Intelligent Robot Control

Lecture 1: Introduction and overview

Tadej Petrič

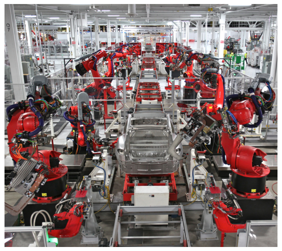
tadej.petric@ijs.si

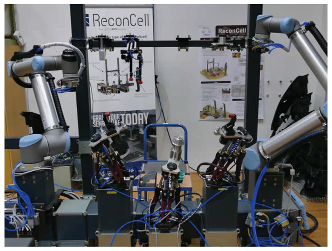
Institut
"Jožef Stefan"
Ljubljana, Slovenija

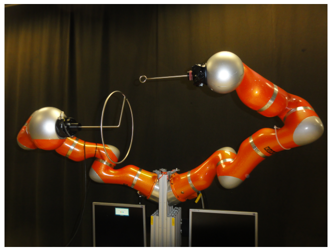
General information

- Prerequisites
  - ⋆ Knowledge on robot kinematics and dynamics is necessary!
  - ⋆ Knowledge on automatic control is useful.
- Aims
  - ⋆ Knowledge of robot kinematics, dynamics, simulation, and control.
- Textbooks
  - ⋆ Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G., **Robotics: Modelling, Planning and Control**, Springer-Verlag, London, UK, 2009.
  - ⋆ Siciliano, B.; Khatib, O.(Eds.), **Springer Handbook of Robotics**, Springer-Verlag Berlin Heidelberg 2008.
- Materials
  - ⋆ `\\www.ijs.si/~leon/mps/`
- Contact
  - ⋆ ✉ `leon.zlajpah@ijs.si`

Robots & Robotics







"A **robot** is a machine — especially one programmable by a computer — capable of carrying out a complex series of actions automatically. Robots can be guided by an external control device or the control may be embedded within. Robots may be constructed to take on human form but most robots are machines designed to perform a task with no regard to how they look."

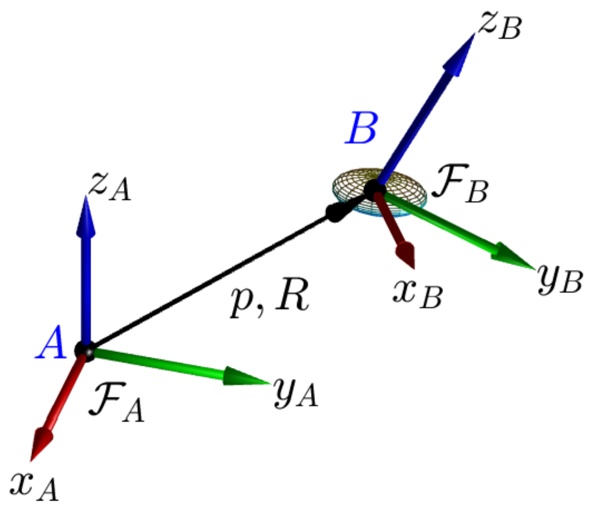
"**Robotics** is an interdisciplinary branch of engineering and science that includes mechanical engineering, electrical engineering, computer science, and others. Robotics deals with the design, construction, operation, and use of robots, as well as computer systems for their control, sensory feedback, and information processing."

$\boxed{\textbf{Motion} \Leftrightarrow \textbf{Forces} \Leftrightarrow \textbf{Control}}$

Program

- Introduction
  - ⋆ kinematics and dynamics of robot mechanisms
  - ⋆ joint space and task space
  - ⋆ motion planning
  - ⋆ robot control
- Modeling and simulation of robot mechanisms
  - ⋆ simulation in MATLAB/Simulink, HAPTIX, ...
  - ⋆ real-time simulation
- Robot control systems
  - ⋆ dynamic manipulation using force, vision or tactile sensors
  - ⋆ compliance robot control
  - ⋆ optimal robot control
- Redundant robot systems
  - ⋆ task decomposition
  - ⋆ redundancy resolution
  - ⋆ obstacle avoidance
- Robot cooperation
  - ⋆ Kinematics and dynamics of dual-arm robots
  - ⋆ Control of dual-arm robots

Position and orientation of rigid bodies



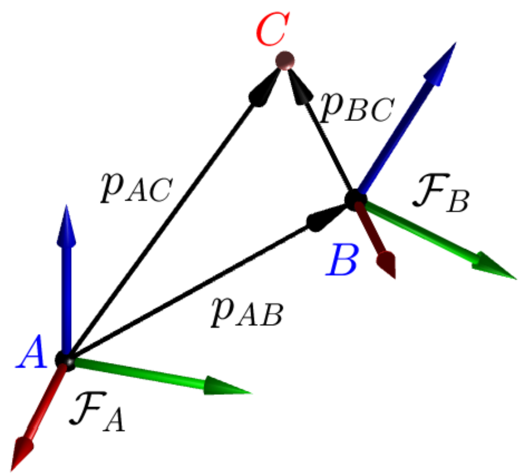
Frame $\mathcal{F}_B$ is attached to the body.

Position of the body $\boldsymbol{p} \in \mathbb{R}^3$ expressed in base frame $\mathcal{F}_A$:

$$\boldsymbol{p} = {}^A\boldsymbol{p}_{AB} = [{}^Ax_{AB}\ {}^Ay_{AB}\ {}^Az_{AB}]^T$$

Orientation of the body $\mathbf{R} \in \mathrm{SO}(3) \subset \mathbb{R}^{3\times 3}$ expressed in base frame $\mathcal{F}_A$:

$$\mathbf{R} = {}^A\mathbf{R}_B = [{}^Ax_B\ {}^Ay_B\ {}^Az_B]^T = [\boldsymbol{n}\ \boldsymbol{s}\ \boldsymbol{a}]$$



Representation of point C (fixed in $\mathcal{F}_B$)in different frames:

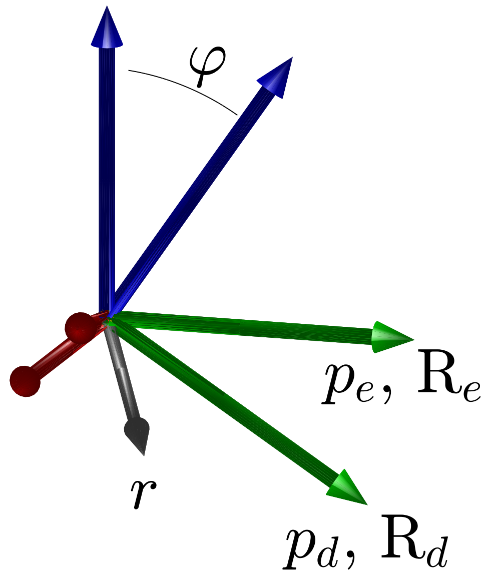
$$\boldsymbol{p}_{AC} = \boldsymbol{p}_{AB} + \boldsymbol{p}_{BC} = \boldsymbol{p}_{AB} + \mathbf{R}_B\ {}^B\boldsymbol{p}_{BC}$$

$${}^B\boldsymbol{p}_{BC} = -\mathbf{R}_B^T\,\boldsymbol{p}_{AB} + \mathbf{R}_B^T\,\boldsymbol{p}_{AC}$$

Orientation representation

The orientation representation with rotation matrix $\mathbf{R}$ is redundant due to the orthonormality constraints. Using a set of Euler angles $\Phi = [\alpha, \beta, \gamma]^T$ we can obtain a minimal representation of the orientation as

$$\mathbf{R}(\Phi) = \mathbf{R}_a(\alpha)\mathbf{R}_b(\beta)\mathbf{R}_c(\gamma)$$



The main drawback of Euler angles representation are **representation singularities**.

Alternatives are angle/axis representation of the rotation matrix

$$\mathbf{R} = \mathbf{R}(\varphi, \boldsymbol{r})$$

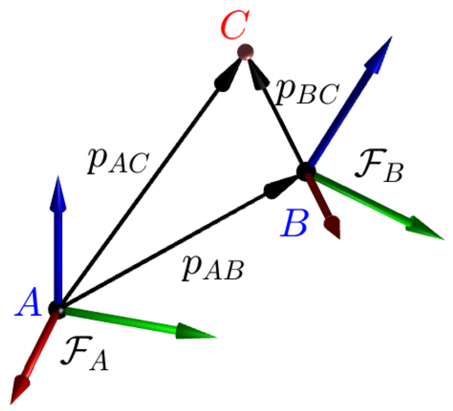
where φ and $\boldsymbol{r}$ are the angle and the vector of rotation,

or Euler parameters (unit quaternions)

$$\mathcal{Q} = \{\eta, \boldsymbol{\epsilon}\} = \{\cos(\varphi/2)\ ,\ \sin(\varphi/2)\boldsymbol{r}\}$$

Homogenous transformation

Homogenous transformation describes the relation between two reference frames, i.e. the relative pose consisting of position and orientation.



$$\boldsymbol{p}_{AC} = \boldsymbol{p}_{AB} + \boldsymbol{p}_{BC} = \boldsymbol{p}_{AB} + \mathbf{R}_B\, {}^B\boldsymbol{p}_{BC}$$

$${}^A\widehat{\boldsymbol{p}}_{AC} = \left[\begin{array}{c} {}^A\boldsymbol{p}_{BC} \\ \hline 1 \end{array}\right] = \left[\begin{array}{c|c} {}^A\mathbf{R}_B & \boldsymbol{p}_{AB} \\ \hline 0\ 0\ 0 & 1 \end{array}\right] \left[\begin{array}{c} {}^B\boldsymbol{p}_{BC} \\ \hline 1 \end{array}\right] = {}^AT_B\, {}^B\widehat{\boldsymbol{p}}_{BC}$$

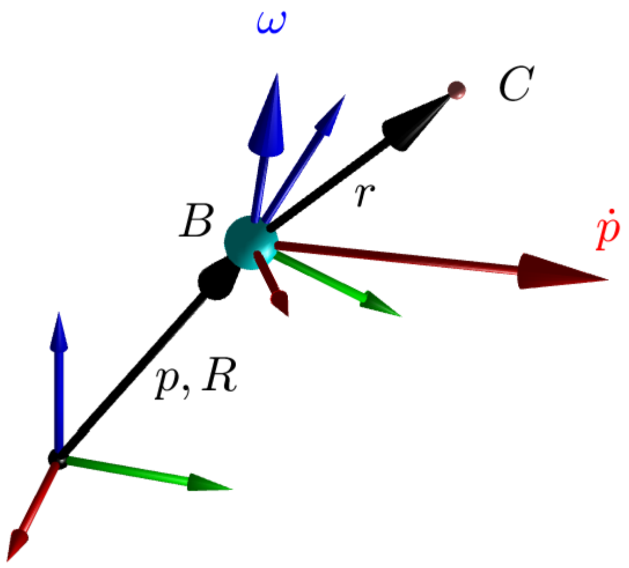
Some properties:

- **Inverse**: $({}^AT_B)^{-1} = \left[\begin{array}{c|c} {}^A\mathbf{R}_B^T & -{}^A\mathbf{R}_B^T\, {}^A\boldsymbol{p}_B \\ \hline 0\ 0\ 0 & 1 \end{array}\right] = \left[\begin{array}{c|c} {}^B\mathbf{R}_A & {}^B\boldsymbol{p}_A \\ \hline 0\ 0\ 0 & 1 \end{array}\right] = {}^BT_A$
- **Compound transformations**: ${}^AT_C = {}^AT_B\, {}^BT_C$ (not commutative!)

Linear and angular velocities of rigid bodies

Linear velocity $\dot{\boldsymbol{p}} \in \mathbb{R}^3$ of a rigid body is a vector equal to the time rate of change of its linear position.

Angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ is a vector quantity that describes the angular speed at which the orientation of the rigid body is changing and the instantaneous axis about which the body is rotating.



Body B:

$$\dot{\boldsymbol{p}} = \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} \qquad \mathbf{S}(\boldsymbol{\omega}) = \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}t}\mathbf{R}^T$$

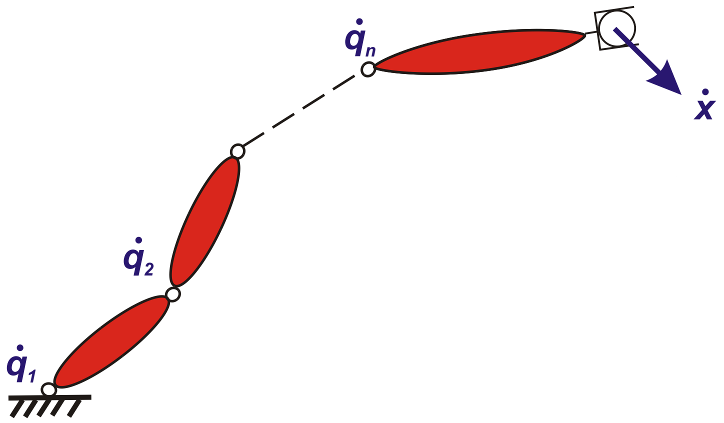
Fixed point C on body B:

$$\dot{\boldsymbol{p}}_C = \dot{\boldsymbol{p}} + \boldsymbol{\omega} \times \boldsymbol{r} = \dot{\boldsymbol{p}} + \mathbf{S}(\mathbf{r})\boldsymbol{\omega} \qquad \boldsymbol{\omega}_C = \boldsymbol{\omega}$$

$$\mathbf{S}(\boldsymbol{a}) = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} ; \ \boldsymbol{p} = \int \dot{\boldsymbol{p}}\,\mathrm{d}t \ ; \ R \neq \int \boldsymbol{\omega}\,\mathrm{d}t \ ; \ {}^{A}\boldsymbol{\omega} = {}^{A}\dot{\boldsymbol{p}}_B + {}^{B}\dot{\boldsymbol{p}}$$

Kinematics

Robot is seen as **(open) kinematic chain of rigid bodies interconnected by (revolute or prismatic) joints**.



Parameterization:

Unambiguous and minimal characterization of the robot configuration

n = degrees of freedom (DOF)
n = robot joints (rotational or translational)

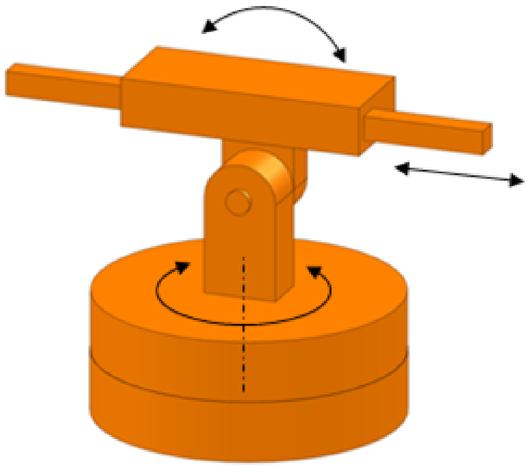
Configuration on n-DOF robot is described by joint coordinates

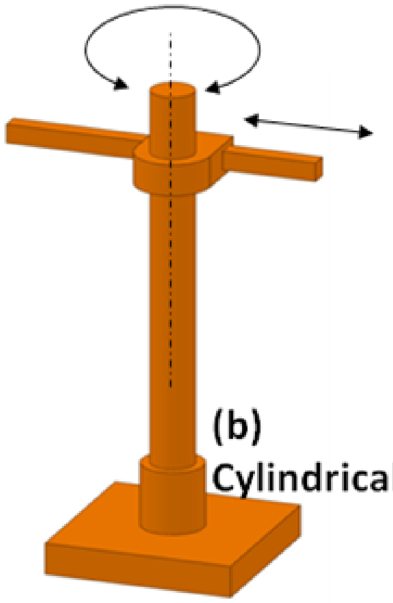
$$\boldsymbol{q} = [q_1, q_2, \ldots, q_n]^T \qquad q_i \in Q_i = [q_{i,min}, q_{i,max}]$$

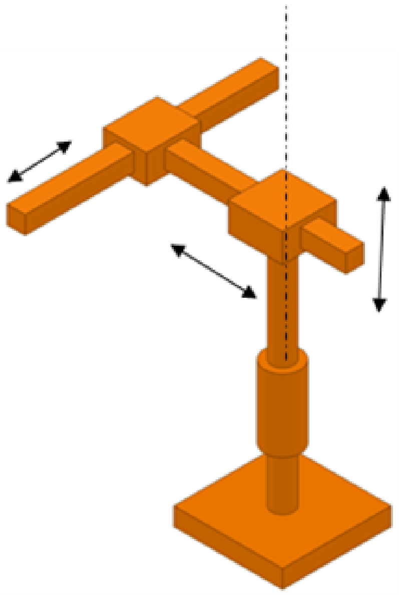
The configuration space $\mathcal{C}$ is the space where the joint variables $\boldsymbol{q}$ are defined

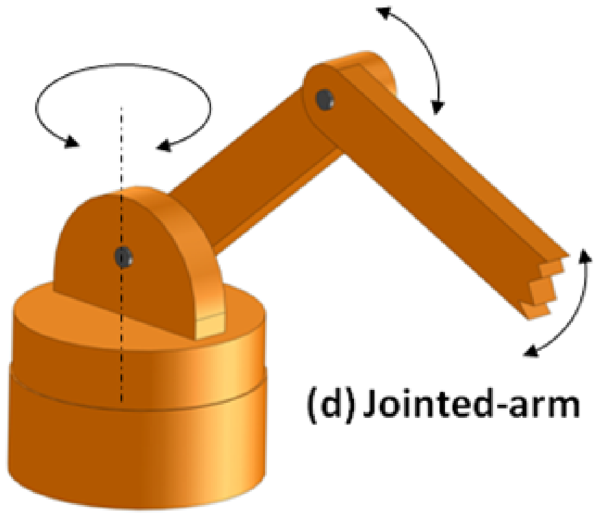
$$\mathcal{C} : Q_1 \times Q_2 \times \cdots \times Q_n$$

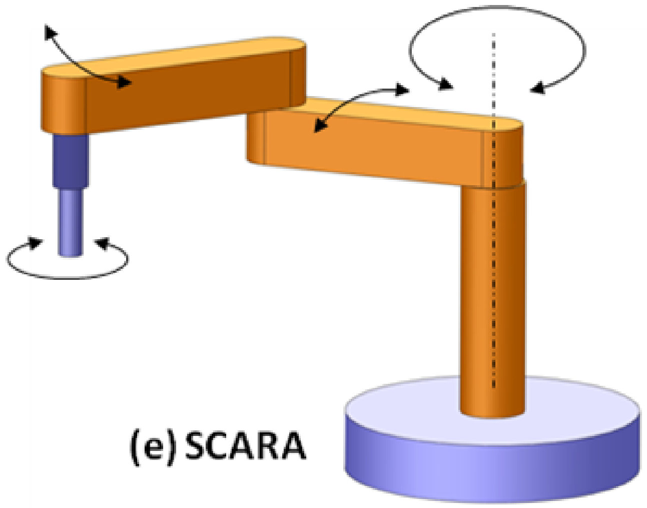
Robot kinematic types


(a) Polar


(b) Cylindrical

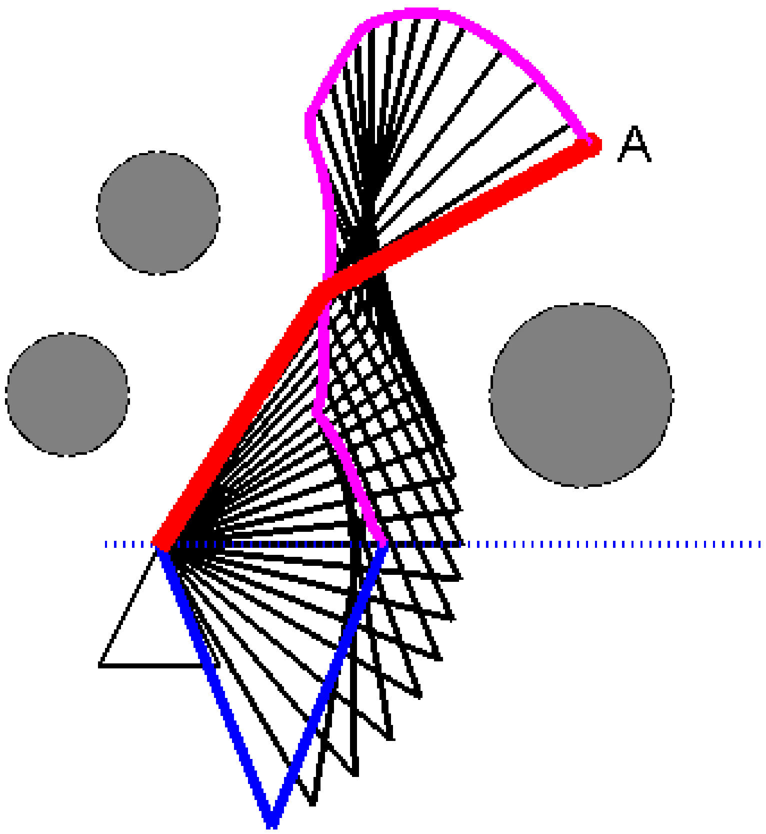

(c) Cartesian

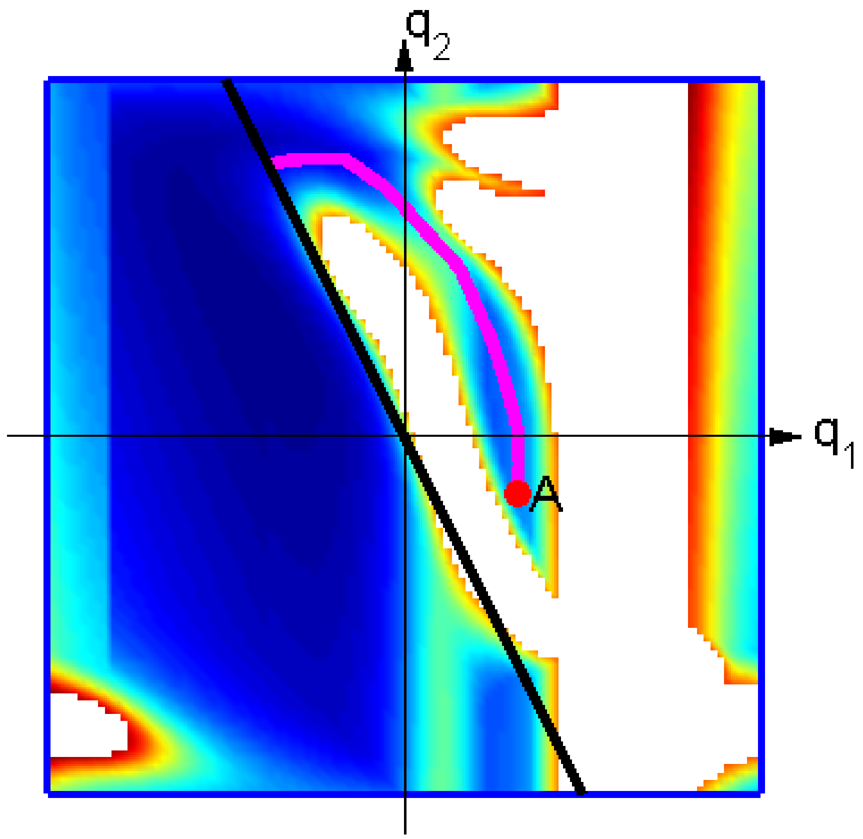

(d) Jointed-arm


(e) SCARA

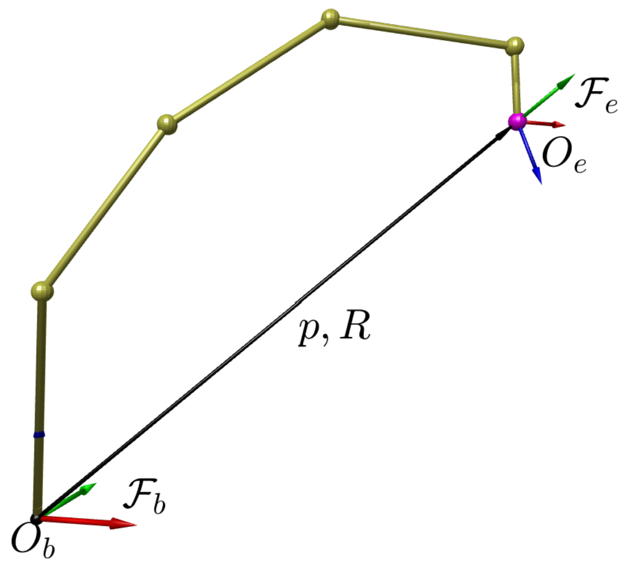
2R Manipulator

$$\mathcal{C} : [-\pi, \pi] \times [-\pi, \pi]$$





Robot pose



Frame $\mathcal{F}_b$ attached to the end-effector.

End-effector position O_e (operational point) in base coordinate frame $\mathcal{F}_b$:

$$\mathbf{T}_e = \begin{bmatrix} \mathbf{R}(\boldsymbol{q}) & \boldsymbol{p}(\boldsymbol{q}) \\ \mathbf{0} & 1 \end{bmatrix}$$

The operational space $\mathcal{O}$ is space, where the positions/orientations of the robot end-effector are defined (6-dimensional Cartesian space).

Reachable workspace: is the set of all positions $\boldsymbol{p}$ which the robot can reach with **at least one** orientation

Dextrous workspace: is the set of all positions $\boldsymbol{p}$ which the robot can reach with **any feasible** orientation

Direct kinematics

We relate the configuration to the pose, i.e. **position** and **orientation** of the end-effector.

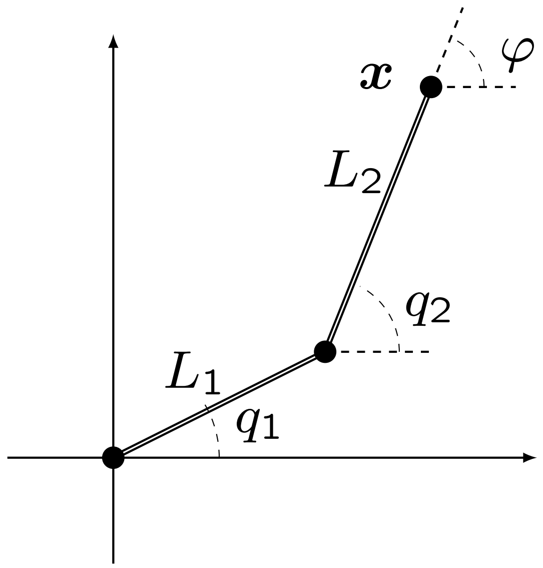
$$\boxed{\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{q})}$$

The direct kinematic function $\boldsymbol{f}$ depends on what $\boldsymbol{x}$ represents.

Analytic solution of $\boldsymbol{f}$ can be obtained:

- by geometric inspection
- using homogenous transformations: $\mathbf{T}_e = \mathbf{T}_1\,{}^1\mathbf{T}_2 \cdots {}^{n-1}\mathbf{T}_n\,{}^n\mathbf{T}_E$

Planar 2R robot:



$$\boldsymbol{x} = \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} = \begin{bmatrix} L_1\cos(q_1) + L_2\cos(q_1+q_2) \\ L_1\sin(q_1) + L_2\sin(q_1+q_2) \\ q_1+q_2 \end{bmatrix}$$

$$\mathbf{T}_e = \sum_{i=1}^{2} \begin{bmatrix} \cos(q_i) & -\sin(q_i) & 0 & L_i\cos(q_i) \\ \sin(q_i) & \cos(q_i) & 0 & L_i\sin(q_i) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Task space

Task space $\mathcal{T} \subseteq \mathcal{O}$ is the space where the operation of robot is required.

DOFs needed for some common tasks:

$m = 2$

- pointing in space (two angles)
- positioning in plane (two positions)

$m = 3$

- orientation in space (three angles)
- positioning and orientation in plane (two positions and one angle)

$m = 5$

- positioning and pointing in space (three positions and two angles)

$m = 6$

- positioning and orientation in space (three positions and three angles)

Direct kinematics map

Direct (forward) kinematics represents the mapping form $\mathcal{C}$ to $\mathcal{T}$.

Why this mapping is important:

- defines the workspace
- trajectory planning
- motion control

Different compact descriptions of positional and/or orientation (pose) components:

$\boldsymbol{x} = \{\boldsymbol{p}, \mathbf{R}\}$	Position vector, Rotation matrix
$\boldsymbol{x} = \{\boldsymbol{p}, Q\}$	Position vector, Quaternion
$\boldsymbol{x} = [\boldsymbol{p}^T, \boldsymbol{\phi}^T]^T$	Position vector, Euler angles

where

$$\boldsymbol{p} = [x, y, z]^T \qquad \boldsymbol{\phi} = [\alpha, \beta, \gamma]^T \qquad \mathcal{Q} = \{\eta, \boldsymbol{\epsilon}\}$$

Description selection depends on the task.

Inverse kinematics

The inverse kinematics problem is: given the position of the end-effector $\boldsymbol{x}$ in $\mathcal{T}$ find the joint coordinates $\boldsymbol{q}$

$$\boxed{\boldsymbol{q} = \boldsymbol{f}^{-1}(\boldsymbol{x})}$$

Problems:

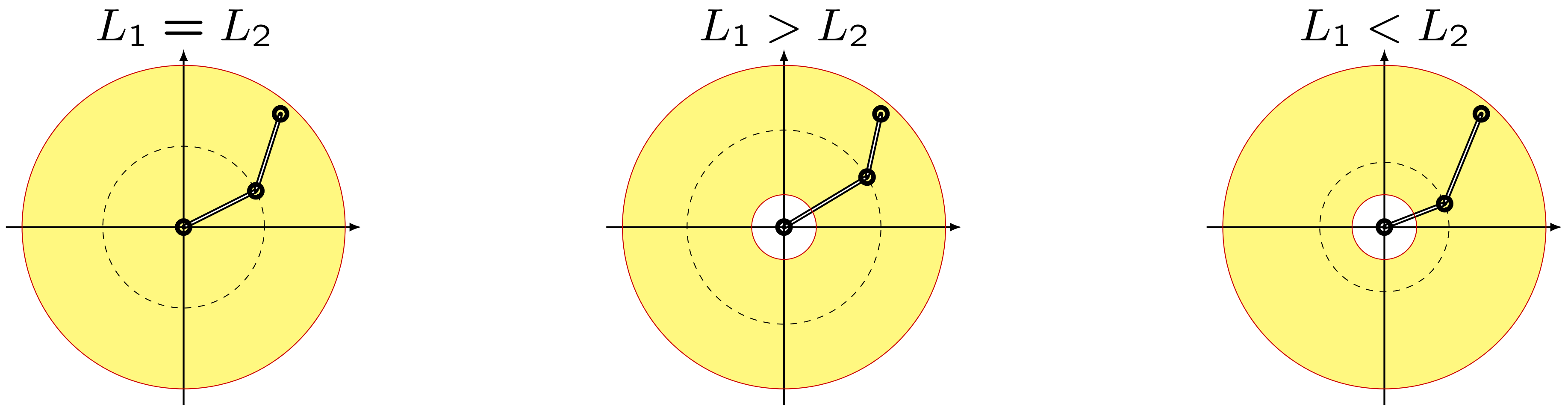
Existence: For a solution $\boldsymbol{x}$ must be in $\mathcal{O}$.

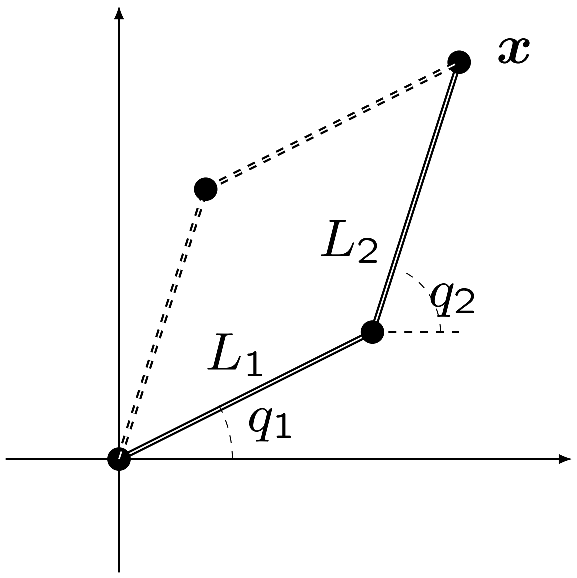
Uniqueness: Multiple solutions exist.

Methods: Analytical (not always possible) or numerical (Newton, gradient).

Planar 2R robot

Task space is the position $\boldsymbol{x}$ of the end-effector and $\boldsymbol{x} \in \mathcal{O}$





$$\varphi = \text{atan2}(y, x);$$

$$\boldsymbol{q} = \begin{bmatrix} \varphi \pm \dfrac{\text{acos}(x^2 + y^2 + L_1^2 - L_2^2)}{2L_1\sqrt{x^2+y^2}} \\ \mp \dfrac{\text{acos}(x^2 + y^2 - L_1^2 - L_2^2)}{2L_1L_2} \end{bmatrix}$$

Inverse kinematics - numerical solution

When an analytical (close-form) solution $\boldsymbol{q}$ for $\boldsymbol{x}(\boldsymbol{f}(\boldsymbol{q})$ does not exists or it is not easy to be found we can solve it using numerical methods.

Newton method:

$$q(k+1) = \boldsymbol{q}(k) + \frac{\partial \boldsymbol{f}(\boldsymbol{q}(k))}{\partial \boldsymbol{q}} (\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{q}(k)))$$

Gradient method:

$$\boldsymbol{q}(k+1) = \boldsymbol{q}(k) - k\nabla_q H(\boldsymbol{q}(k)) \qquad H(\boldsymbol{q}) = \frac{1}{2}\|x - \boldsymbol{f}(q)\|^2$$

Both methods are actually feedback schemes and are similar to kinematic controllers.

Problems:

- singularities
- multiple solutions (redundancy)
- convergence

Differential kinematics

Differential kinematics is the relation between the velocities in the joint space and velocities in the task space.

A: Map joint velocities to end-effector **instantaneous linear and angular velocities**:

$$\boxed{\boldsymbol{v} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_p \\ \mathbf{J}_o \end{bmatrix} \dot{\boldsymbol{q}} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}}}$$

$$\mathbf{S}(\boldsymbol{\omega}) = \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}t}\mathbf{R}$$

$\mathbf{J}(\boldsymbol{q})$ is geometric (or basic) Jacobian matrix - always $(6 \times n)$ matrix.

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_p \\ \mathbf{J}_o \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{p1} & \cdots & \mathbf{J}_{pn} \\ \mathbf{J}_{o1} & \cdots & \mathbf{J}_{on} \end{bmatrix}$$

Jacobian computation

To get the geometric Jacobian $\mathbf{J}$ contributions of joints velocities for the linear and angular velocity of the end-effector are calculated separately

$$\dot{\boldsymbol{p}} = \sum_{i=1}^{n} \frac{\partial \boldsymbol{p}}{\partial q_i} \dot{q}_i = \sum_{i=1}^{n} \mathbf{J}_{pi} \dot{q}_i$$

$$\boldsymbol{\omega} = \sum_{i=1}^{n} \boldsymbol{\omega}_{i-1,i} = \sum_{i=1}^{n} \mathbf{J}_{oi} \dot{q}_i$$

where

$$\begin{bmatrix} \mathbf{J}_{pi} \\ \mathbf{J}_{oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{z}_i \\ \mathbf{0} \end{bmatrix} & \text{for a \textbf{prismatic} joint} \\ \begin{bmatrix} \boldsymbol{z}_i \times (\boldsymbol{p} - \boldsymbol{p}i - 1) \\ \boldsymbol{z}_i \end{bmatrix} & \text{for a \textbf{revolute} joint} \end{cases}$$

Differential kinematics . . .

B: Map joint velocities to **time derivative of position and angles** in a minimal representation of orientation — obtained by time differentiation of $\boldsymbol{x} = [x, y, z, \alpha, \beta, \gamma]^T$

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \frac{\mathrm{d}\boldsymbol{f}(\boldsymbol{q})}{\mathrm{d}t} = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}} \frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}t}$$

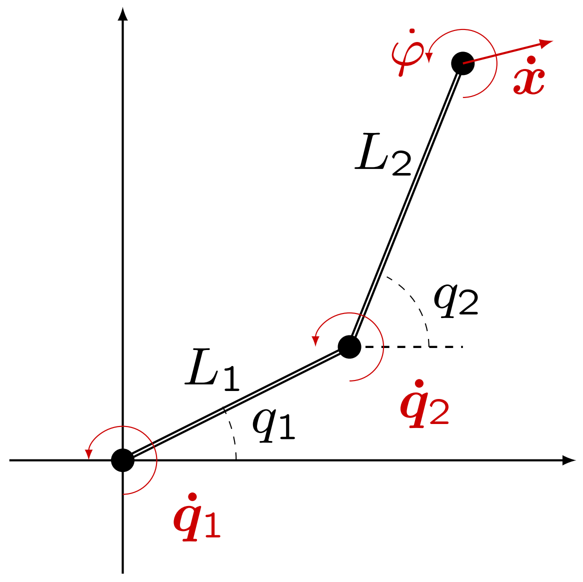
$$\boxed{\dot{\boldsymbol{x}} = \mathbf{J}_A \dot{\boldsymbol{q}}}$$

$$\mathbf{J}_A = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}}$$ $(m \times n)$ analytical Jacobian matrix

Note that in general

$$\boldsymbol{\omega} \neq \frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t}$$ and $$\mathbf{J}(\boldsymbol{q}) = \mathbf{T}(\boldsymbol{q})\mathbf{J}_A(\boldsymbol{q})$$

Jacobians for planar 2R robot



Geometric Jacobian:

$$\mathbf{J} = \begin{bmatrix} -L_1 \sin(q_1) - L_2 \sin(q_1 + q_2) & -L_2 sin(q_1 + q_2) \\ L_1 \cos(q_1) + L_2 \cos(q_1 + q_2) & L_2 cos(q_1 + q_2) \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$

Analytical Jacobian:

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} = \begin{bmatrix} L_1 \cos(q_1) + L_2 \cos(q_1 + q_2) \\ L_1 \sin(q_1) + L_2 \sin(q_1 + q_2) \\ q_1 + q_2 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} -L_1 \sin(q_1) - L_2 \sin(q_1 + q_2) & -L_2 sin(q_1 + q_2) \\ L_1 \cos(q_1) + L_2 \cos(q_1 + q_2) & L_2 cos(q_1 + q_2) \\ 1 & 1 \end{bmatrix}$$

Acceleration and higher order relations

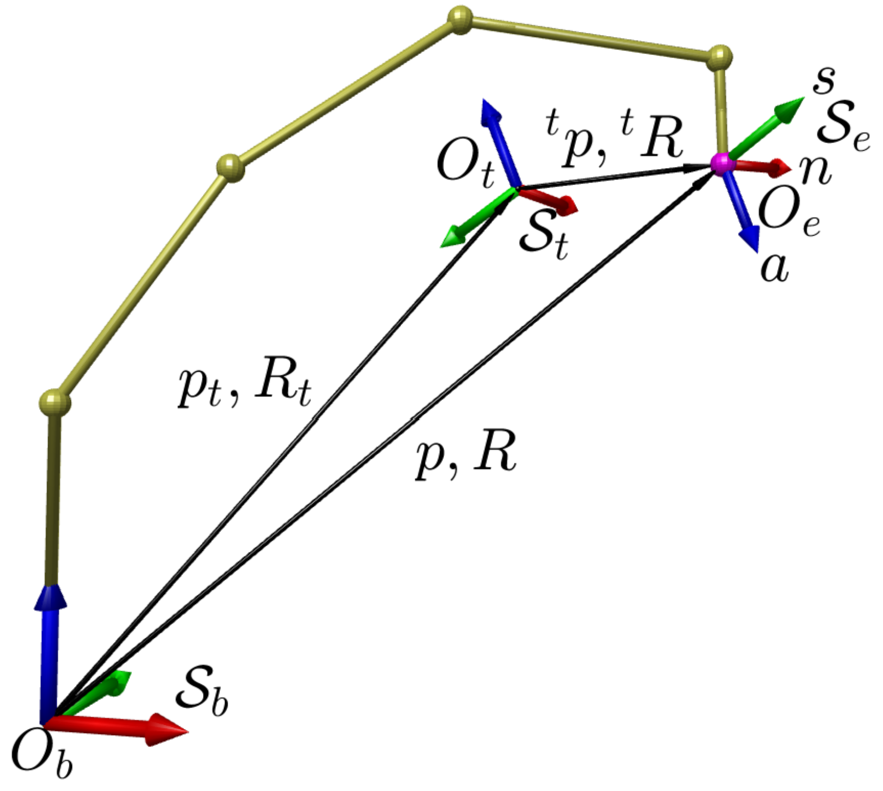
Differential relationships between motion in the joint space and motion in the task space can be established at acceleration level (second order) or at higher orders

velocity	$\dot{\boldsymbol{x}} = \mathbf{J}_A \dot{\boldsymbol{q}}$
acceleration	$\ddot{\boldsymbol{x}} = \mathbf{J}_A \ddot{\boldsymbol{q}} + \dot{\mathbf{J}}_A \dot{\boldsymbol{q}}$
jerk	$\dddot{\boldsymbol{x}} = \mathbf{J}_A \dddot{\boldsymbol{q}} + 2\dot{\mathbf{J}}_A \ddot{\boldsymbol{q}} + \ddot{\mathbf{J}}_A \dot{\boldsymbol{q}}$
snap	$\ddddot{\boldsymbol{x}} = \mathbf{J}_A \ddddot{\boldsymbol{q}} + \ldots$

The same holds also for the instantaneous velocity, accelerations, . . .

velocity	$\boldsymbol{v} = \mathbf{J} \dot{\boldsymbol{q}}$
acceleration	$\dot{\boldsymbol{v}} = \mathbf{J} \ddot{\boldsymbol{q}} + \dot{\mathbf{J}} \dot{\boldsymbol{q}}$
jerk	$\ddot{\boldsymbol{v}} = \mathbf{J} \dddot{\boldsymbol{q}} + 2\dot{\mathbf{J}} \ddot{\boldsymbol{q}} + \ddot{\mathbf{J}} \dot{\boldsymbol{q}}$
snap	$\dddot{\boldsymbol{v}} = \mathbf{J} \ddddot{\boldsymbol{q}} + \ldots$

Kinematics in arbitrary task frame



End-effector location ${}^t\mathbf{T}$ in $\mathcal{S}_t$ can be expressed as

$$\begin{aligned}
{}^t\mathbf{T} &= \begin{bmatrix} {}^t\mathbf{R} & {}^t\boldsymbol{p} \\ \mathbf{0} & 1 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{R}_t^T & -\mathbf{R}_t^T\boldsymbol{p}_t \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \boldsymbol{p} \\ \mathbf{0} & 1 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{R}_t^T\mathbf{R} & \mathbf{R}_t^T(\boldsymbol{p} - \boldsymbol{p}_t) \\ \mathbf{0} & 1 \end{bmatrix}
\end{aligned}$$

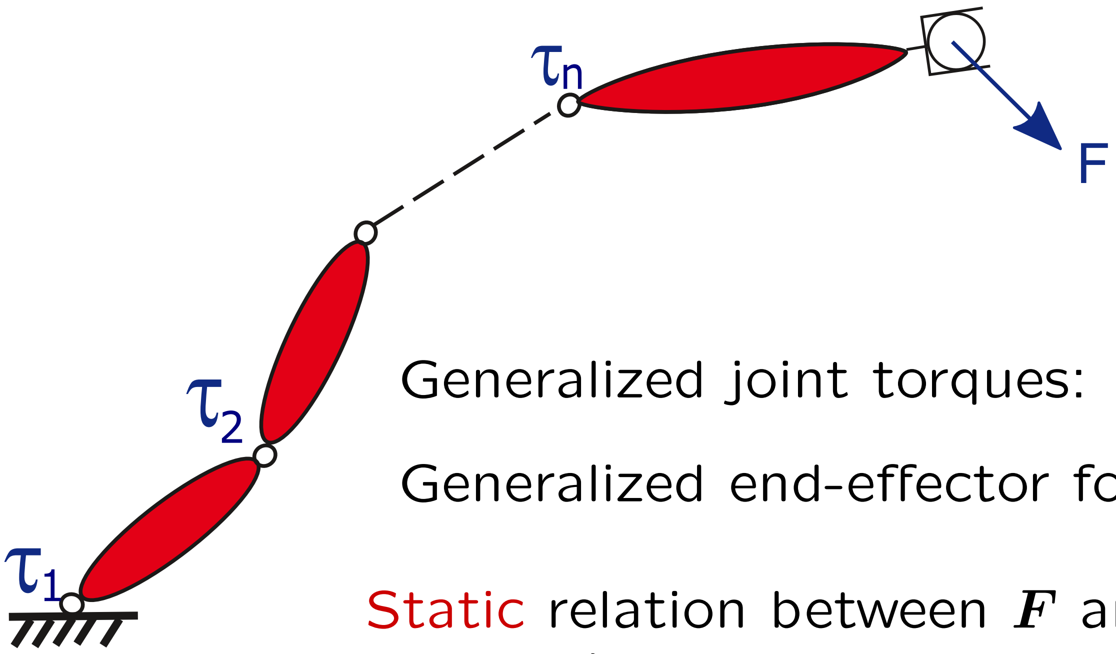
Velocities:

$${}^t\boldsymbol{v} = \widetilde{\mathbf{R}}_t^T\boldsymbol{v} - \widetilde{\mathbf{R}}_t^T\mathbf{J}_t\boldsymbol{v}_t\,,$$

$$\widetilde{\mathbf{R}}_t = \begin{bmatrix} \mathbf{R}_t & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{R}_t \end{bmatrix} \qquad \mathbf{J}_t = \begin{bmatrix} \mathbf{I}_{3\times3} & -\mathbf{S}(\boldsymbol{p} - \boldsymbol{p}_t) \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix}$$

Statics

The goal of statics is to determine the relationship between the generalized **forces acting on the end-effector** and generalized **joint torques**.



Generalized joint torques: $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_n]^T$

Generalized end-effector forces: $\boldsymbol{F} = [F_x, F_y, F_z, M_x, M_y, M_z]^T$

Static relation between $\boldsymbol{F}$ and $\boldsymbol{\tau}$ (obtained using virtual work principle):

$$\boxed{\boldsymbol{\tau} = \mathbf{J}^T \boldsymbol{F}}$$

Dynamics of rigid bodies

Euler-Lagrange approach:

Energy based; Simpler and more intuitive, and also more suitable to understand the effects of changes in the mechanical parameters; Analytical model. Drawbacks: Not computationally efficient.

Using total kinetic energy $\mathcal{T}$ and potential energy $\mathcal{U}$ the Lagrangian of the rigid body is defined as:

$$\mathcal{L} = \mathcal{T} - \mathcal{U}$$

The generalized forces $\boldsymbol{\tau}$ are calculated using the Lagrangian equation:

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

Newton-Euler approach:

Balance of forces/torques; Efficient recursive algorithm; Suitable for real-time control. Drawback: Numeric solution (non closed form)

Robot dynamic models

Dynamic models provide the relation between the **generalized forces** (joint and end-effector!) and the **motion** of the robot considering all dynamic properties of the system.

Direct dynamics: From known generalized forces $\boldsymbol{\tau}(t)$ and $\boldsymbol{F}_e(t)$ determine the motion $\ddot{\boldsymbol{q}}(t)$ ($\ddot{\boldsymbol{q}}(t)$:

$$\boldsymbol{\tau}(t), \boldsymbol{F}_e(t) \Rightarrow \ddot{\boldsymbol{q}}(t)$$

Inverse dynamics: From known motion determine generalized forces $\boldsymbol{\tau}(t)$:

$$\ddot{\boldsymbol{q}}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{q}(t) \Rightarrow \boldsymbol{\tau}(t)$$

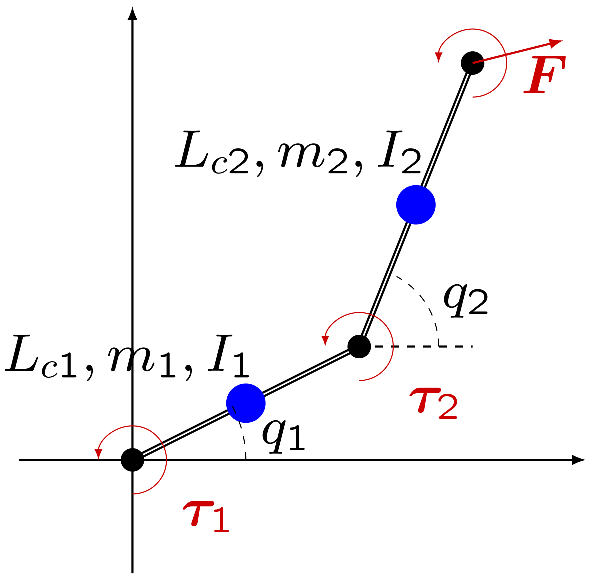
Robot dynamic models . . .

Using **Lagrangian formulation** and considering contribution of external forces the **equations of motion** of a robot are given in the form

$$\boxed{\boldsymbol{\tau} = \mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{q}}) + \mathbf{J}^T\boldsymbol{F}}$$

$\mathbf{H}(\boldsymbol{q})$	$\mathbb{R}^{n\times n}$	inertia matrix (symmetric, positive definite)
$\mathbf{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$	$\mathbb{R}^{n}$	vector of centripetal and Coriolis forces
$\boldsymbol{g}(\boldsymbol{q})$	$\mathbb{R}^{n}$	vector of gravity forces
$\boldsymbol{D}(\dot{\boldsymbol{q}})$	$\mathbb{R}^{n}$	vector of friction forces
$\boldsymbol{F}$	$\mathbb{R}^{m}$	vector of external forces/torques

Dynamic model of 2R robot



$$
\begin{aligned}
\mathcal{T}_1 &= \frac{1}{2}(m_1 L_{c1}^2 + I_1)\dot{q}_1^2 \\
\mathcal{U}_1 &= m_1 g L_{c1} \sin(q_1) \\
\mathcal{T}_2 &= \frac{1}{2}(m_2 \boldsymbol{x}_1^T \boldsymbol{x}_1 + I_2 \dot{q}_1^2) \\
\mathcal{U}_2 &= m_2 g (L_1 \sin(q_1) + L_{c2} \sin(q_1 + q_2)) \\
\boldsymbol{x}_1^T \boldsymbol{x}_1 &= L_1 \dot{q}_1^2 + L_{c2}^2(\dot{q}_1^2 + \dot{q}_2^2) + 2 L_1 L_{c2} \cos(q_2)(\dot{q}_1^2 + \dot{q}_1 \dot{q}_2^2)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{H} &= \begin{bmatrix} (m_1 L_{c1}^2 + m_2(L_1 + L_{c2}^2 + 2 L_1 L_{c2} \cos(q_2)) + I_1 & 0 \\ 0 & m_2(L_{c2}^2 + L_1 L_{c2} \cos(q_2)) + I_2 \end{bmatrix} \\
\mathbf{C} &= (-m_2 L - 1 L_{c2} \sin(q_2)) \begin{bmatrix} \dot{q}_2 & (\dot{q}_1 + \dot{q}_2) \\ -\dot{q}_1 & 0 \end{bmatrix} \\
g &= \begin{bmatrix} m_1 g L_{c1} \cos(q_1) + m_2 g (L_1 \cos(q_1) + L_{c2} \cos(q_1 + q_2)) \\ m_2 g L_{c2} \cos(q_1 + q_2) \end{bmatrix}
\end{aligned}
$$