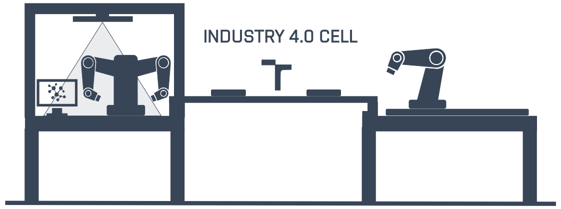
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

Programming for robots and manipulators
Lecture 5

Roman Parak



NETME Centre
FACULTY OF MECHANICAL ENGINEERING institute of automation and computer science

Brno, 2023

Content


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA STROJNÍHO INŽENÝRSTVÍ

Introduction

Kinematics describes the analytical relationship between the joint positions and the end-effector position and orientation.

Differential Kinematics describes the analytical relationship between the joint motion and the end-effector motion in terms of velocities.

Robot dynamics is concerned with the relationship between the forces acting on a robot mechanism and the accelerations they produce.

Typically, the robot mechanism is modelled as a rigid-body system, in which case robot dynamics is the application of rigid-body dynamics to robots.

The two main problems in robot dynamics are:
A) Forward dynamics – Given the forces, work out the accelerations.
B) Inverse dynamics – Given the accelerations, work out the forces.

Differential Kinematics

Differential Kinematics

Differential kinematics relates the velocities of the manipulator components. These velocities may be the velocities at the joints of the manipulator or the velocities of one or more links in the manipulator kinematic chain.

The standard approach to differential kinematics is to relate joint and end-effector velocities through the Jacobian matrix, which allows the calculation of the end-effector velocities given the joint velocities (Forward differential kinematics) or, to determine the joint velocities in order to move the end-effector with a prescribed speed (Inverse differential kinematics).

Inverse Matrix of $J(\theta)^{-1}$
$\dot{\theta}_1, \dot{\theta}_2, \dots, \dot{\theta}_n$

Forward Differential Kinematics $\longrightarrow$

Jacobian Matrix $J(\theta)$
$\dot{x}, \dot{y}, \dot{z}, \dots$

Inverse Matrix of $J(\theta)^{-1}$
$\dot{\theta}_1, \dot{\theta}_2, \dots, \dot{\theta}_n$

$\longleftarrow$ Inverse Differential Kinematics

Jacobian Matrix $J(\theta)$
$\dot{x}, \dot{y}, \dot{z}, \dots$

Jacobian is a multidimensional form of the derivative, in robotics, Jacobian is a time-varying linear transformations, which determines the relationship between joint velocity (input) and end-effector velocity (output).

$$\dot{x} = J(\theta)\dot{\theta}$$

Endpoint Velocity | Jacobian Matrix | Joint Velocity

Most manipulator have values of θ where the Jacobian becomes singular. If the Jacobian matrix is non singular, we can calculate "Inverse Jacobian matrix".

$$\dot{\theta} = J(\theta)^{-1}\dot{x}$$

The Jacobian is singular when its determinant is equal to 0 → $\det(J) = 0$.

The Jacobian is defined as:

$$J(\theta) = \begin{bmatrix} \frac{\partial x_1}{\partial \theta_1} & \cdots & \frac{\partial x_1}{\partial \theta_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_m}{\partial \theta_1} & \cdots & \frac{\partial x_m}{\partial \theta_n} \end{bmatrix}$$

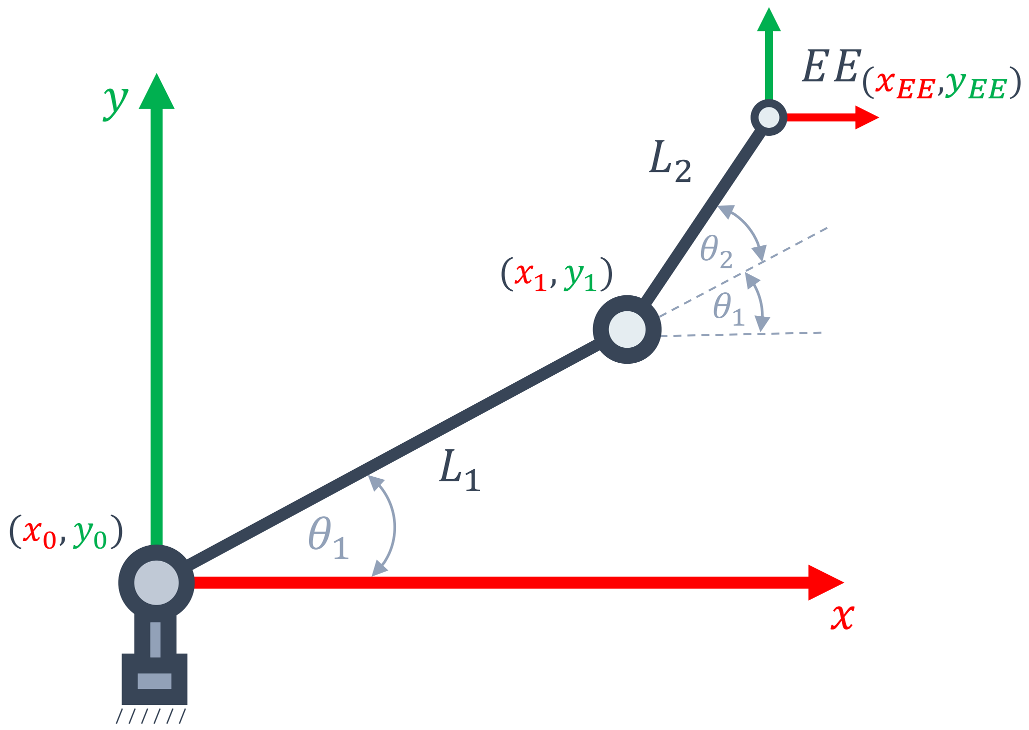
where $\boldsymbol{J(\theta)}$ is the $\boldsymbol{m\ x\ n}$ analytical Jacobian matrix.

Jacobians are useful for planning and executing smooth trajectories, determining singular configurations, executing coordinated anthropomorphic motion, deriving dynamic equations of motion, and transforming forces and torques from the end-effector to the
manipulator joints.

Example

Differential Inverse Kinematics

$$(\theta_1, \theta_2) \leftarrow (x_{EE}, y_{EE})$$



Forward Kinematics

$$(\theta_1, \theta_2) \rightarrow (x_{EE}, y_{EE})$$

$$x_1 = L_1 cos\theta_1$$
$$y_1 = L_1 sin\theta_1$$

$$x = L_1 cos\theta_1 + L_2 cos(\theta_1 + \theta_2)$$
$$y = L_1 sin\theta_1 + L_2 sin(\theta_1 + \theta_2)$$

Jacobian Matrix

$$\begin{bmatrix} p\dot{x} \\ p\dot{y} \end{bmatrix} = J(\theta_1, \theta_2) \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

Jacobian Matrix

$$\begin{bmatrix} p\dot{x} \\ p\dot{y} \end{bmatrix} = J(\theta_1, \theta_2) \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} =$$

$$= \begin{bmatrix} -L_1 sin\theta_1 - L_2 sin(\theta_1 + \theta_2) & -L_2 sin(\theta_1 + \theta_2) \\ L_1 cos\theta_1 + L_2 cos(\theta_1 + \theta_2) & L_2 cos(\theta_1 + \theta_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

Singular Jacobian

$$\det\left(\begin{bmatrix} -L_1 sin\theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}\right) = 0$$

Dynamics

The dynamics of a system describes how the controls θ_t influence the change-of-state of the system:

$$x_{t+1} = f(x_t, \theta_t)$$

The notation x_t refers to the dynamic state of the system: e.g., joint positions and velocities $x_t = (\theta_t, \dot{\theta}_t)$. Function f is an arbitrary function, often smooth.

Describes the relationship between Forces ($\vec{f}_n$)/Torques ($\vec{\tau}_n$):

Joint Torques
$\vec{\tau}_1, \vec{\tau}_2, \ldots$

Forward Dynamics →

Joint Motion
$\vec{\theta}_1, \vec{\theta}_2, ..; \dot{\vec{\theta}}_1, \dot{\vec{\theta}}_1, ..$

Joint Torques
$\vec{\tau}_1, \vec{\tau}_2, \ldots$

← Inverse Dynamics

Joint Motion
$\vec{\theta}_1, \vec{\theta}_2, ..; \dot{\vec{\theta}}_1, \dot{\vec{\theta}}_1, ..$

For many applications with fixed-based robots we need to find a multi-body dynamics formulated as:

$$M(\theta)\ddot{\theta} + b\left(\theta, \dot{\theta}\right) + g(\theta) = \tau + J_c(\theta)^T F_C$$

$M(\theta)$ → Generalized mass matrix (orthogonal).
$\theta, \dot{\theta}, \ddot{\theta}$ → Generalized position, velocity and acceleration vectors.
$b(\theta, \dot{\theta})$ → Coriolis and centrifugal terms.
$g(\theta)$ → Gravitational terms.
τ → External generalized forces.
$J_c(\theta)$ → Geometric Jacobian corresponding to the external forces.
F_C → External Cartesian forces (e.g. from contacts).

There are typically two ways to derive the equation of motion for an open-chain robot:

Lagrangian Method
Energy-based method.
Dynamic equations in closed form.
Often used for study of dynamic properties and analysis of control methods.

Newton-Euler Method
Balance of forces/torques.
Dynamic equations in numeric/recursive form.
Often used for numerical solution of forward/inverse dynamics.

Many other formal methods based on basic principles in mechanics are available for the derivation of the robot dynamic model:
Principle of d'Alembert, of Hamilton, of virtual works, etc.

Lagrange Method is one of the analytical methods used to describe the motion of physical systems (closely tied to both the d'Alembert and Hamilton principles).

The method is centered around three fundamental concepts:

1. The definition of generalized coordinates θ and generalized velocities $\dot{\theta}$, which may or may not encode the information regarding the constraints applicable to the system.

2. A scalar function called the Lagrangian function $\mathcal{L}$. For mechanical systems, itis exactly the difference between the total Kinetic energy T and the total Potential energy U, of the system at each instant:

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

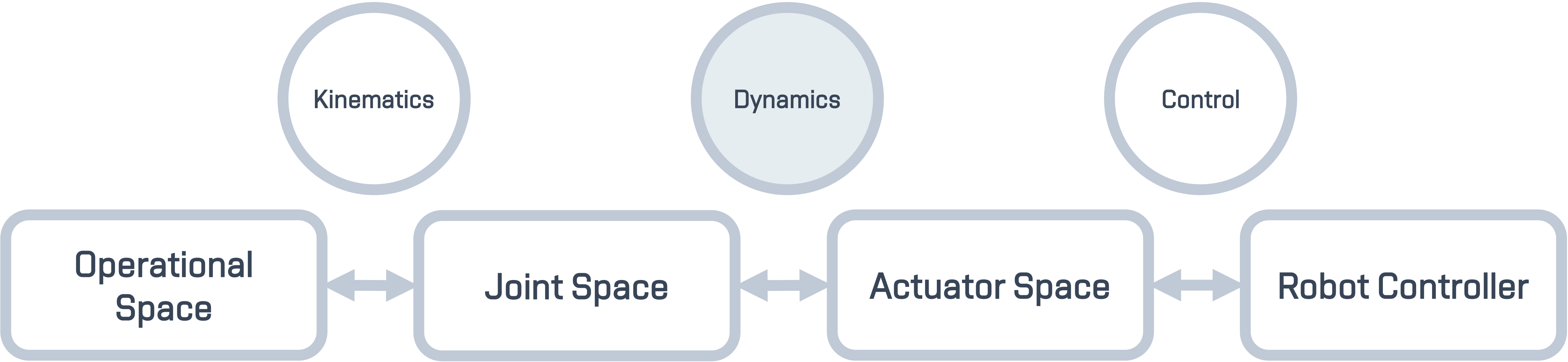
3. The so-called Euler-Lagrange equation, also known as the Euler-Lagrange of the second kind, which applies to the Lagrangian function $\mathcal{L}$ and to the total external generalized forces τ:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}}\right) - \left(\frac{\partial \mathcal{L}}{\partial \theta}\right) = \tau$$

In the most general case, the Lagrangian is a function of the generalized coordinates and velocities θ and $\dot{\theta}$, and it may also have an explicit dependence on time t, hence we redefine the aforementioned scalar energy functions as:

$$T = T(t, \theta, \dot{\theta})$$
$$U = U(t, \theta)$$
$$\mathcal{L} = \mathcal{L}(t, \theta, \dot{\theta})$$

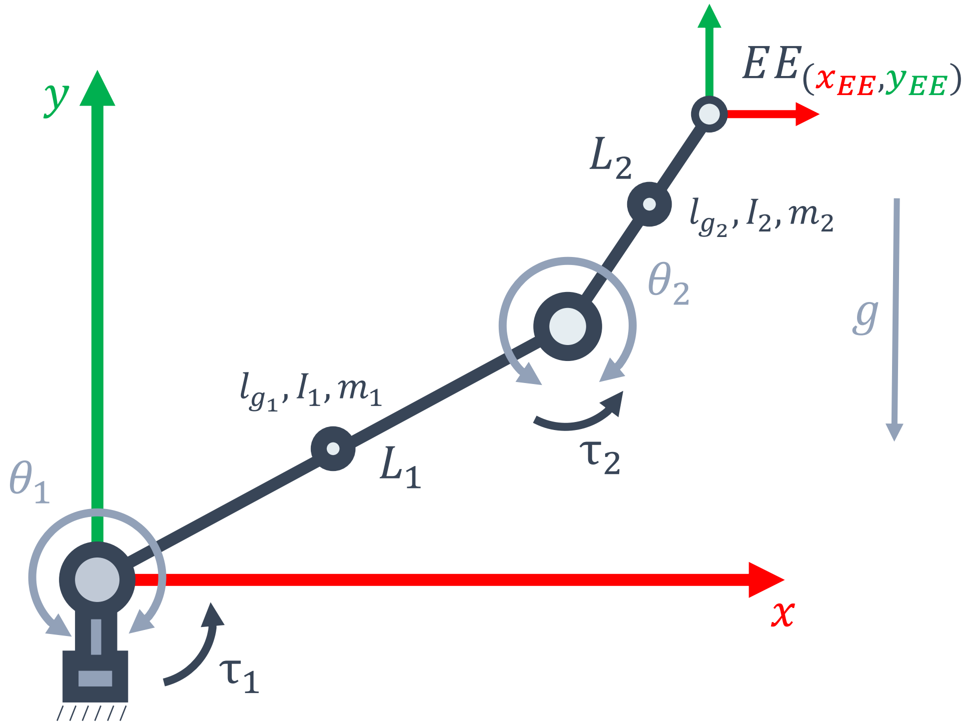
Robotic Control


Kinematics
Dynamics
Control
Operational Space
Joint Space
Actuator Space
Robot Controller

Example

Differential Inverse Kinematics

$$\tau = M(\theta)\ddot{\theta} + b(\theta,\dot{\theta}) + g(\theta)$$



Lagrangian Method

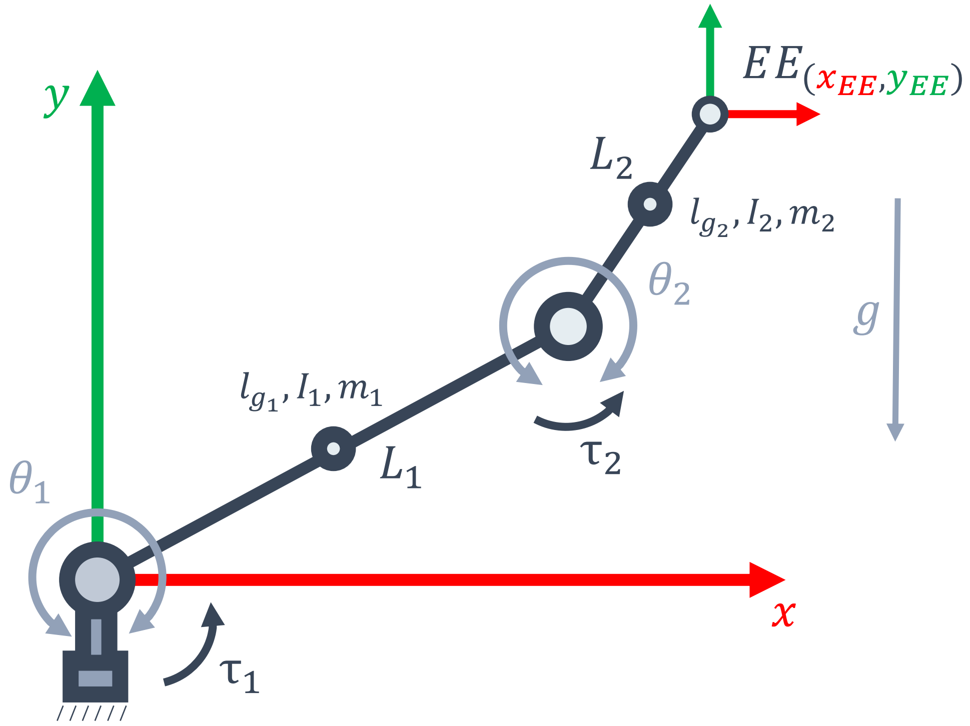
$$\tau_i = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_i}\right) - \left(\frac{\partial \mathcal{L}}{\partial \theta_i}\right)$$

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

$$\mathcal{L}(\theta,\dot{\theta}) = T(\theta,\dot{\theta}) - U(\theta)$$

Differential Inverse Kinematics

$$\tau = M(\theta)\ddot{\theta} + b(\theta, \dot{\theta}) + g(\theta)$$



Forward Kinematics

$$(\theta_1, \theta_2) \rightarrow (x_{EE}, y_{EE})$$

$$x_1 = L_1 cos\theta_1$$
$$y_1 = L_1 sin\theta_1$$

$$x = L_1 cos\theta_1 + L_2 cos(\theta_1 + \theta_2)$$
$$y = L_1 sin\theta_1 + L_2 sin(\theta_1 + \theta_2)$$

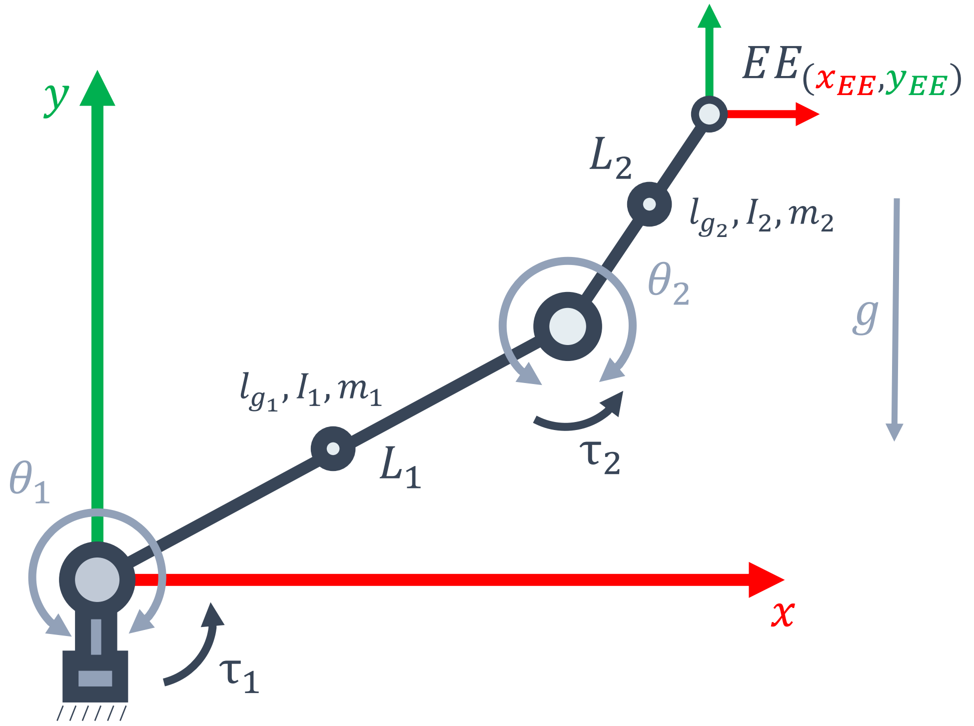
$$x_{m_1} = l_{g_1}\cos(\theta_1) \qquad \dot{x}_{m_1} = -l_{g_1}\dot{\theta}_1\sin(\theta_1)$$
$$y_{m_1} = l_{g_1}\sin(\theta_1) \qquad \dot{y}_{m_1} = l_{g_1}\dot{\theta}_1\cos(\theta_1)$$

$$x_{m_2} = L_1\cos(\theta_1) + l_{g_2}\cos(\theta_1 + \theta_2)$$
$$y_{m_2} = L_1\sin(\theta_1) + l_{g_2}\sin(\theta_1 + \theta_2)$$

$$\dot{x}_{m_2} = -L_1\dot{\theta}_1\sin(\theta_1) - l_{g_2}(\dot{\theta}_1 + \dot{\theta}_2)sin(\theta_1 + \theta_2)$$
$$\dot{y}_{m_2} = L_1\dot{\theta}_1\cos(\theta_1) + l_{g_2}(\dot{\theta}_1 + \dot{\theta}_2)cos(\theta_1 + \theta_2)$$

Differential Inverse Kinematics

$$\tau = M(\theta)\ddot{\theta} + b\left(\theta, \dot{\theta}\right) + g(\theta)$$



Kinetic energy

$$T_1 = \frac{1}{2} m_1 \left(\dot{x}_{m_1}^2 + \dot{y}_{m_1}^2\right) + \frac{1}{2} I_1 \dot{\theta}_1^2$$

$$T_2 = \frac{1}{2} m_2 \left(\dot{x}_{m_2}^2 + \dot{y}_{m_2}^2\right) + \frac{1}{2} I_2 (\dot{\theta}_1 + \dot{\theta}_2)^2$$

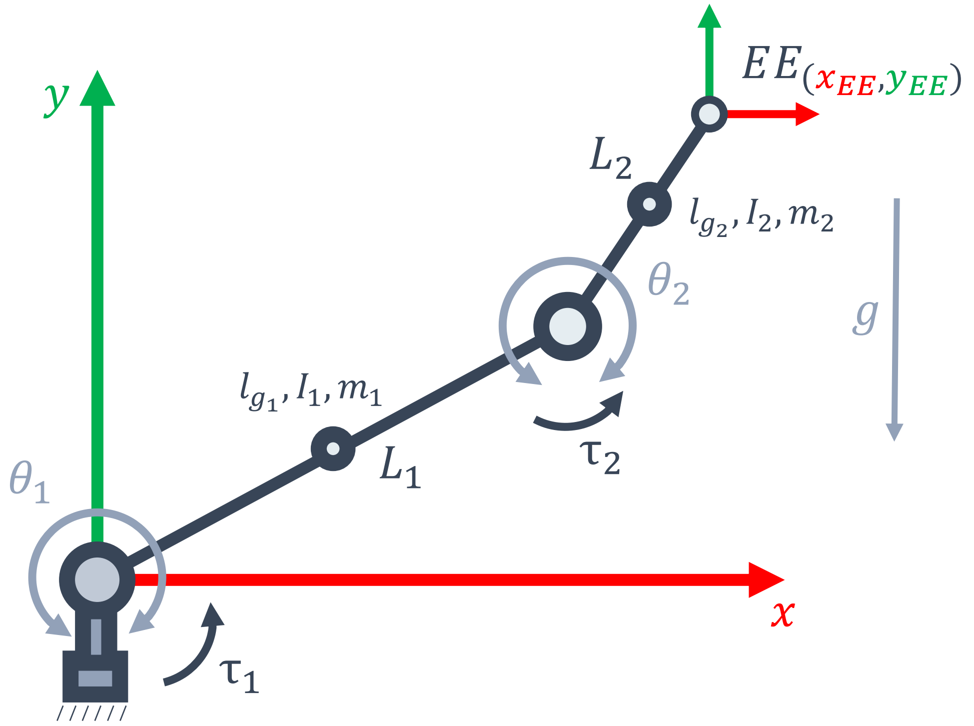
Potential energy

$$U_1 = m_1 l_{g_1} g sin(\theta_1)$$

$$U_2 = m_2 g(L_1 sin(\theta_1) + l_{g_2} \sin(\theta_1 + \theta_2))$$

Differential Inverse Kinematics

$$\tau = M(\theta)\ddot{\theta} + b(\theta,\dot{\theta}) + g(\theta)$$



Torque equation

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

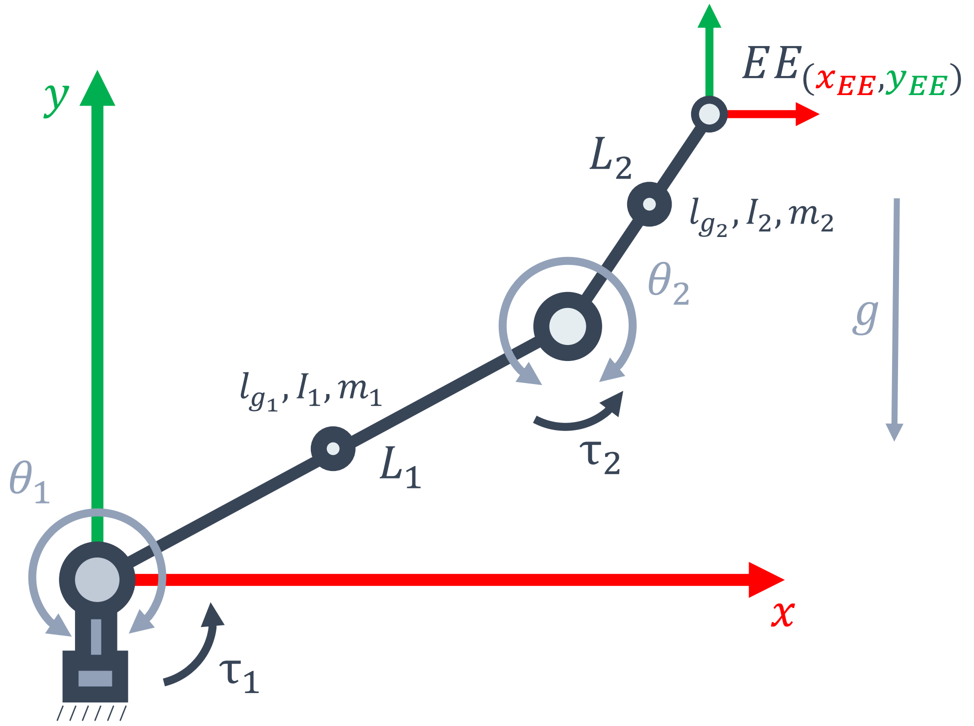
$$\tau_i = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_i}\right) - \left(\frac{\partial \mathcal{L}}{\partial \theta_i}\right)$$

$$\mathcal{L} = (T_1 + T_2) - (U_1 + U_1)$$

$$\tau = [\tau_1; \tau_2]^T, \theta = [\theta_1; \theta_2]^T$$

Differential Inverse Kinematics

$$\tau = M(\theta)\ddot{\theta} + b(\theta, \dot{\theta}) + g(\theta)$$



Torque equation

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

$$\tau_i = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_i}\right) - \left(\frac{\partial \mathcal{L}}{\partial \theta_i}\right)$$

$$\mathcal{L} = (T_1 + T_2) - (U_1 + U_1)$$

$$\tau = [\tau_1; \tau_2]^T, \theta = [\theta_1; \theta_2]^T$$

Torque equations can then be split up into the form of the general equation. The different parameters can be extracted from this, by isolating the angular acceleration, velocities and positions.

$$\tau = M(\theta)\ddot{\theta} + b\left(\theta, \dot{\theta}\right) + g(\theta)$$

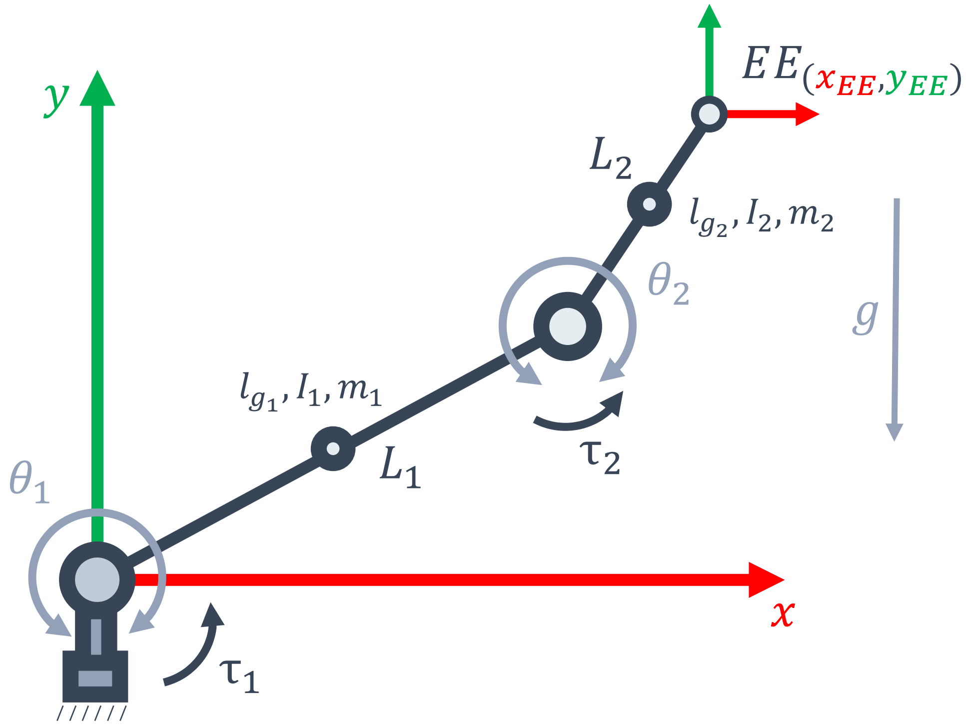
$$M(\theta) = \begin{bmatrix} I_1 + I_2 + m_1 l_{g_1}^2 + m_2(L_1^2 + l_{g_2}^2 + 2L_1 l_{g_2} + 2L_1 l_{g_2}\cos(\theta_2)) & I_2 + m_2(l_{g_2}^2 + L_1 l_{g_2}\cos(\theta_2)) \\ I_2 + m_2(l_{g_2}^2 + L_1 l_{g_2}\cos(\theta_2)) & I_2 + m_2 l_{g_2}^2 \end{bmatrix}$$

$$b\left(\theta, \dot{\theta}\right) = \begin{bmatrix} -m_2 L_1 l_{g_2}\dot{\theta}_2\left(2\dot{\theta}_1 + \dot{\theta}_2\right)\sin(\theta_2) \\ m_2 L_1 l_{g_2}\dot{\theta}_1^2 \sin(\theta_2) \end{bmatrix}$$

$$g(\theta) = \begin{bmatrix} m_1 g l_{g_1}\cos(\theta_1) + m_2 g(L_1\cos(\theta_1) + l_{g_2}\cos(\theta_1 + \theta_2)) \\ m_2 g l_{g_2}\cos(\theta_1 + \theta_2) \end{bmatrix}$$

Differential Inverse Kinematics

$$\tau = M(\theta)\ddot{\theta} + b(\theta,\dot{\theta}) + g(\theta)$$



Motion equation

$$\tau = M(\theta)\ddot{\theta} + b(\theta,\dot{\theta}) + g(\theta)$$

$$\ddot{\theta} = M(\theta)^{-1}\left(-b(\theta,\dot{\theta}) - g(\theta) + \tau\right)$$

Ordinary Differential Equations (ODE)

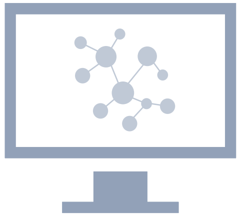
$$\frac{d}{dt}\begin{bmatrix}\theta \\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\dot{\theta} \\ M(\theta)^{-1}\left(-b(\theta,\dot{\theta}) - g(\theta) + \tau\right)\end{bmatrix}$$

Why do we need knowledge of Kinematics and Dynamics?

Design and Implementation of a Robotic Application

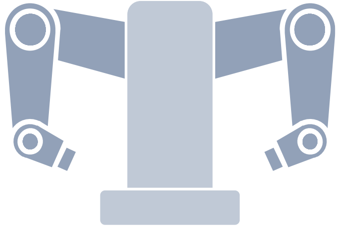
SOFTWARE

- Simulation Software
- Control Software
- Additional Libraries
- ?



HARDWARE

- Robotic Construction
- Controller
- Additional Hardware
- ?



NETME Centre

FACULTY OF MECHANICAL ENGINEERING institute of automation and computer science

Thank You!

Questions?

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE