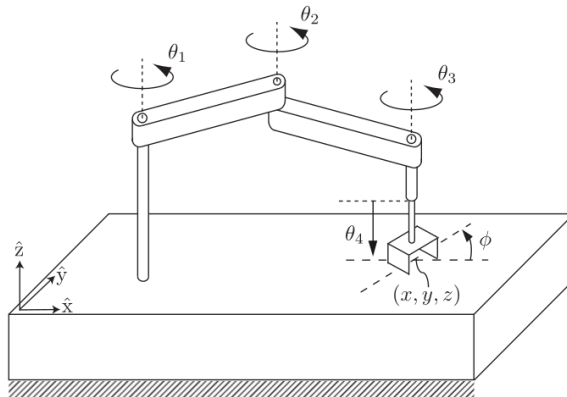
Failure: 1 DoF Grübler's formula requires independent constraints provided by the joints.



Application of Grübler's formula



Application of Grübler's formula



$$6 (5 - 1 - 4) + (1 + 1 + 1 + 1) = 4 \text{ DoF}$$



Kinematics tasks

- ▶ Forward kinematics (FK)
 - ▶ calculation of the pose of the end-effector from joint coordinates
 - ▶ $f_{\text{fk}} : \mathbf{q} \rightarrow T_{ee}$
 - ▶ $\mathbf{q} \in \mathbb{R}^N$, where N is number of joints
 - ▶ $T_{ee} \in SE(2)/SE(3)$



Kinematics tasks

- ▶ Forward kinematics (FK)

- ▶ calculation of the pose of the end-effector from joint coordinates
- ▶ $f_{fk} : \mathbf{q} \rightarrow T_{ee}$
- ▶ $\mathbf{q} \in \mathbb{R}^N$, where N is number of joints
- ▶ $T_{ee} \in SE(2)/SE(3)$

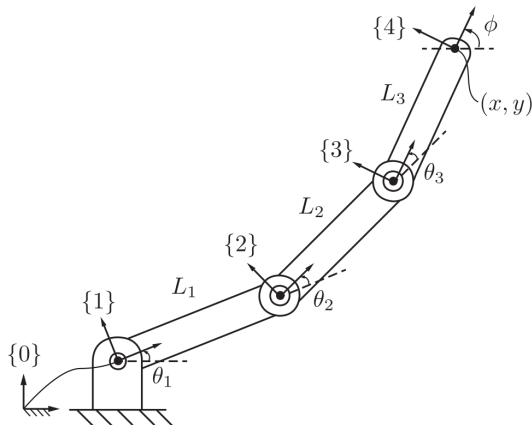
- ▶ Inverse kinematics (IK)

- ▶ calculation of joint coordinates from the given end-effector pose
- ▶ $f_{ik} : T_{ee} \rightarrow \mathbf{q}$
- ▶ $\mathbf{q} \in \mathbb{R}^N$, where N is number of joints
- ▶ $T_{ee} \in SE(2)/SE(3)$



Forward kinematics

Goal: compute FK, i.e. x, y, ϕ from $\mathbf{q} = (\theta_1 \ \theta_2 \ \theta_3)^\top$



Frame {0} origin is located in the first joint axis of rotation.



Solution

- ▶ Trigonometry solution:

$$x = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

$$\phi = \theta_1 + \theta_2 + \theta_3$$

- ▶ harder to compute for spatial manipulators



Solution

- ▶ Trigonometry solution:

$$x = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

$$\phi = \theta_1 + \theta_2 + \theta_3$$

- ▶ harder to compute for spatial manipulators
- ▶ Transformation based solution:

$$T_{04} = R(\theta_1)T_x(L_1)R(\theta_2)T_x(L_2)R(\theta_3)T_x(L_3)$$

$$R \in SE(2), T_x \in SE(2)$$

- ▶ more systematic solution



Solution

- ▶ Trigonometry solution:

$$x = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

$$\phi = \theta_1 + \theta_2 + \theta_3$$

- ▶ harder to compute for spatial manipulators
- ▶ Transformation based solution:

$$T_{04} = R(\theta_1)T_x(L_1)R(\theta_2)T_x(L_2)R(\theta_3)T_x(L_3)$$

$$R \in SE(2), T_x \in SE(2)$$

- ▶ more systematic solution
- ▶ how to get x, y, ϕ from $T = T_{04}$?



Solution

- ▶ Trigonometry solution:

$$x = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

$$\phi = \theta_1 + \theta_2 + \theta_3$$

- ▶ harder to compute for spatial manipulators
- ▶ Transformation based solution:

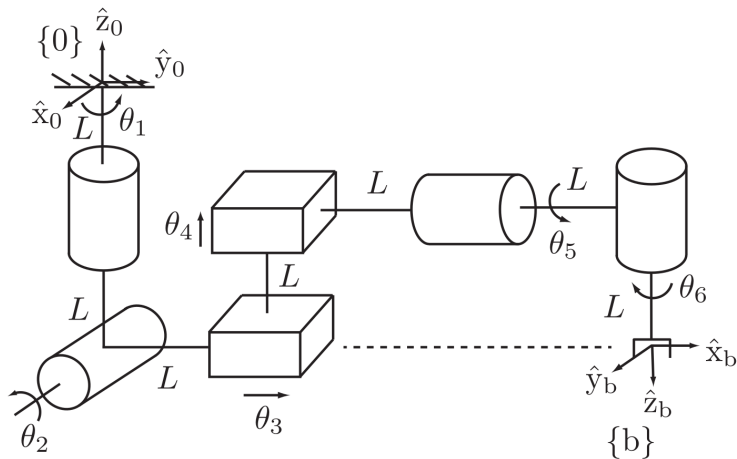
$$T_{04} = R(\theta_1)T_x(L_1)R(\theta_2)T_x(L_2)R(\theta_3)T_x(L_3)$$

$$R \in SE(2), T_x \in SE(2)$$

- ▶ more systematic solution
- ▶ how to get x, y, ϕ from $T = T_{04}$?
- ▶ $x = T_{13}, \quad y = T_{23}, \quad \phi = \text{atan2}(T_{21}, T_{22})$



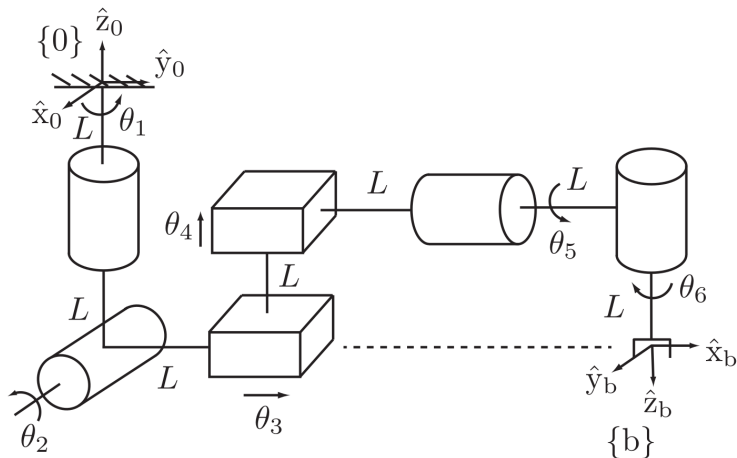
Forward kinematics for spatial robot



$$T_{0b} = ?$$



Forward kinematics for spatial robot

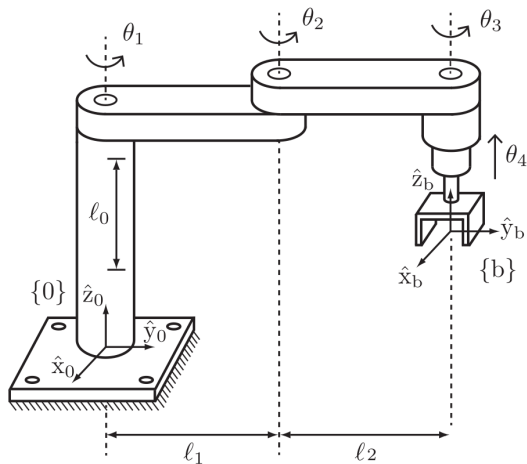


$T_{0b} =$

$$R_z(\theta_1)T_z(-L)R_x(\theta_2)T_y(L)T_y(\theta_3)T_z(L + \theta_4)T_y(L)R_y(\theta_5)R_z(-\theta_6)T_z(L)R_z(\pi/2)R_x(\pi)$$



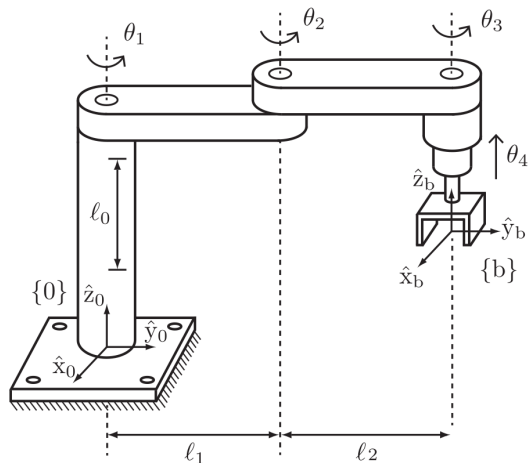
Forward kinematics for spatial robot



$$T_{0b} = ?$$



Forward kinematics for spatial robot

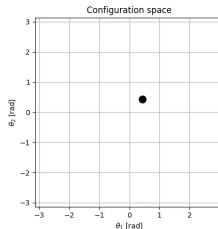
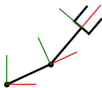


$$T_{0b} = T_z(\ell_0)R_z(\theta_1)T_y(\ell_1)R_z(\theta_2)T_y(\ell_2)R_z(\theta_3)T_z(-\theta_4)$$



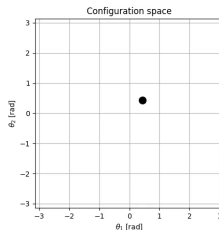
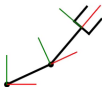
Configuration space and Task space

- ▶ Configuration space for 2 DoF robot
 - ▶ every point corresponds to a configuration



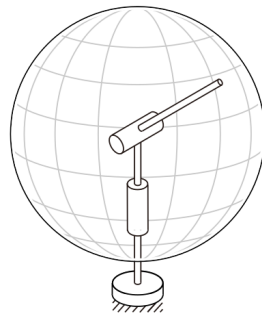
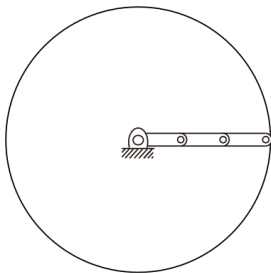
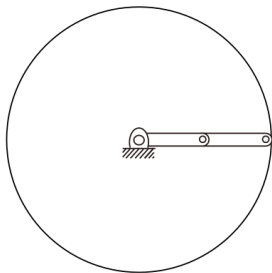
Configuration space and Task space

- ▶ Configuration space for 2 DoF robot
 - ▶ every point corresponds to a configuration
- ▶ Task space
 - ▶ a space in which the robot's task can be naturally expressed (robot independent)
 - ▶ a point in a task-space can be reached by multiple configurations
 - ▶ e.g. manipulating spatial object, task space is $SE(3)$
 - ▶ e.g. drawing on a paper, task space is $\mathbb{R}^2$



Workspace of the robot

- ▶ Specification of the configurations that the end-effector of the robot can reach
- ▶ Depends on the robot structure
- ▶ End-effector orientation is often ignored (but it depends on the task)



- ▶ Universal Robot Description Format
- ▶ XML file that describes robots' kinematics, geometry, dynamics
- ▶ Used in Robotic Operating System (ROS)
- ▶ Limited to open kinematic chains (including tree structures)
- ▶ Robot is described by:
 - ▶ Links (rigid body)
 - ▶ Joints (connects two links together)



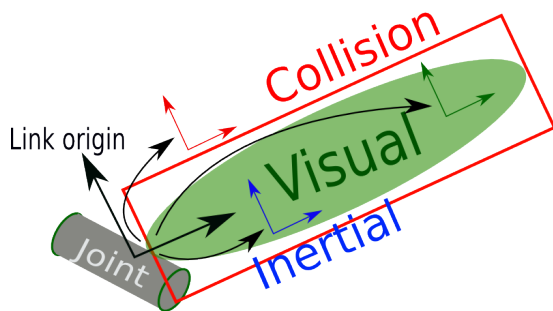
URDF - Links

- Why we need visual and collision?

```
<robot name="robot">
  <link name="link">
    <inertial>
      ...
    </inertial>

    <visual>
      ...
    </visual>

    <collision>
      ...
    </collision>
  </link>
</robot>
```



URDF - Links

- Why we need visual and collision?

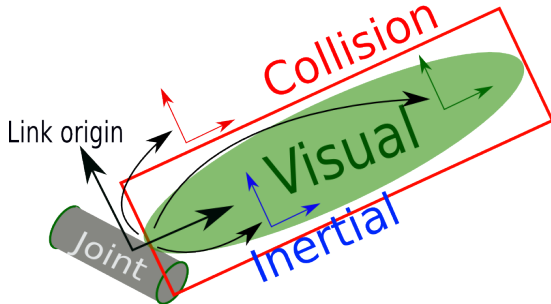
- RPY: Roll Pitch Yaw

$$R = R_z(\text{yaw})R_y(\text{pitch})R_x(\text{roll})$$

```
<robot name="robot">
  <link name="link">
    <inertial>
      <origin xyz="0 0 0.5" rpy="0 0 0"/>
      <mass value="1"/>
      <inertia ixx="100" ixy="0" ixz="0" iyy="100" iyz="0" izz="100"/>
    </inertial>

    <visual>
      ...
    </visual>

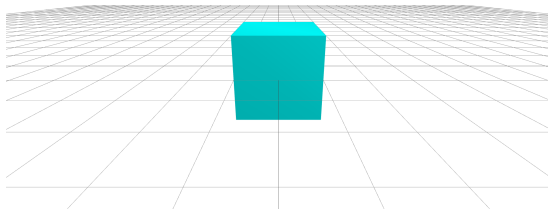
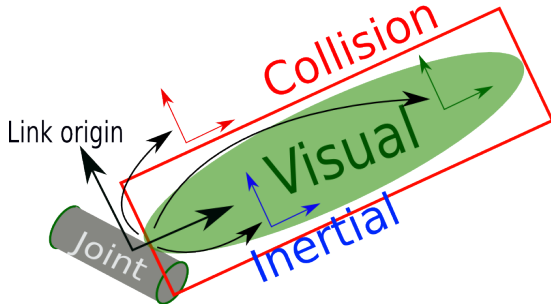
    <collision>
      ...
    </collision>
  </link>
</robot>
```



URDF - Links

- ▶ Why we need visual and collision?
- ▶ RPY: Roll Pitch Yaw
 $R = R_z(\text{yaw})R_y(\text{pitch})R_x(\text{roll})$

```
<robot name="robot">  
  <link name="link">  
    <inertial>  
      ...  
    </inertial>  
  
    <visual>  
      <origin xyz="0 0 0" rpy="0 0 0"/>  
      <geometry>  
        <box size="1 1 1"/>  
      </geometry>  
      <material name="Cyan">  
        <color rgba="0 1.0 1.0 1.0"/>  
      </material>  
    </visual>  
  
    <collision>  
      ...  
    </collision>  
  </link>  
</robot>
```



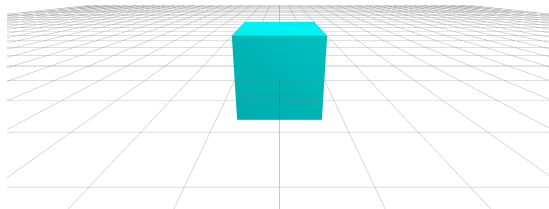
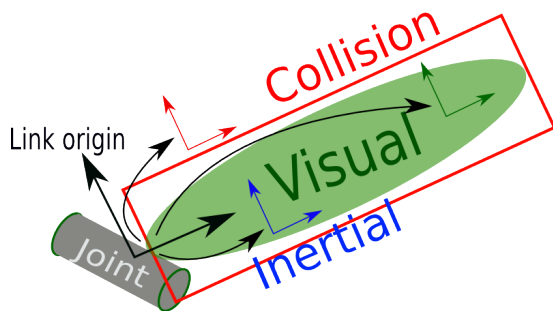
URDF - Links

- ▶ Why we need visual and collision?
- ▶ RPY: Roll Pitch Yaw
 $R = R_z(\text{yaw})R_y(\text{pitch})R_x(\text{roll})$

```
<robot name="robot">
  <link name="link">
    <inertial>
      ...
    </inertial>

    <visual>
      ...
    </visual>

    <collision>
      <origin xyz="0 0 0" rpy="0 0 0"/>
      <geometry>
        <cylinder radius="1" length="0.5"/>
      </geometry>
    </collision>
  </link>
</robot>
```



URDF - Links

- Why we need visual and collision?

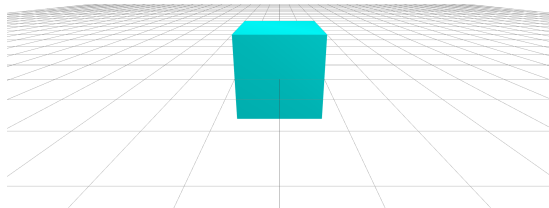
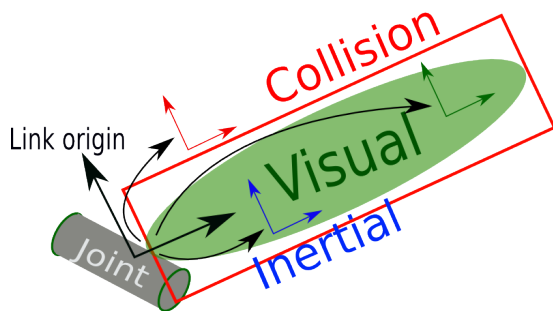
- RPY: Roll Pitch Yaw

$$R = R_z(\text{yaw})R_y(\text{pitch})R_x(\text{roll})$$

```
<robot name="robot">
  <link name="link">
    <inertial>
      ...
    </inertial>

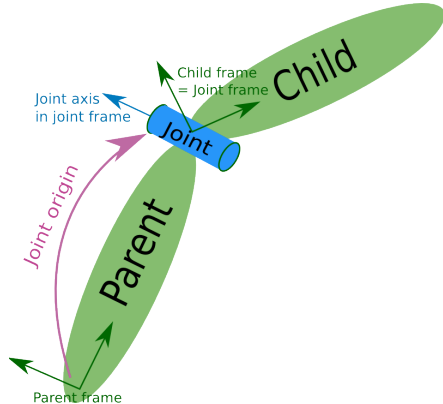
    <visual>
      ...
    </visual>

    <collision>
      ...
    </collision>
  </link>
</robot>
```



URDF - Joints

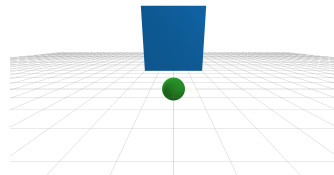
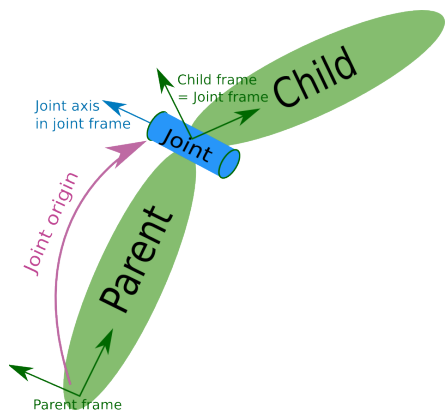
```
<link name="l0"></link> <!-- sphere -->  
<link name="l1"></link> <!-- box -->
```



URDF - Joints

```
<link name="l0"></link> <!-- sphere -->  
<link name="l1"></link> <!-- box -->
```

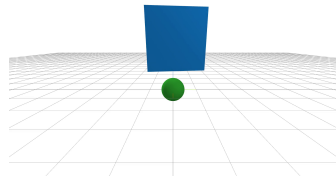
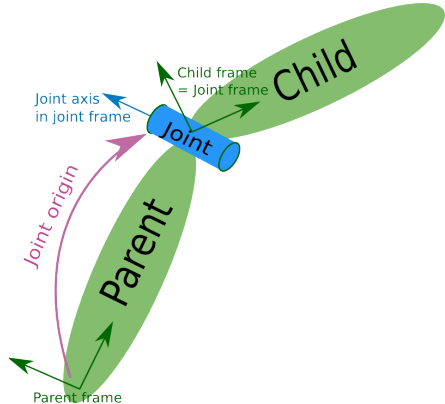
```
<joint name="j0" type="fixed">  
  <origin xyz="0 0 1" rpy="0 0 0"/>  
  <parent link="l0"/>  
  <child link="l1"/>  
</joint>
```



URDF - Joints

```
<link name="l0"></link> <!-- sphere -->
<link name="l1"></link> <!-- box -->

<joint name="j0" type="revolute">
  <origin xyz="0 0 1" rpy="0 0 0"/>
  <parent link="l0"/>
  <child link="l1"/>
  <axis xyz="0 0 1"/>
  <limit effort="30" velocity="1.0" lower="-3.14" upper="3.14" />
</joint>
```

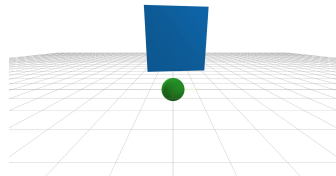
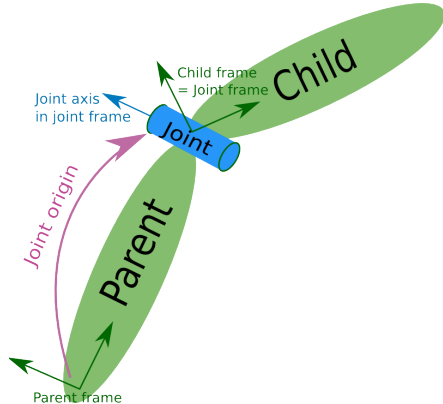


URDF - Joints

```
<link name="l0"></link> <!-- sphere -->
<link name="l1"></link> <!-- box -->

<joint name="j0" type="revolute">
  <origin xyz="0 0 1" rpy="0 0 0"/>
  <parent link="l0"/>
  <child link="l1"/>
  <axis xyz="0 0 1"/>
  <limit effort="30" velocity="1.0" lower="-3.14" upper="3.14" />
</joint>
```

- Can the upper limit be smaller than π ?

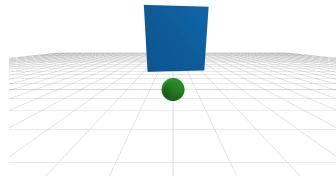
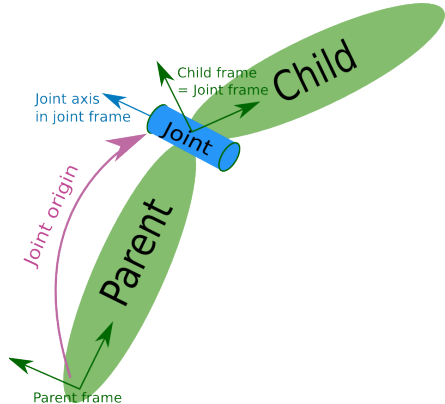


URDF - Joints

```
<link name="l0"></link> <!-- sphere -->
<link name="l1"></link> <!-- box -->

<joint name="j0" type="revolute">
  <origin xyz="0 0 1" rpy="0 0 0"/>
  <parent link="l0"/>
  <child link="l1"/>
  <axis xyz="0 0 1"/>
  <limit effort="30" velocity="1.0" lower="-3.14" upper="3.14" />
</joint>
```

- ▶ Can the upper limit be smaller than π ?
- ▶ Can the upper limit be larger than π ?

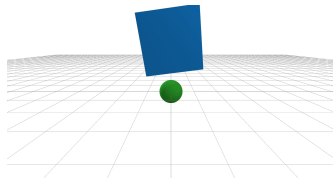
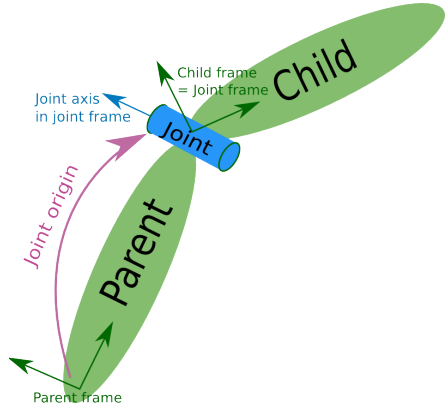


URDF - Joints

```
<link name="l0"></link> <!-- sphere -->
<link name="l1"></link> <!-- box -->

<joint name="j0" type="revolute">
  <origin xyz="0 0 1" rpy="0 0 0"/>
  <parent link="l0"/>
  <child link="l1"/>
  <axis xyz="1 0 0"/>
  <limit effort="30" velocity="1.0" lower="-3.14" upper="3.14" />
</joint>
```

- ▶ Can the upper limit be smaller than π ?
- ▶ Can the upper limit be larger than π ?

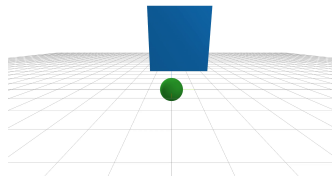
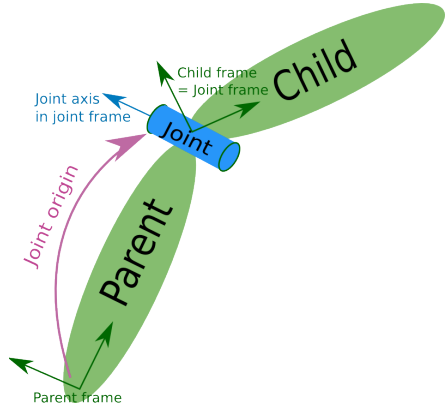


URDF - Joints

```
<link name="l0"></link> <!-- sphere -->
<link name="l1"></link> <!-- box -->

<joint name="joint0" type="prismatic">
  <origin xyz="0 0 1" rpy="0 0 0"/>
  <parent link="l0"/>
  <child link="l1"/>
  <axis xyz="0 1 0"/>
  <limit effort="30" velocity="1.0" lower="-1.0" upper="1.0" />
</joint>
```

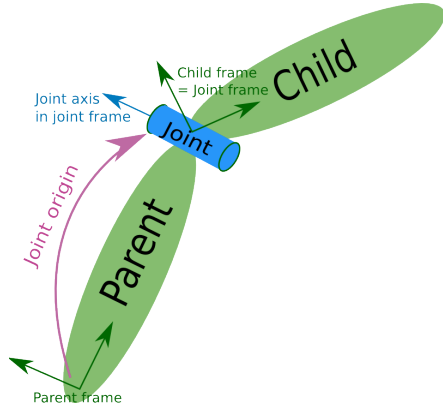
- ▶ Can the upper limit be smaller than π ?
- ▶ Can the upper limit be larger than π ?



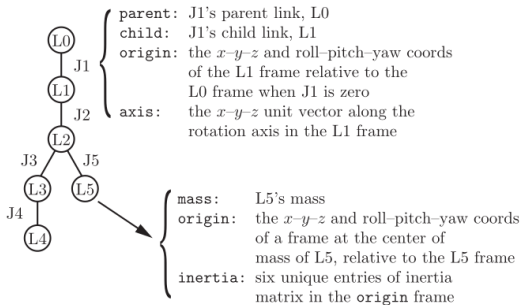
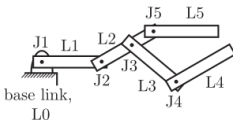
URDF - Joints

```
<link name="l0"></link> <!-- sphere -->  
<link name="l1"></link> <!-- box -->
```

- ▶ Can the upper limit be smaller than π ?
- ▶ Can the upper limit be larger than π ?
- ▶ Other joint types: continuous, planar, floating



URDF example



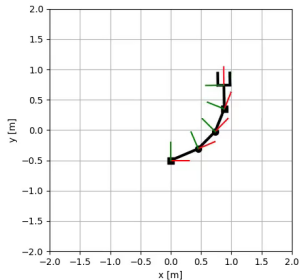
Summary

- ▶ Robotic manipulator (joints, links, end-effector)
- ▶ Joint types (DoF, constraints)
- ▶ Open/Closed kinematic chain
- ▶ Grübler's formula
- ▶ Forward/Inverse kinematics
- ▶ Configuration space / Task space / Workspace
- ▶ URDF



Laboratory

- Implement FK for planar manipulator



Laboratory

- ▶ Implement FK for planar manipulator
- ▶ Create your own URDF model

