



Robotics: Dynamics of open chain

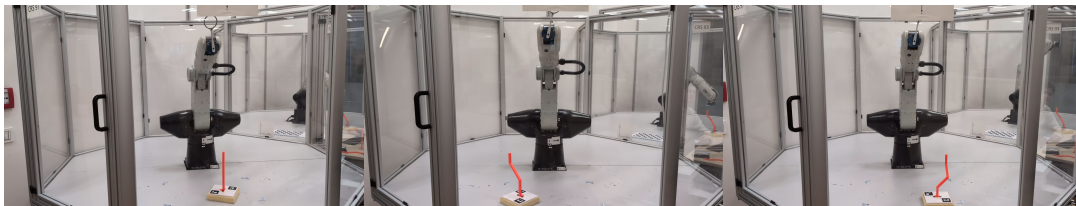
Vladimír Petřík

vladimir.petrík@cvut.cz

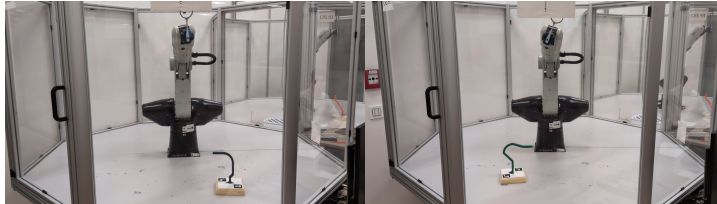
24.11.2024

Final project

- 8 teams evaluated (3 x 20p; 2x 23p; 3x 25p)



Final project



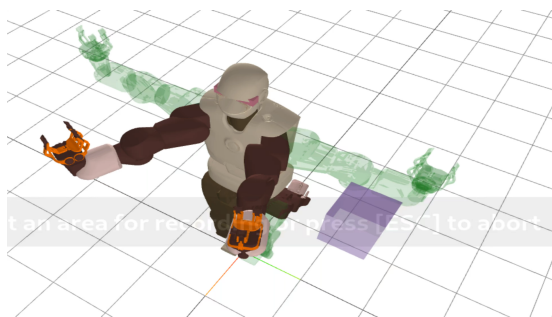
Motivation

- ▶ We studied kinematics of open chains
 - ▶ Forward kinematics
 - ▶ Inverse kinematics
 - ▶ Planning of paths/trajectories

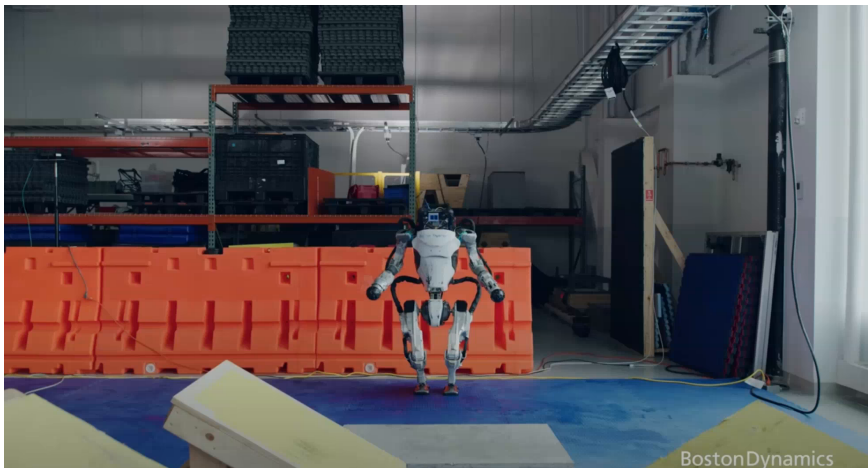


Motivation

- ▶ We studied kinematics of open chains
 - ▶ Forward kinematics
 - ▶ Inverse kinematics
 - ▶ Planning of paths/trajectories
- ▶ Dynamics of open chains
 - ▶ Motion of the robot taking into account forces, torques, and gravity
 - ▶ Motion described by the equation of motion
 - ▶ Can be used to compute control of the robot
 - ▶ It can answer the question when humanoid robot falls down



Motivation



Equation of motion

- ▶ Describes the motion of the robot
- ▶ Differential equation of the second order
- ▶ For robotics, equation of motion has the form $\tau = M(\mathbf{q})\ddot{\mathbf{q}} + h(\mathbf{q}, \dot{\mathbf{q}})$
 - ▶ τ - vector of joint forces/torques
 - ▶ M - mass matrix
 - ▶ h - vector of Coriolis, gravity and friction terms
 - ▶ h is often in the form $h = C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q})$
 - ▶ C - Coriolis matrix
 - ▶ g - effect of gravity



Dynamics tasks

- ▶ Forward dynamics

- ▶ Inverse dynamics



Dynamics tasks

- ▶ Forward dynamics
 - ▶ Given $\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}$ compute $\ddot{\mathbf{q}}$
 - ▶ Why we need it?

- ▶ Inverse dynamics



Dynamics tasks



- ▶ Forward dynamics
 - ▶ Given \mathbf{q} , $\dot{\mathbf{q}}$, $\boldsymbol{\tau}$ compute $\ddot{\mathbf{q}}$
 - ▶ Why we need it?
 - ▶ Used for simulation
 - ▶ How the robot moves for given forces/torques
 - ▶ $\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q})(\boldsymbol{\tau} - \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}))$
- ▶ Inverse dynamics



Dynamics tasks



- ▶ Forward dynamics
 - ▶ Given \mathbf{q} , $\dot{\mathbf{q}}$, $\boldsymbol{\tau}$ compute $\ddot{\mathbf{q}}$
 - ▶ Why we need it?
 - ▶ Used for simulation
 - ▶ How the robot moves for given forces/torques
 - ▶ $\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q})(\boldsymbol{\tau} - \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}))$
- ▶ Inverse dynamics
 - ▶ Given \mathbf{q} , $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ compute $\boldsymbol{\tau}$
 - ▶ Why we need it?



Dynamics tasks



- ▶ Forward dynamics
 - ▶ Given \mathbf{q} , $\dot{\mathbf{q}}$, $\boldsymbol{\tau}$ compute $\ddot{\mathbf{q}}$
 - ▶ Why we need it?
 - ▶ Used for simulation
 - ▶ How the robot moves for given forces/torques
 - ▶ $\ddot{\mathbf{q}} = M^{-1}(\mathbf{q})(\boldsymbol{\tau} - h(\mathbf{q}, \dot{\mathbf{q}}))$
- ▶ Inverse dynamics
 - ▶ Given \mathbf{q} , $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ compute $\boldsymbol{\tau}$
 - ▶ Why we need it?
 - ▶ Used for control
 - ▶ What forces/torques are needed to move the robot in desired way
 - ▶ $\boldsymbol{\tau} = M(\mathbf{q})\ddot{\mathbf{q}} + h(\mathbf{q}, \dot{\mathbf{q}})$



Forward dynamics integration - simulation

- ▶ Explicit Euler Integration
- ▶ $\dot{\mathbf{q}}_{t+1} = \dot{\mathbf{q}}_t + \ddot{\mathbf{q}}_t \Delta t$
 - ▶ $\ddot{\mathbf{q}}_t = M^{-1}(\mathbf{q}_t)(\boldsymbol{\tau}_t - \mathbf{h}(\mathbf{q}_t, \dot{\mathbf{q}}_t))$
 - ▶ Δt - time step, e.g. 0.001 s (unstable for large time steps)
- ▶ $\mathbf{q}_{t+1} = \mathbf{q}_t + \dot{\mathbf{q}}_{t+1} \Delta t$



$$\boldsymbol{\tau} = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



$$\boldsymbol{\tau} = \begin{pmatrix} 1 & 1 \end{pmatrix}^T$$

Equation of motion derivation

- ▶ Lagrangian formulation
 - ▶ Kinetic energy
 - ▶ Potential energy
 - ▶ Elegant for simple structures
- ▶ Newton-Euler formulation
 - ▶ Dynamic equation of rigid body
 - ▶ Efficient recursive formulation for forward/inverse dynamics
- ▶ Both formulations lead to the same equation of motion



Lagrangian formulation

- ▶ Generalized coordinates \mathbf{q}

- ▶ Generalized forces $\boldsymbol{\tau}$

- ▶ Lagrangian

$$\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) = \mathcal{K}(\mathbf{q}, \dot{\mathbf{q}}) - \mathcal{P}(\mathbf{q})$$

- ▶ Kinetic energy $\mathcal{K}(\mathbf{q}, \dot{\mathbf{q}})$

- ▶ Potential energy $\mathcal{P}(\mathbf{q})$

- ▶ Equation of motion

$$\boldsymbol{\tau} = \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} - \frac{\partial \mathcal{L}}{\partial \mathbf{q}}$$

- ▶ Also called Euler-Lagrange equation with external forces

- ▶ Examples:

- ▶ Particle of mass moving vertically in gravitation field

- ▶ Planar robot arm



Simulation of PP



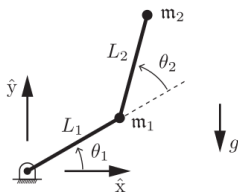
$$\tau = (0 \quad 0)^\top$$



$$\tau = (0 \quad -100y_G)^\top$$



Equation of Motion - RR



$$\begin{aligned}\tau_1 = & \left(m_1 L_1^2 + m_2 (L_1^2 + 2L_1 L_2 \cos \theta_2 + L_2^2) \right) \ddot{\theta}_1 \\ & + m_2 (L_1 L_2 \cos \theta_2 + L_2^2) \ddot{\theta}_2 - m_2 L_1 L_2 \sin \theta_2 (2\dot{\theta}_1 \dot{\theta}_2 + \dot{\theta}_2^2) \\ & + (m_1 + m_2) L_1 g \cos \theta_1 + m_2 g L_2 \cos(\theta_1 + \theta_2),\end{aligned}$$

$$\begin{aligned}\tau_2 = & m_2 (L_1 L_2 \cos \theta_2 + L_2^2) \ddot{\theta}_1 + m_2 L_2^2 \ddot{\theta}_2 + m_2 L_1 L_2 \dot{\theta}_1^2 \sin \theta_2 \\ & + m_2 g L_2 \cos(\theta_1 + \theta_2).\end{aligned}$$

$$M(\theta) = \begin{bmatrix} m_1 L_1^2 + m_2 (L_1^2 + 2L_1 L_2 \cos \theta_2 + L_2^2) & m_2 (L_1 L_2 \cos \theta_2 + L_2^2) \\ m_2 (L_1 L_2 \cos \theta_2 + L_2^2) & m_2 L_2^2 \end{bmatrix},$$

$$c(\theta, \dot{\theta}) = \begin{bmatrix} -m_2 L_1 L_2 \sin \theta_2 (2\dot{\theta}_1 \dot{\theta}_2 + \dot{\theta}_2^2) \\ m_2 L_1 L_2 \dot{\theta}_1^2 \sin \theta_2 \end{bmatrix},$$

$$g(\theta) = \begin{bmatrix} (m_1 + m_2) L_1 g \cos \theta_1 + m_2 g L_2 \cos(\theta_1 + \theta_2) \\ m_2 g L_2 \cos(\theta_1 + \theta_2) \end{bmatrix},$$



Simulation of RRR



$$\tau = (0 \quad 0 \quad 0)^T$$



$$\tau = (10 \quad 10 \quad 10)^T$$



Understanding mass matrix

- ▶ Kinetic energy
 - ▶ Point mass $\frac{1}{2}m\dot{x}^2$
 - ▶ Robot $\frac{1}{2}\dot{\mathbf{q}}^\top M(\mathbf{q})\dot{\mathbf{q}}$



Understanding mass matrix

- ▶ Kinetic energy
 - ▶ Point mass $\frac{1}{2}m\dot{x}^2$
 - ▶ Robot $\frac{1}{2}\dot{\mathbf{q}}^\top M(\mathbf{q})\dot{\mathbf{q}}$
- ▶ Mass
 - ▶ Point mass m is positive
 - ▶ $M(\mathbf{q})$ is symmetric positive definite matrix



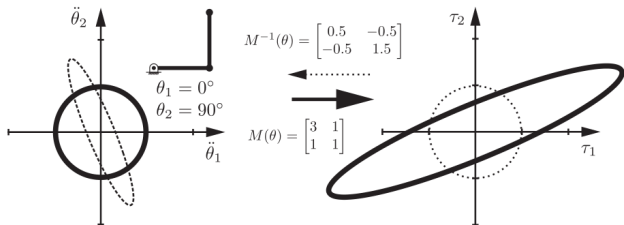
Understanding mass matrix

- ▶ Kinetic energy
 - ▶ Point mass $\frac{1}{2}m\dot{x}^2$
 - ▶ Robot $\frac{1}{2}\dot{\mathbf{q}}^\top \mathbf{M}(\mathbf{q})\dot{\mathbf{q}}$
- ▶ Mass
 - ▶ Point mass m is positive
 - ▶ $\mathbf{M}(\mathbf{q})$ is symmetric positive definite matrix
- ▶ Point mass in Cartesian coordinates
 - ▶ Independent of direction of acceleration
 - ▶ Acceleration is scalar multiplication of force



Understanding mass matrix

- ▶ Kinetic energy
 - ▶ Point mass $\frac{1}{2}m\dot{x}^2$
 - ▶ Robot $\frac{1}{2}\dot{\mathbf{q}}^\top M(\mathbf{q})\dot{\mathbf{q}}$
- ▶ Mass
 - ▶ Point mass m is positive
 - ▶ $M(\mathbf{q})$ is symmetric positive definite matrix
- ▶ Point mass in Cartesian coordinates
 - ▶ Independent of direction of acceleration
 - ▶ Acceleration is scalar multiplication of force
- ▶ Mass matrix in generalized coordinates
 - ▶ Effective mass depends on the acceleration direction
 - ▶ Unit acceleration mapping to torques
 - ▶ The same magnitude of acceleration can be achieved by different torques (depending on the direction)



End-effector effective mass

- ▶ How massy would end-effector feel if we move it by hand? Depends on the direction of force.
 - ▶ Kinetic energy must be constant: $\frac{1}{2}V^\top \Lambda(\mathbf{q})V = \frac{1}{2}\dot{\mathbf{q}}^\top M(\mathbf{q})\dot{\mathbf{q}}$
 - ▶ $\Lambda(\mathbf{q})$ effective mass of end-effector
 - ▶ $V = (\dot{x}, \dot{y})^\top$ velocity of end-effector



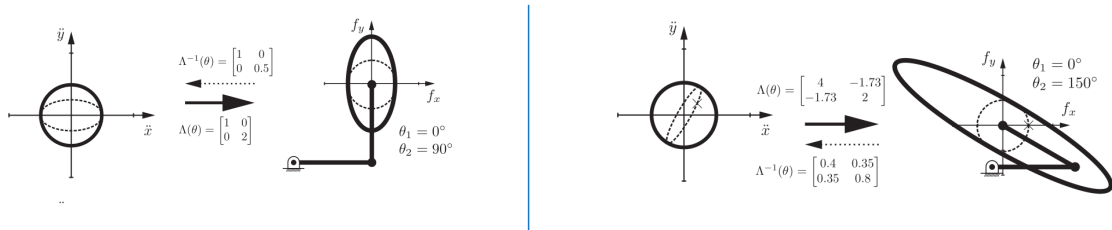
End-effector effective mass

- ▶ How massy would end-effector feel if we move it by hand? Depends on the direction of force.
 - ▶ Kinetic energy must be constant: $\frac{1}{2}V^\top \Lambda(\mathbf{q})V = \frac{1}{2}\dot{\mathbf{q}}^\top M(\mathbf{q})\dot{\mathbf{q}}$
 - ▶ $\Lambda(\mathbf{q})$ effective mass of end-effector
 - ▶ $V = (\dot{x}, \dot{y})^\top$ velocity of end-effector
 - ▶ Jacobian $V = J(\mathbf{q})\dot{\mathbf{q}}$
 - ▶ $V^\top \Lambda(\mathbf{q})V = (J^{-1}V)^\top M(\mathbf{q})(J^{-1}V) = V^\top (J^{-\top} M(\mathbf{q}) J^{-1}) V$
 - ▶ End-effector mass matrix: $\Lambda(\mathbf{q}) = J^{-\top}(\mathbf{q}) M(\mathbf{q}) J^{-1}(\mathbf{q})$



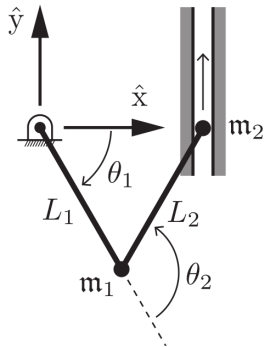
End-effector effective mass

- ▶ How massy would end-effector feel if we move it by hand? Depends on the direction of force.
 - ▶ Kinetic energy must be constant: $\frac{1}{2}V^\top \Lambda(q)V = \frac{1}{2}\dot{q}^\top M(q)\dot{q}$
 - ▶ $\Lambda(q)$ effective mass of end-effector
 - ▶ $V = (\dot{x}, \dot{y})^\top$ velocity of end-effector
 - ▶ Jacobian $V = J(q)\dot{q}$
 - ▶ $V^\top \Lambda(q)V = (J^{-1}V)^\top M(q)(J^{-1}V) = V^\top (J^{-\top}M(q)J^{-1})V$
 - ▶ End-effector mass matrix: $\Lambda(q) = J^{-\top}(q)M(q)J^{-1}(q)$



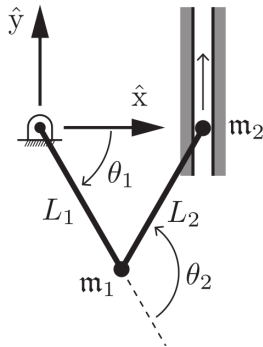
Constrained dynamics

- ▶ Robot subject to a set of k velocity constraints
 - ▶ e.g. closed kinematics chain
 - ▶ writing with a pen (constant height)
 - ▶ $A(\mathbf{q})\dot{\mathbf{q}} = 0, A \in \mathbb{R}^{k \times n}$



Constrained dynamics

- ▶ Robot subject to a set of k velocity constraints
 - ▶ e.g. closed kinematics chain
 - ▶ writing with a pen (constant height)
 - ▶ $A(\mathbf{q})\dot{\mathbf{q}} = 0, A \in \mathbb{R}^{k \times n}$
- ▶ Equation of motion
 - ▶ $\boldsymbol{\tau} = M(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) + A^\top(\mathbf{q})\boldsymbol{\lambda}, \quad \text{s.t. } A(\mathbf{q})\dot{\mathbf{q}} = 0$
 - ▶ $\boldsymbol{\lambda}$ - vector of Lagrange multipliers
 - ▶ $A^\top(\mathbf{q})\boldsymbol{\lambda}$ - force applied against constraints expressed as joint forces/torques
 - ▶ Lambda can be computed analytically:
$$\boldsymbol{\lambda} = (AM^{-1}A^\top)^{-1}(AM^{-1}(\boldsymbol{\tau} - \mathbf{h}) + \dot{A}\dot{\mathbf{q}})$$



Constrained dynamics tasks

- ▶ Forward dynamics
 - ▶ first compute λ
 - ▶ compute \ddot{q}



Constrained dynamics tasks

- ▶ Forward dynamics
 - ▶ first compute λ
 - ▶ compute \ddot{q}
- ▶ Inverse dynamics
 - ▶ compute τ from given λ and \ddot{q}
 - ▶ λ defines force against constraints
 - ▶ if constraint is in the end-effector space: $J^\top \mathbf{f} = A^\top \lambda$
 - ▶ e.g. how much pushing against the table with \mathbf{f}_d
 - ▶ $\lambda = (J^{-\top} A^\top)^\dagger \mathbf{f}_d$



Use of constrained dynamics

Proximal and Sparse Resolution of Constrained Dynamic Equations

Justin Carpentier

Rohan Budhiraja

Nicolas Mansard

Robotics: Science and Systems
2021



Summary

- ▶ Dynamics of open chains
- ▶ Equation of motion
 - ▶ Lagrangian formulation
 - ▶ Newton-Euler formulation
- ▶ Forward dynamics
- ▶ Inverse dynamics
- ▶ Constrained dynamics

