Robotics 2

Dynamic model of robots:

Analysis, properties, extensions, uses

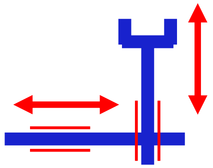
Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

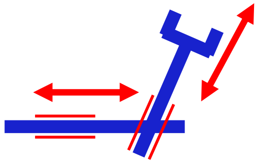
Analysis of inertial couplings

- Cartesian robot



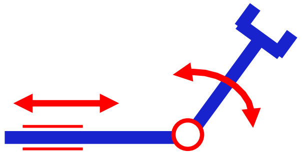
$$M = \begin{pmatrix} m_{11} & 0 \\ 0 & m_{22} \end{pmatrix}$$

- Cartesian "skew" robot



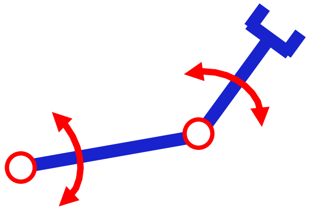
$$M = \begin{pmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{pmatrix}$$

- PR robot



$$M = \begin{pmatrix} m_{11} & m_{12}(q_2) \\ m_{12}(q_2) & m_{22} \end{pmatrix}$$

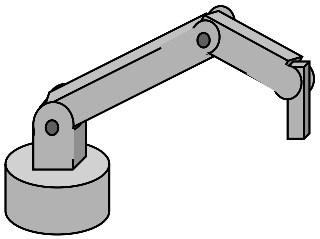
- 2R robot



$$M = \begin{pmatrix} m_{11}(q_2) & m_{12}(q_2) \\ m_{12}(q_2) & m_{22} \end{pmatrix}$$

- 3R articulated robot

  (under simplifying
  assumptions on the CoMs)



$$M = \begin{pmatrix} m_{11}(q_2, q_3) & 0 & 0 \\ 0 & m_{22}(q_3) & m_{23}(q_3) \\ 0 & m_{23}(q_3) & m_{33} \end{pmatrix}$$

Analysis of gravity term

- absence of gravity
  - constant U_g (motion on horizontal plane)
  - applications in remote space
- static balancing
  - distribution of masses (including motors)
- mechanical compensation
  - articulated system of springs
  - closed kinematic chains

$$\Rightarrow \quad g(q) \approx 0$$

Bounds on dynamic terms

- for an open-chain (serial) manipulator, there always exist positive real constants k_0 to k_7 such that, for any value of q and $\dot{q}$

$$k_0 \le \|M(q)\| \le k_1 + k_2\|q\| + k_3\|q\|^2$$

inertia matrix

$$\|S(q,\dot{q})\| \le (k_4 + k_5\|q\|)\ \|\dot{q}\|$$

factorization matrix of Coriolis/centrifugal terms

$$\|g(q)\| \le k_6 + k_7\|q\|$$

gravity vector

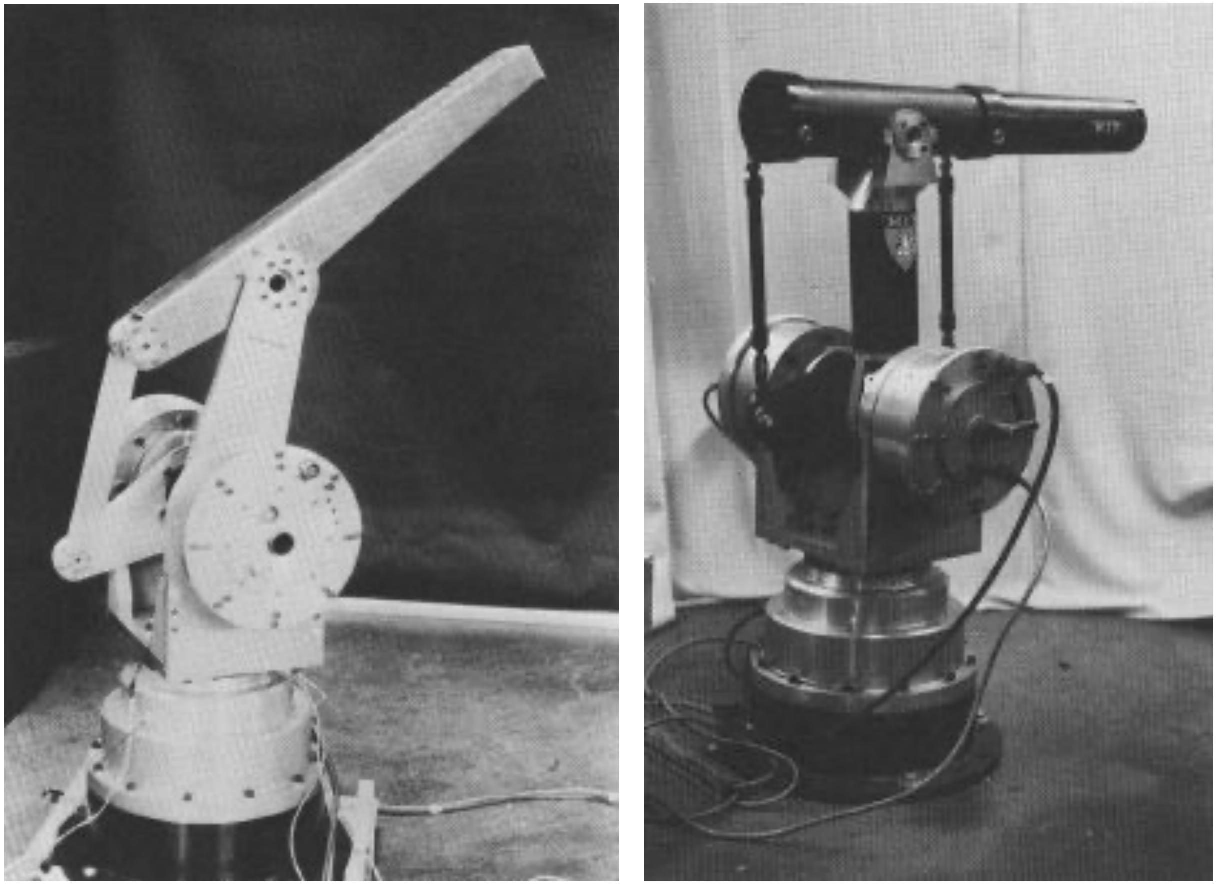
- if the robot has only **revolute** joints, these simplify to

$$\boxed{k_0 \le \|M(q)\| \le k_1 \quad \|S(q,\dot{q})\| \le k_4\|\dot{q}\| \quad \|g(q)\| \le k_6}$$

(the same holds true with bounds $q_{i,min} \le q_i \le q_{i,max}$ on prismatic joints)

NOTE: norms are either for vectors or for matrices (induced norms)

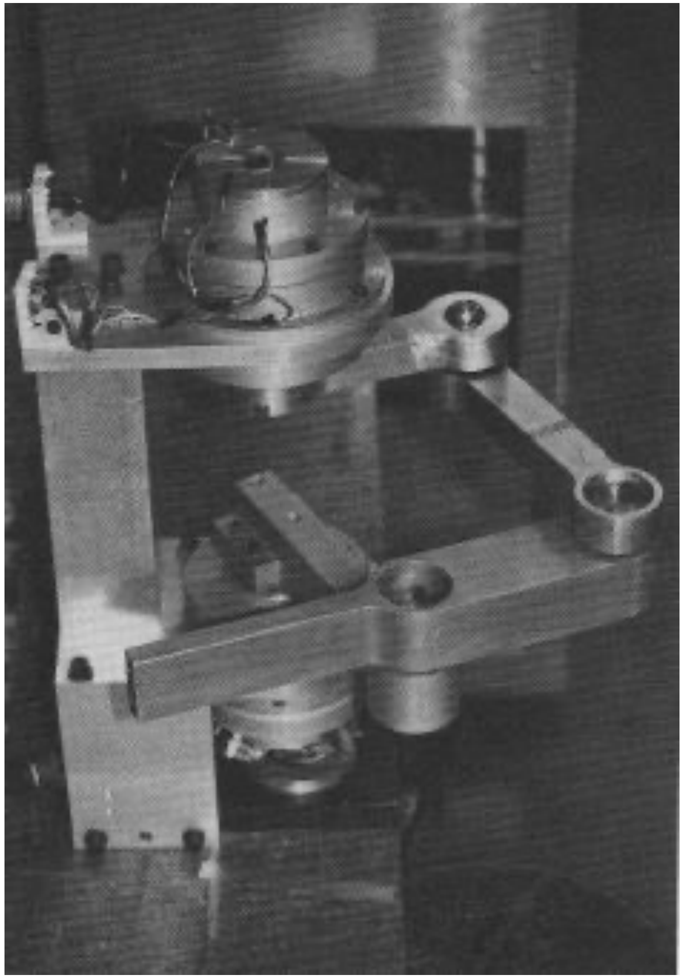
Robots with closed kinematic chains - 1



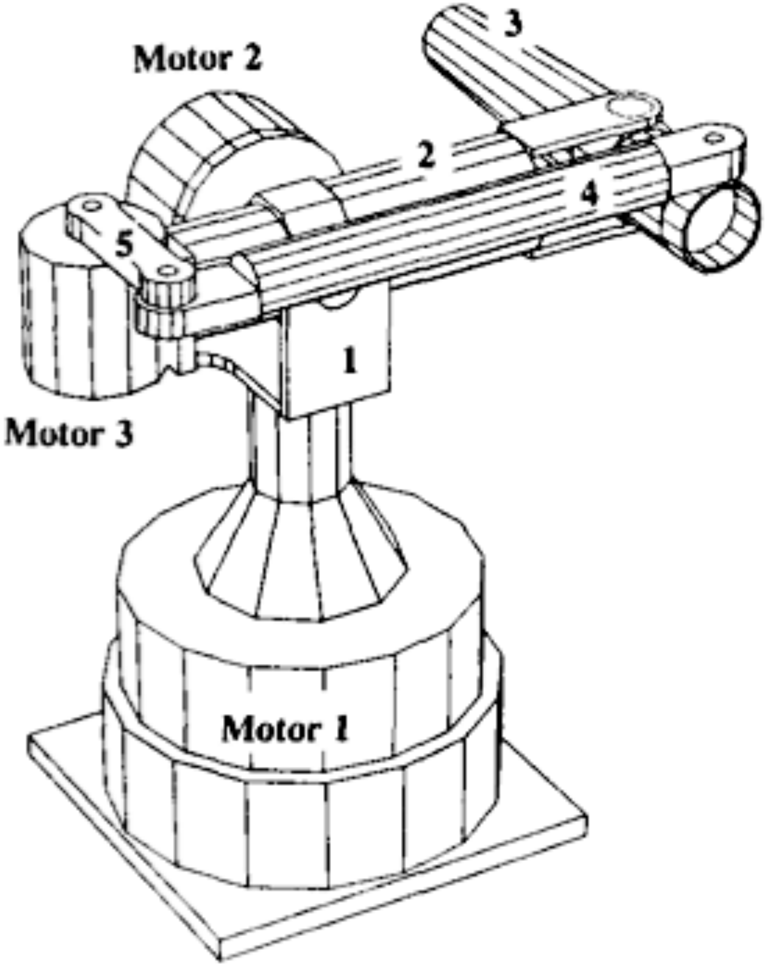
Comau Smart NJ130

MIT Direct Drive Mark II and Mark III

Robots with closed kinematic chains - 2



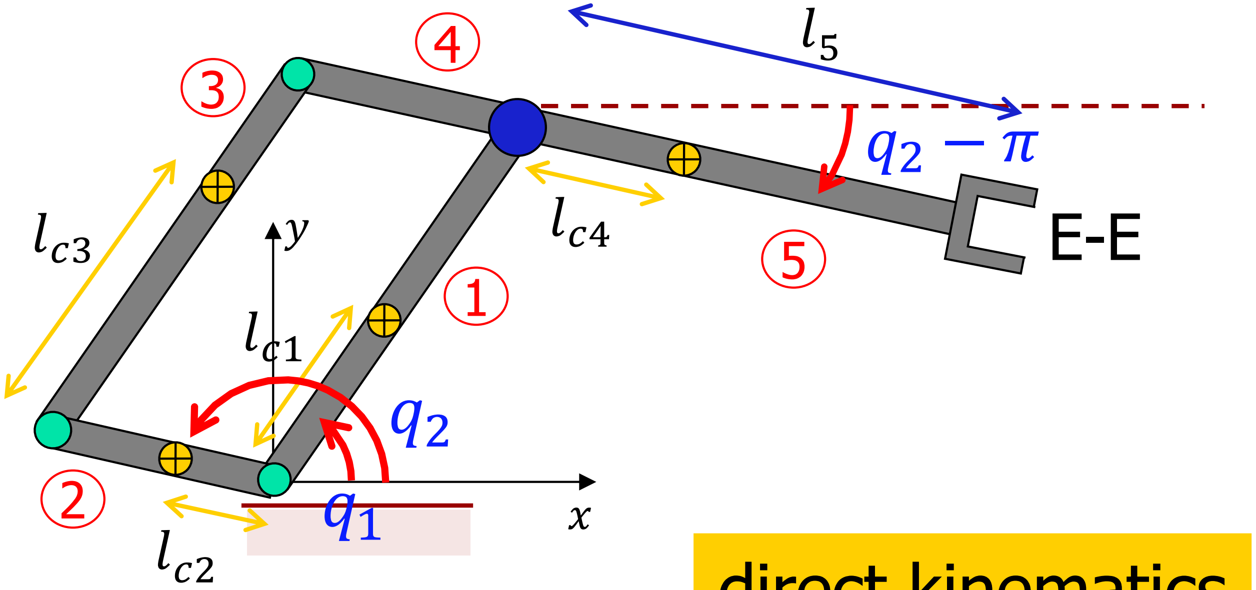
MIT Direct Drive Mark IV
(planar five-bar linkage)



UMinnesota Direct Drive Arm
(spatial five-bar linkage)

Robot with parallelogram structure

(planar) kinematics and dynamics



⊕ center of mass: arbitrary l_{ci}

parallelogram:

$$l_1 = l_3$$
$$l_2 = l_4$$

direct kinematics

$$p_{EE} = \begin{pmatrix} l_1 c_1 \\ l_1 s_1 \end{pmatrix} + \begin{pmatrix} l_5 \cos(q_2 - \pi) \\ l_5 \sin(q_2 - \pi) \end{pmatrix} = \begin{pmatrix} l_1 c_1 \\ l_1 s_1 \end{pmatrix} - \begin{pmatrix} l_5 c_2 \\ l_5 s_2 \end{pmatrix}$$

position of center of masses

$$p_{c1} = \begin{pmatrix} l_{c1} c_1 \\ l_{c1} s_1 \end{pmatrix} \quad p_{c2} = \begin{pmatrix} l_{c2} c_2 \\ l_{c2} s_2 \end{pmatrix} \quad p_{c3} = \begin{pmatrix} l_2 c_2 \\ l_2 s_2 \end{pmatrix} + \begin{pmatrix} l_{c3} c_1 \\ l_{c3} s_1 \end{pmatrix} \quad p_{c4} = \begin{pmatrix} l_1 c_1 \\ l_1 s_1 \end{pmatrix} - \begin{pmatrix} l_{c4} c_2 \\ l_{c4} s_2 \end{pmatrix}$$

Kinetic energy

linear/angular velocities

$$v_{c1} = \begin{pmatrix} -l_{c1}s_1 \\ l_{c1}c_1 \end{pmatrix} \dot{q}_1 \quad v_{c3} = \begin{pmatrix} -l_{c3}s_1 \\ l_{c3}c_1 \end{pmatrix} \dot{q}_1 + \begin{pmatrix} -l_2 s_2 \\ l_2 c_2 \end{pmatrix} \dot{q}_2 \qquad \omega_1 = \omega_3 = \dot{q}_1$$

$$v_{c2} = \begin{pmatrix} -l_{c2}s_2 \\ l_{c2}c_2 \end{pmatrix} \dot{q}_2 \quad v_{c4} = \begin{pmatrix} -l_1 s_1 \\ l_1 c_1 \end{pmatrix} \dot{q}_1 + \begin{pmatrix} l_{c4}s_2 \\ -l_{c4}c_2 \end{pmatrix} \dot{q}_2 \qquad \omega_2 = \omega_4 = \dot{q}_2$$

Note: a (planar) 2D notation is used here!

T_i

$$T_1 = \frac{1}{2} m_1 l_{c1}^2 \dot{q}_1^2 + \frac{1}{2} I_{c1,zz} \dot{q}_1^2 \qquad T_2 = \frac{1}{2} m_2 l_{c2}^2 \dot{q}_2^2 + \frac{1}{2} I_{c2,zz} \dot{q}_2^2$$

$$T_3 = \frac{1}{2} m_3 (l_2^2 \dot{q}_2^2 + l_{c3}^2 \dot{q}_1^2 + 2 l_2 l_{c3} c_{2-1} \dot{q}_1 \dot{q}_2) + \frac{1}{2} I_{c3,zz} \dot{q}_1^2$$

$$T_4 = \frac{1}{2} m_4 (l_1^2 \dot{q}_1^2 + l_{c4}^2 \dot{q}_2^2 - 2 l_1 l_{c4} c_{2-1} \dot{q}_1 \dot{q}_2) + \frac{1}{2} I_{c4,zz} \dot{q}_2^2$$

Robot inertia matrix

$$T = \sum_{i=1}^{4} T_i = \frac{1}{2} \dot{q}^T M(q) \dot{q}$$

$$M(q) = \begin{pmatrix} I_{c1,zz} + m_1 l_{c1}^2 + I_{c3,zz} + m_3 l_{c3}^2 + m_4 l_1^2 & \text{symm} \\ (m_3 l_2 l_{c3} - m_4 l_1 l_{c4}) c_{2-1} & I_{c2,zz} + m_2 l_{c2}^2 + I_{c4,zz} + m_4 l_{c4}^2 + m_3 l_2^2 \end{pmatrix}$$

structural condition
in mechanical design

$$m_3 l_2 l_{c3} = m_4 l_1 l_{c4} \qquad (*)$$

⇓

$M(q)$ diagonal and constant ⇒ centrifugal and Coriolis terms $\equiv 0$

mechanically DECOUPLED and LINEAR
dynamic model (up to the gravity term $g(q)$)

⇔

$$\begin{pmatrix} M_{11} & 0 \\ 0 & M_{22} \end{pmatrix} \begin{pmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

big advantage for the design of motion control laws!

Potential energy and gravity terms

from the y-components of vectors p_{ci}

U_i

$$U_1 = m_1 g_0 l_{c1} s_1 \qquad U_2 = m_2 g_0 l_{c2} s_2$$

$$U_3 = m_3 g_0 (l_2 s_2 + l_{c3} s_1) \qquad U_4 = m_4 g_0 (l_1 s_1 - l_{c4} s_2)$$

$$U = \sum_{i=1}^{4} U_i$$

$$g(q) = \left(\frac{\partial U}{\partial q}\right)^T = \begin{pmatrix} g_0(m_1 l_{c1} + m_3 l_{c3} + m_4 l_1) c_1 \\ g_0(m_2 l_{c2} + m_3 l_2 - m_4 l_{c4}) c_2 \end{pmatrix} = \begin{pmatrix} g_1(q_1) \\ g_2(q_2) \end{pmatrix}$$

gravity components are always "decoupled"

in addition, when (*) holds ⇒

$$\begin{aligned} m_{11}\ddot{q}_1 + g_1(q_1) &= u_1 \\ m_{22}\ddot{q}_2 + g_2(q_2) &= u_2 \end{aligned}$$

u_i are (non-conservative) torques performing work on q_i

further structural conditions in the mechanical design lead to $g(q) \equiv 0$!!

Adding dynamic terms ...

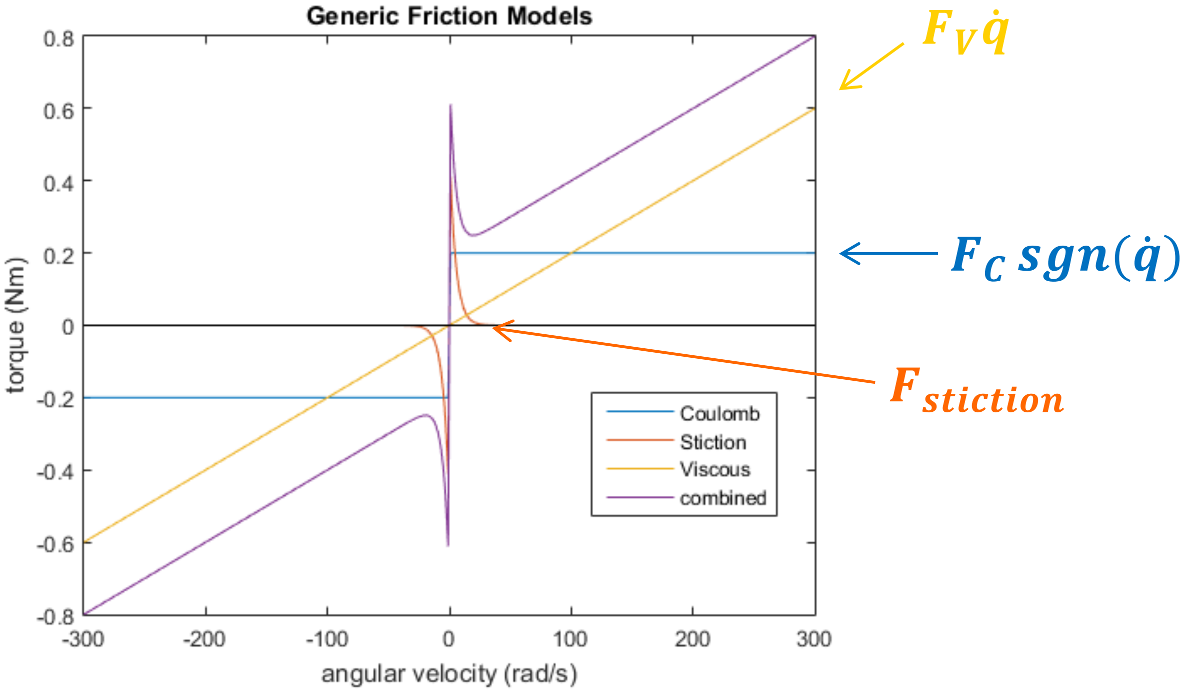
1) dissipative phenomena due to friction at the joints/transmissions
   - viscous, Coulomb, stiction, Stribeck, LuGre (dynamic)...
   - local effects at the joints
   - difficult to model in general, except for:

$$\boxed{u_{V,i} = -F_{V,i}\,\dot{q}_i} \qquad \boxed{u_{C,i} = -F_{C,i}\,\mathrm{sgn}(\dot{q}_i)}$$

in general:

$$u_{diss}^T\,\dot{q} < 0$$

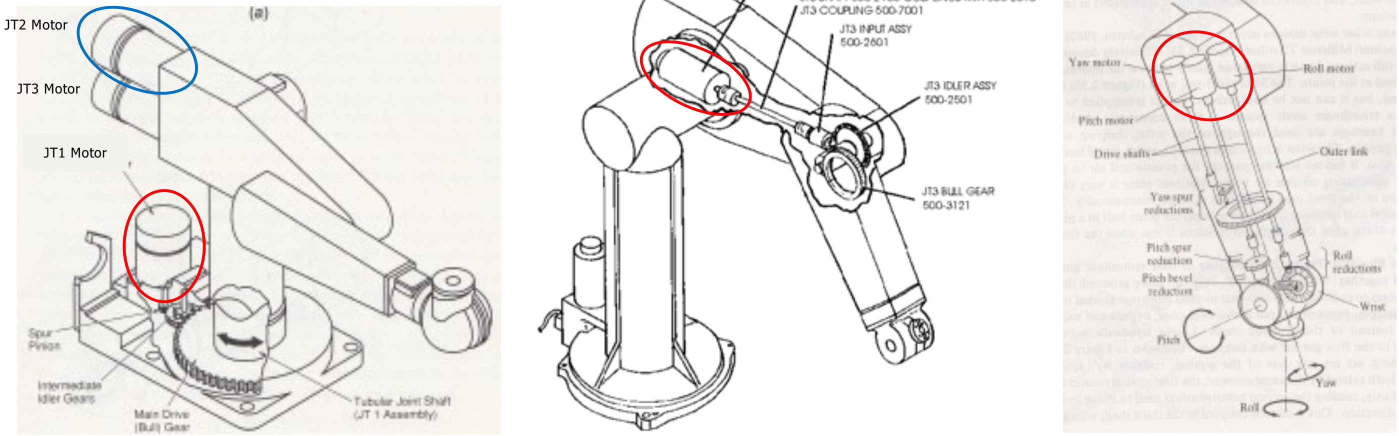
(component-wise too)

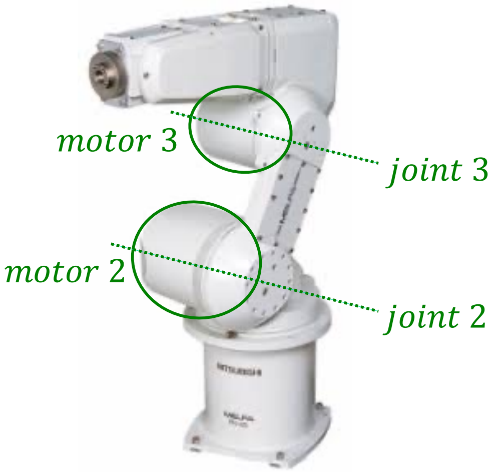


Adding dynamic terms ...

2) inclusion of electrical actuators (as additional rigid bodies)
   - motor i mounted on link $i-1$ (or before), with very few exceptions
   - often with its spinning axis aligned with joint axis i
   - (balanced) mass of motor included in total mass of carrying link
   - (rotor) inertia is to be added to robot kinetic energy
   - transmissions with reduction gears (often, large reduction ratios)
   - in some cases, multiple motors cooperate in moving multiple links: use a transmission coupling matrix Γ (with off-diagonal elements)

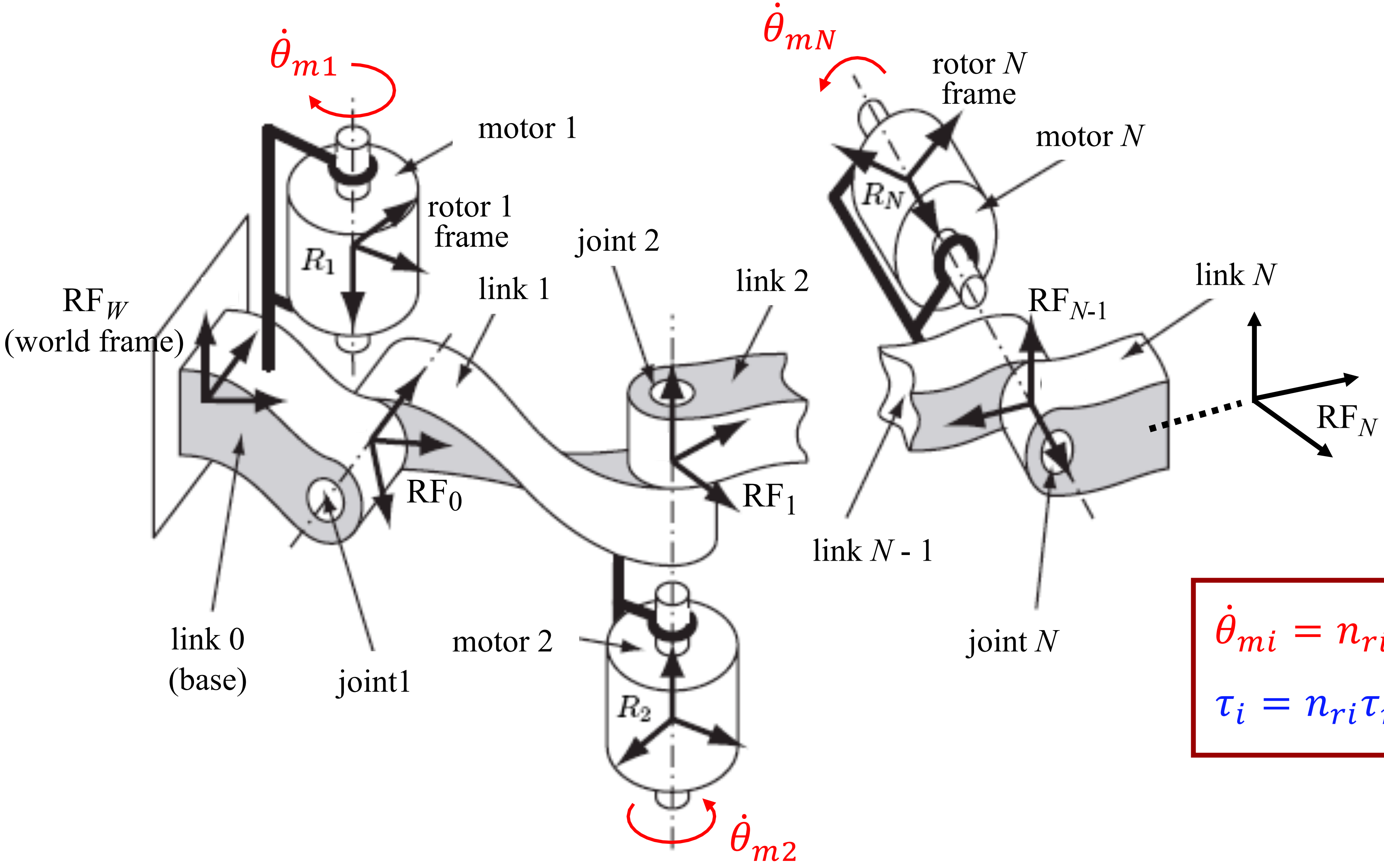
Unimation PUMA family





Mitsubishi RV-3S

Placement of motors along the chain



$$\dot{\theta}_{mi} = n_{ri}\dot{\theta}_i$$

$$\tau_i = n_{ri}\tau_{mi}$$

Resulting dynamic model

- simplifying assumption: in the rotational part of the kinetic energy, only the "spinning" rotor velocity is considered

$$T_{mi} = \frac{1}{2} I_{mi}\dot{\theta}_{mi}^2 = \frac{1}{2} I_{mi}n_{ri}^2\dot{q}_i^2 = \frac{1}{2} B_{mi}\dot{q}_i^2 \qquad T_m = \sum_{i=1}^{N} T_{mi} = \frac{1}{2}\dot{q}^T \underset{\substack{\uparrow \\ \text{diagonal}, > 0}}{B_m}\dot{q}$$

- including all added terms, the robot dynamics becomes

$$(M(q) + \underset{\substack{\downarrow \\ \text{constant}}}{B_m})\ddot{q} + c(\underset{\substack{\uparrow \\ \text{does NOT} \\ \text{contribute to } c}}{q},\dot{q}) + g(q) + \overbrace{\underbrace{F_V\dot{q} + F_C\,\mathrm{sgn}(\dot{q})}_{\substack{F_V > 0,\, F_C > 0 \\ \text{diagonal}}}}^{\text{moved to the left ...}} = \tau$$

constant → does NOT contribute to c

motor torques (**after** reduction gears)

- scaling by the reduction gears, looking from the motor side

diagonal

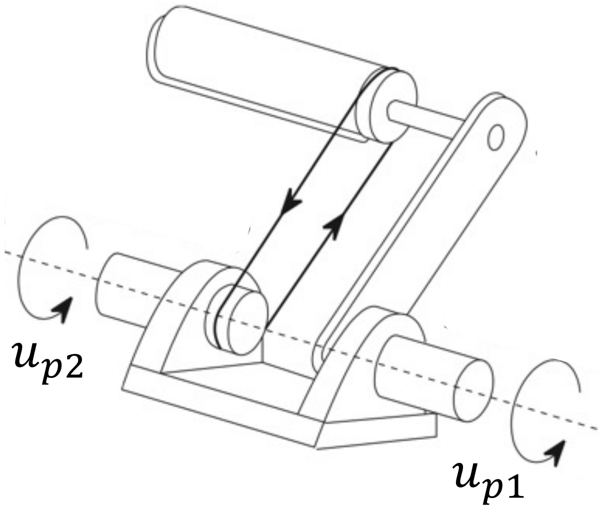
$$\left(I_m + \mathrm{diag}\left\{\frac{m_{ii}(q)}{n_{ri}^2}\right\}\right)\ddot{\theta}_m + \mathrm{diag}\left\{\frac{1}{n_{ri}}\right\}\left(\sum_{j=1}^{N} \bar{M}_j(q)\ddot{q}_j + f(q,\dot{q})\right) = \tau_m$$

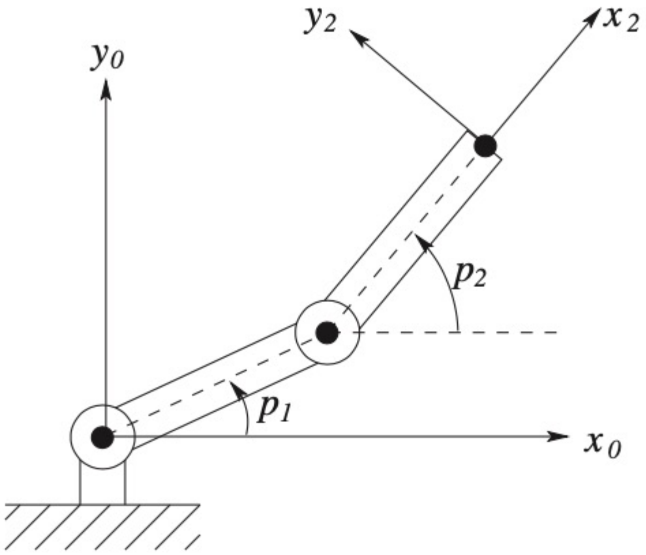
except the terms m_{jj}

motor torques (**before** reduction gears)

Special actuation and associated coordinates

planar 2R robot with remotely driven forearm





- motor 1 moves link 1 by p_1
- motor 2 at the base moves the absolute angle p_2 of link 2
- derive the dynamic model from scratch using the $\boldsymbol{p}$ coordinates

$$M(p)\ddot{p} + c(p,\dot{p}) + g(p) = u_p$$

$$M(p) = \begin{pmatrix} a_1 - a_3 & a_2 c_{2-1} \\ a_2 c_{2-1} & a_3 \end{pmatrix}$$

$$c(p,\dot{p}) = \begin{pmatrix} -a_2 s_{2-1}\, \dot{p}_2^2 \\ a_2 s_{2-1}\, \dot{p}_1^2 \end{pmatrix}$$

no more Coriolis forces!

$$g(p) = \begin{pmatrix} a_4 c_1 \\ a_5 c_2 \end{pmatrix}$$

$c_1 = \cos p_1 \quad c_2 = \cos p_2$

$c_{2-1} = \cos(p_2 - p_1) \quad s_{2-1} = \sin(p_2 - p_1)$

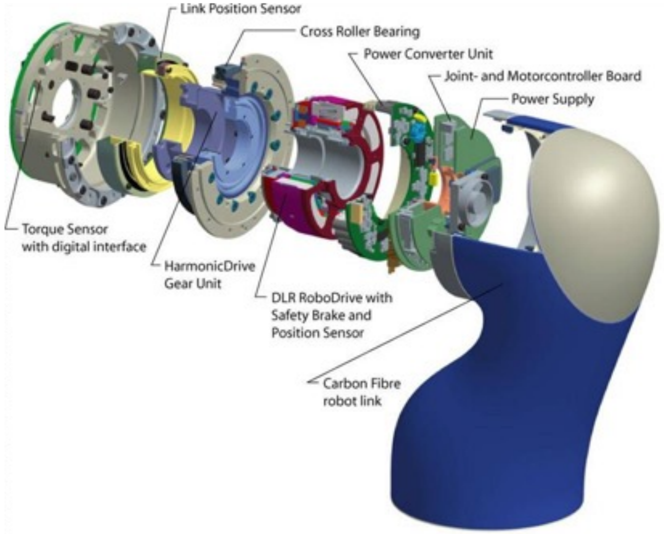
Including joint elasticity

- in industrial robots, use of motion transmissions based on
  - belts
  - harmonic drives
  - long shafts

  introduces flexibility between actuating motors (input) and driven links (output)
- in research robots, compliance in transmissions is introduced on purpose for safety (human collaboration) and/or energy efficiency
  - actuator relocation by means of (compliant) cables and pulleys
  - harmonic drives and lightweight (but rigid) link design
  - redundant (macro-mini or parallel) actuation, with elastic couplings
- in both cases, flexibility is modeled as concentrated at the joints
- in most cases, assuming small joint deformation (elastic domain)

Robots with joint elasticity

Dexter
with cable transmissions



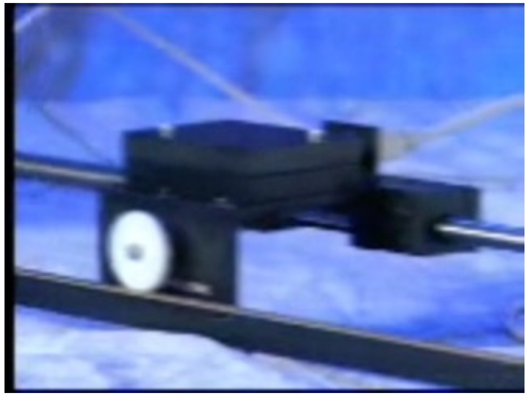
DLR LWR-III
with harmonic drives

motor

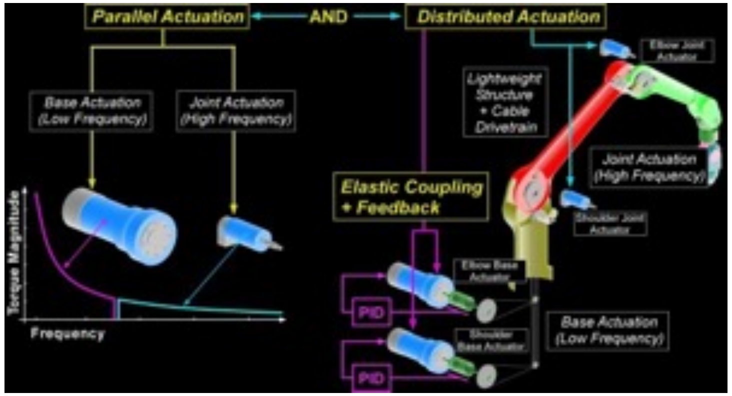
elastic spring (stiffness K)

load/link

Quanser Flexible Joint
(1-dof linear, educational)



video



Stanford DECMMA
with micro-macro actuation

Dynamic model of robots with elastic joints

- introduce $2N$ generalized coordinates
  - $q = N$ link positions
  - $\theta = N$ motor positions (after reduction, $\theta_i = \theta_{mi}/n_{ri}$)
- add motor kinetic energy T_m to that of the links $T_q = \frac{1}{2}\dot{q}^T M(q)\dot{q}$

$$T_{mi} = \frac{1}{2} I_{mi}\dot{\theta}_{mi}^2 = \frac{1}{2} I_{mi} n_{ri}^2 \dot{\theta}_i^2 = \frac{1}{2} B_{mi}\dot{\theta}_i^2 \qquad T_m = \sum_{i=1}^{N} T_{mi} = \frac{1}{2}\dot{\theta}^T \underset{\text{diagonal, } > 0}{B_m} \dot{\theta}$$

- add elastic potential energy U_e to that due to gravity $U_g(q)$
  - $K =$ matrix of joint stiffness (diagonal, > 0)

$$U_{ei} = \frac{1}{2} K_i \left(q_i - \left(\frac{\theta_{mi}}{n_{ri}} \right) \right)^2 = \frac{1}{2} K_i (q_i - \theta_i)^2 \quad U_e = \sum_{i=1}^{N} U_{ei} = \frac{1}{2} (q - \theta)^T K (q - \theta)$$

- apply Euler-Lagrange equations w.r.t. (q, θ)

$2N$ 2^{nd}-order differential equations

$$\begin{cases} M(q)\ddot{q} + c(q,\dot{q}) + g(q) + K(q-\theta) = 0 \leftarrow \text{no external torques performing work on } q \\ B_m\ddot{\theta} + K(\theta - q) = \tau \end{cases}$$

Use of the dynamic model

inverse dynamics

- given a desired trajectory $q_d(t)$
  - twice differentiable ($\exists\ \ddot{q}_d(t)$)
  - possibly obtained from a task/Cartesian trajectory $r_d(t)$, by (differential) kinematic inversion

the input torque needed to execute this motion (in free space) is

$$\tau_d = (M(q_d) + B_m)\ddot{q}_d + c(q_d, \dot{q}_d) + g(q_d) + F_V \dot{q}_d + F_C\,\mathrm{sgn}(\dot{q}_d)$$

(in contact, with an external wrench) $\ldots - J_{ext}^T(q_d) F_{ext,d}$

- useful also for control (e.g., nominal feedforward)
- however, this way of performing the algebraic computation ($\forall t$) is not efficient when using the Lagrangian modeling approach
  - symbolic terms grow much longer, quite rapidly for larger N
  - in real time, numerical computation is based on Newton-Euler method

State equations

direct dynamics

Lagrangian dynamic model

$$M(q)\ddot{q} + c(q,\dot{q}) + g(q) = u$$

N differential
2^{nd} order
equations

defining the vector of state variables as $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} q \\ \dot{q} \end{pmatrix} \in \mathbb{R}^{2N}$

state equations

$$\boxed{\dot{x} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_2 \\ -M^{-1}(x_1)[c(x_1,x_2) + g(x_1)] \end{pmatrix} + \begin{pmatrix} 0 \\ M^{-1}(x_1) \end{pmatrix} u}$$

$$= \underset{2N \times 1}{f(x)} + \underset{2N \times N}{G(x)}u$$

$2N$ differential
1^{st} order
equations

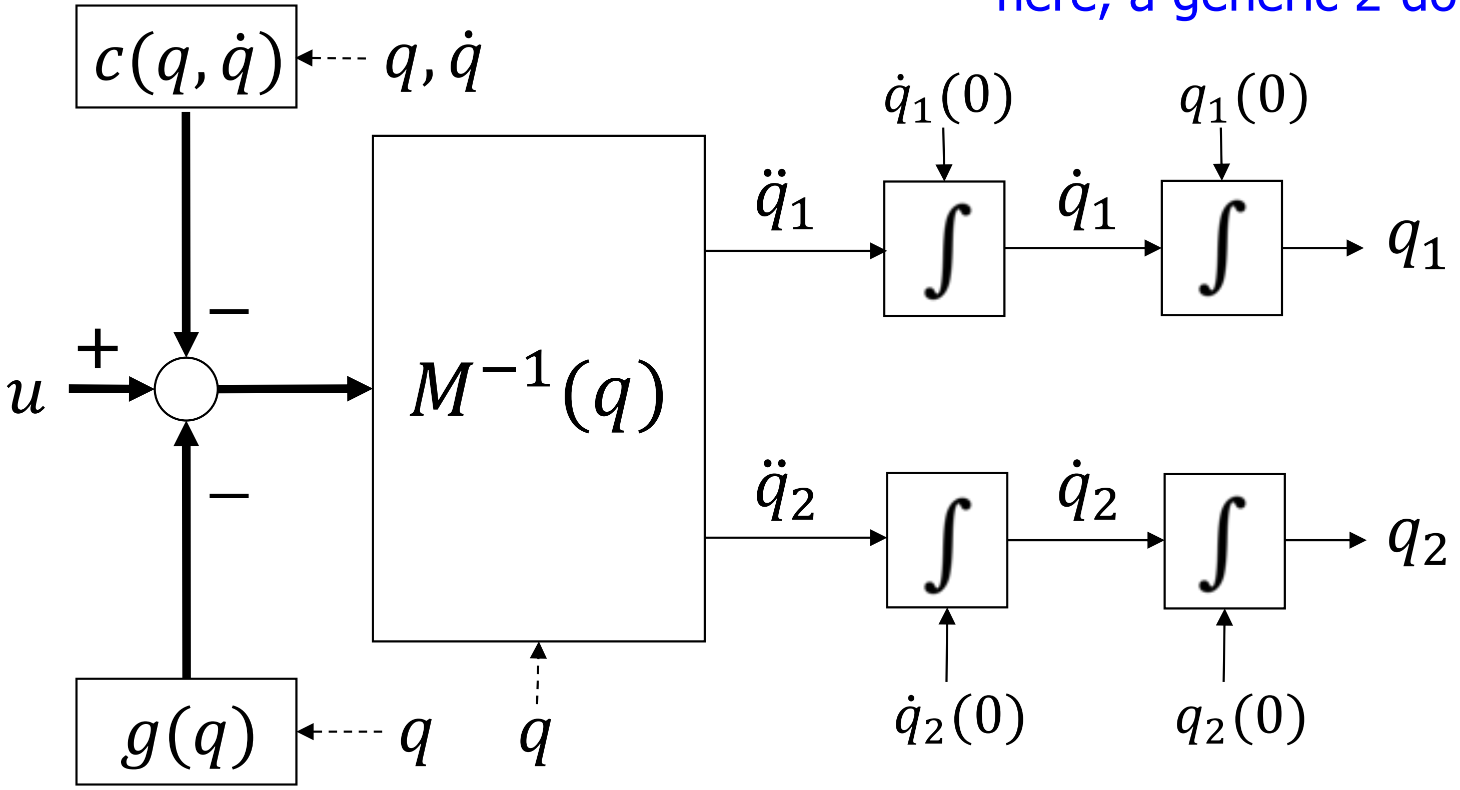
another choice... $\boxed{\tilde{x} = \begin{pmatrix} q \\ M(q)\dot{q} \end{pmatrix}}$ $\longleftarrow$ generalized momentum

$\dot{\tilde{x}} = \ldots$ (do it as exercise)

Dynamic simulation

Simulink block scheme

here, a generic 2-dof robot



input torque command (open-loop or in feedback)

including "inv(M)"

- initialization (dynamic coefficients and initial state)
- calls to (user-defined) Matlab functions for the evaluation of model terms
- choice of a numerical integration method (and of its parameters)

e.g., 4th-order Runge-Kutta (ode45)

Approximate linearization

- we can derive a linear dynamic model of the robot, which is valid locally around a given operative condition
  - useful for analysis, design, and gain tuning of linear (or of the linear part of) control laws
  - approximation by Taylor series expansion, up to the first order
  - linearization around a (constant) equilibrium state or along a (nominal, time-varying) equilibrium trajectory
  - usually, we work with (nonlinear) state equations; for mechanical systems, it is more convenient to directly use the 2nd order model
    - same result, but easier derivation

equilibrium state $(q, \dot{q}) = (q_e, 0)\ [\, \ddot{q} = 0 \,]$ ➡ $g(q_e) = u_e$

equilibrium trajectory $(q, \dot{q}) = (q_d(t), \dot{q}_d(t))\ \ [\, \ddot{q} = \ddot{q}_d(t) \,]$

➡ $M(q_d)\ddot{q}_d + c(q_d, \dot{q}_d) + g(q_d) = u_d$

Linearization at an equilibrium state

- variations around an equilibrium state

$$q = q_e + \Delta q \quad \dot{q} = \cancel{\dot{q}_e} + \dot{\Delta q} = \dot{\Delta q} \quad \ddot{q} = \cancel{\ddot{q}_e} + \ddot{\Delta q} = \ddot{\Delta q} \quad u = u_e + \Delta u$$

- keeping into account the quadratic dependence of c terms on velocity (thus, neglected around the zero velocity)

$$M(q_e)\ddot{\Delta q} + \cancel{g(q_e)} + \underbrace{\left.\frac{\partial g}{\partial q}\right|_{q=q_e}}_{G(q_e)} \Delta q + \cancel{\mathrm{o}\left(\|\Delta q\|, \|\dot{\Delta q}\|\right)} = \cancel{u_e} + \Delta u$$

infinitesimal terms
of second or higher order

- in state-space format, with $\Delta x = \begin{pmatrix} \Delta q \\ \dot{\Delta q} \end{pmatrix}$

$$\dot{\Delta x} = \begin{pmatrix} 0 & I \\ -M^{-1}(q_e)G(q_e) & 0 \end{pmatrix} \Delta x + \begin{pmatrix} 0 \\ M^{-1}(q_e) \end{pmatrix} \Delta u = A\,\Delta x + B\,\Delta u$$

Linearization along a trajectory

- variations around an equilibrium trajectory

$$q = q_d + \Delta q \qquad \dot{q} = \dot{q}_d + \dot{\Delta q} \qquad \ddot{q} = \ddot{q}_d + \ddot{\Delta q} \qquad u = u_d + \Delta u$$

- developing to 1st order the terms in the dynamic model ...

$$M(q_d + \Delta q)\left(\ddot{q}_d + \ddot{\Delta q}\right) + c(q_d + \Delta q, \dot{q}_d + \dot{\Delta q}) + g(q_d + \Delta q) = u_d + \Delta u$$

$$M(q_d + \Delta q) \cong M(q_d) + \sum_{i=1}^{N} \left.\frac{\partial M_i}{\partial q}\right|_{q=q_d} e_i^T \Delta q$$

i-th row of the identity matrix

$$g(q_d + \Delta q) \cong g(q_d) + G(q_d)\Delta q$$

$$c\left(q_d + \Delta q, \dot{q}_d + \dot{\Delta q}\right) \cong c(q_d, \dot{q}_d) + \overbrace{\left.\frac{\partial c}{\partial q}\right|_{\substack{q=q_d \\ \dot{q}=\dot{q}_d}}}^{C_1(q_d, \dot{q}_d)} \Delta q + \underbrace{\left.\frac{\partial c}{\partial \dot{q}}\right|_{\substack{q=q_d \\ \dot{q}=\dot{q}_d}}}_{C_2(q_d, \dot{q}_d)} \dot{\Delta q}$$

Linearization along a trajectory (cont)

- after simplifications …

$$M(q_d)\Delta\ddot{q} + C_2(q_d,\dot{q}_d)\Delta\dot{q} + D(q_d,\dot{q}_d,\ddot{q}_d)\Delta q = \Delta u$$

with

$$D(q_d,\dot{q}_d,\ddot{q}_d) = G(q_d) + C_1(q_d,\dot{q}_d) + \sum_{i=1}^{N} \left.\frac{\partial M_i}{\partial q}\right|_{q=q_d} \ddot{q}_d e_i^T$$

- in state-space format

$$\Delta\dot{x} = \begin{pmatrix} 0 & I \\ -M^{-1}(q_d)D(q_d,\dot{q}_d,\ddot{q}_d) & -M^{-1}(q_d)C_2(q_d,\dot{q}_d) \end{pmatrix}\Delta x + \begin{pmatrix} 0 \\ M^{-1}(q_d) \end{pmatrix}\Delta u = A(t)\,\Delta x + B(t)\,\Delta u$$

a linear, but time-varying system!!

Coordinate transformation

$q \in \mathbb{R}^N$ $\boxed{M(q)\ddot{q} + c(q,\dot{q}) + g(q) = M(q)\ddot{q} + n(q,\dot{q}) = u_q}$ ①

if we wish/need to use a new set of generalized coordinates p

$p \in \mathbb{R}^N$ $\boxed{p = f(q)}$ $\rightarrow$ $\boxed{q = f^{-1}(p)}$

by duality
(principle of virtual work)

$\boxed{\dot{p} = \frac{\partial f}{\partial q}\dot{q} = J(q)\dot{q}}$ $\rightarrow$ $\boxed{\dot{q} = J^{-1}(q)\dot{p}}$ $\boxed{u_q = J^T(q)u_p}$

$\boxed{\ddot{p} = J(q)\ddot{q} + \dot{J}(q)\dot{q}}$ $\rightarrow$ $\boxed{\ddot{q} = J^{-1}(q)\left(\ddot{p} - \dot{J}(q)J^{-1}(q)\dot{p}\right)}$

$\rightarrow$ ①

$$\boxed{M(q)J^{-1}(q)\ddot{p} - M(q)J^{-1}(q)\dot{J}(q)J^{-1}(q)\dot{p} + n(q,\dot{q}) = J^T(q)u_p}$$

$J^{-T}(q)\cdot$ pre-multiplying the whole equation...

Robot dynamic model

after coordinate transformation

$$\boxed{J^{-T}(q)M(q)J^{-1}(q)\ddot{p} + J^{-T}(q)\left(n(q,\dot{q}) - M(q)J^{-1}(q)\dot{J}(q)J^{-1}(q)\dot{p}\right) = u_p}$$

$$\boxed{q \rightarrow p}$$

for actual computation, these inner substitutions are not strictly necessary

$$\boxed{(q,\dot{q}) \rightarrow (p,\dot{p})}$$

non-conservative generalized forces performing work on p

$$\Rightarrow \boxed{M_p(p)\ddot{p} + c_p(p,\dot{p}) + g_p(p) = u_p}$$

$$M_p = J^{-T}MJ^{-1}$$

symmetric, positive definite (out of singularities)

$$g_p = J^{-T}g$$

$$c_p = J^{-T}\left(c - MJ^{-1}\dot{J}\,J^{-1}\dot{p}\right) = J^{-T}c - M_p\dot{J}\,J^{-1}\dot{p}$$

quadratic dependence on $\dot{p}$

$$c_p(p,\dot{p}) = S_p(p,\dot{p})\,\dot{p} \qquad \dot{M}_p - 2S_p$$

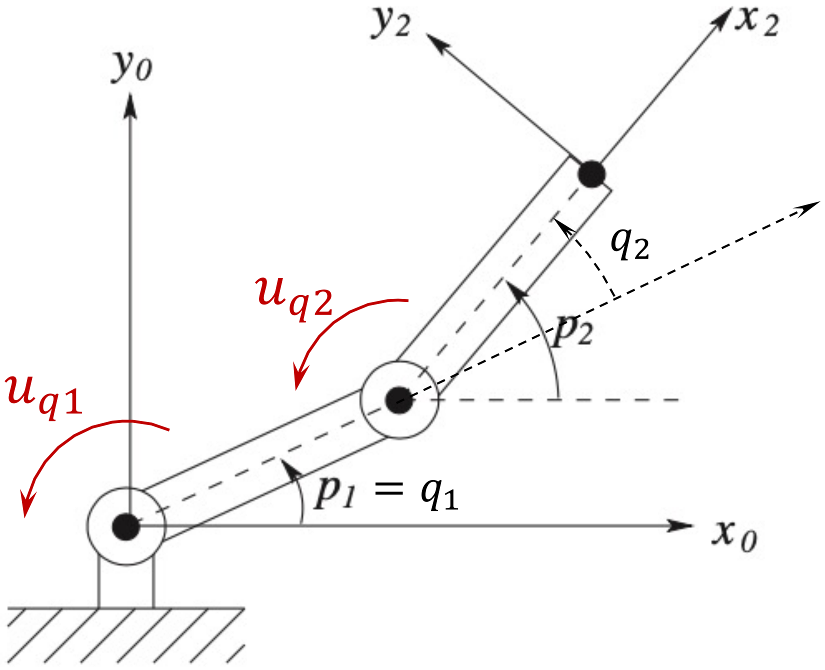
skew-symmetric

when p = E-E pose, this is the robot dynamic model in Cartesian coordinates

NOTE: in this case, we have implicitly assumed than $M = N$ (no redundancy!)

Example of coordinate transformation

planar 2R robot using absolute coordinates



- motor 1 at joint 1, motor 2 at joint 2
- in place of DH angles $\boldsymbol{q}$, use the absolute angles $p_1 = q_1$ and $p_2 = q_1 + q_2$

$$p = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} q = J\, q$$

a linear transformation

$$J^{-1} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \qquad J^{-T} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}$$

- from $M(q)\ddot{q} + c(q,\dot{q}) + g(q) = u_q$ obtained with DH relative coordinates

blue terms are the same found in a direct way in slide #15

$$M_p(p) = J^{-T} M\, J^{-1} = \begin{pmatrix} a_1 - a_3 & a_2 c_{2-1} \\ a_2 c_{2-1} & a_3 \end{pmatrix} \qquad g_p(p) = J^{-T} g = \begin{pmatrix} a_4 c_1 \\ a_5 c_2 \end{pmatrix}$$

$$c_p(p,\dot{p}) = J^{-T} c = \begin{pmatrix} -a_2 s_{2-1}\, \dot{p}_2^2 \\ a_2 s_{2-1}\, \dot{p}_1^2 \end{pmatrix} \qquad u_p = J^{-T} u_q = \begin{pmatrix} u_{q1} - u_{q2} \\ u_{q2} \end{pmatrix}$$

Robot dynamic model
in the task/Cartesian space, with redundancy

dynamic model in the joint space

$$M(q)\ddot{q} + n(q,\dot{q}) = \tau$$

$$\boxed{\begin{gathered} q \in \mathbb{R}^N \\ r = f(q) \in \mathbb{R}^M \\ M < N \end{gathered}}$$

second-order task kinematics

$$\ddot{r} = J(q)\ddot{q} + \dot{J}(q)\dot{q}$$

J is full rank = M

1) isolate the joint acceleration from the dynamics ➡ $\ddot{q} = M^{-1}(q)\left(\tau - n(q,\dot{q})\right)$

2) decompose the joint torques in two complementary spaces

$$\tau = \underset{\in \mathcal{R}(J^T)}{J^T(q)F} + \underset{\in \mathcal{N}(J^T H)}{(I - J^T(q)H(q))\tau_0}$$

H is a generalized inverse of J^T

$$J^T H J^T = J^T$$

torques coming from generalized forces F in the task space …

… and joint torques $\tau_0 \notin \mathcal{R}(J^T)$

➡ $\tau_0 = J^T(q)F, \forall F \in \mathbb{R}^M \Rightarrow \left(I - J^T(q)H(q)\right)J^T(q)F = 0$

3) substitute 1) and 2) in the differential task kinematics

$$\Rightarrow \ddot{r} = \underbrace{J(q)M^{-1}(q)\big(J^T(q)}_{}F + (I - J^T(q)H(q))\tau_0 - n(q,\dot{q})\big) + \dot{J}(q)\dot{q}$$

4) isolate on the right-hand side the generalized forces F in the task space …

Robot dynamic model

in the task/Cartesian space, with redundancy

➡ $$\left(J(q)M^{-1}(q)J^T(q)\right)^{-1}\ddot{r} = F + \left(J(q)M^{-1}(q)J^T(q)\right)^{-1}\left(J(q)M^{-1}(q)\left(\left(I - J^T(q)H(q)\right)\tau_0 - n(q,\dot{q})\right) + \dot{J}(q)\dot{q}\right)$$

5) choose as generalized inverse $H = (JM^{-1}J^T)^{-1}JM^{-1} = \left(J_M^{\#}\right)^T$, i.e., the transpose of the inertia-weighted pseudoinverse of the task Jacobian (see block of slides #2)

➡ in this way, the joint torque component τ_0 will **NOT** affect the task acceleration $\ddot{r}$

$$\left(J(q)M^{-1}(q)J^T(q)\right)^{-1}\ddot{r} = F + \left(J(q)M^{-1}(q)J^T(q)\right)^{-1}\left(\dot{J}(q)\dot{q} - J(q)M^{-1}(q)\,n(q,\dot{q})\right)$$

6) the resulting (M −dimensional) task dynamics is then

$$M_r(q)\ddot{r} + n_r(q,\dot{q}) = F \quad \ldots + F_{ext}$$

external forces can be added on the rhs of the equations in a dynamically consistent way!

with

$$M_r(q) = \left(J(q)M^{-1}(q)J^T(q)\right)^{-1} \quad \text{task inertia matrix}$$

$$n_r(q,\dot{q}) = M_r(q)\left(J(q)M^{-1}(q)\,n(q,\dot{q}) - \dot{J}(q)\dot{q}\right)$$

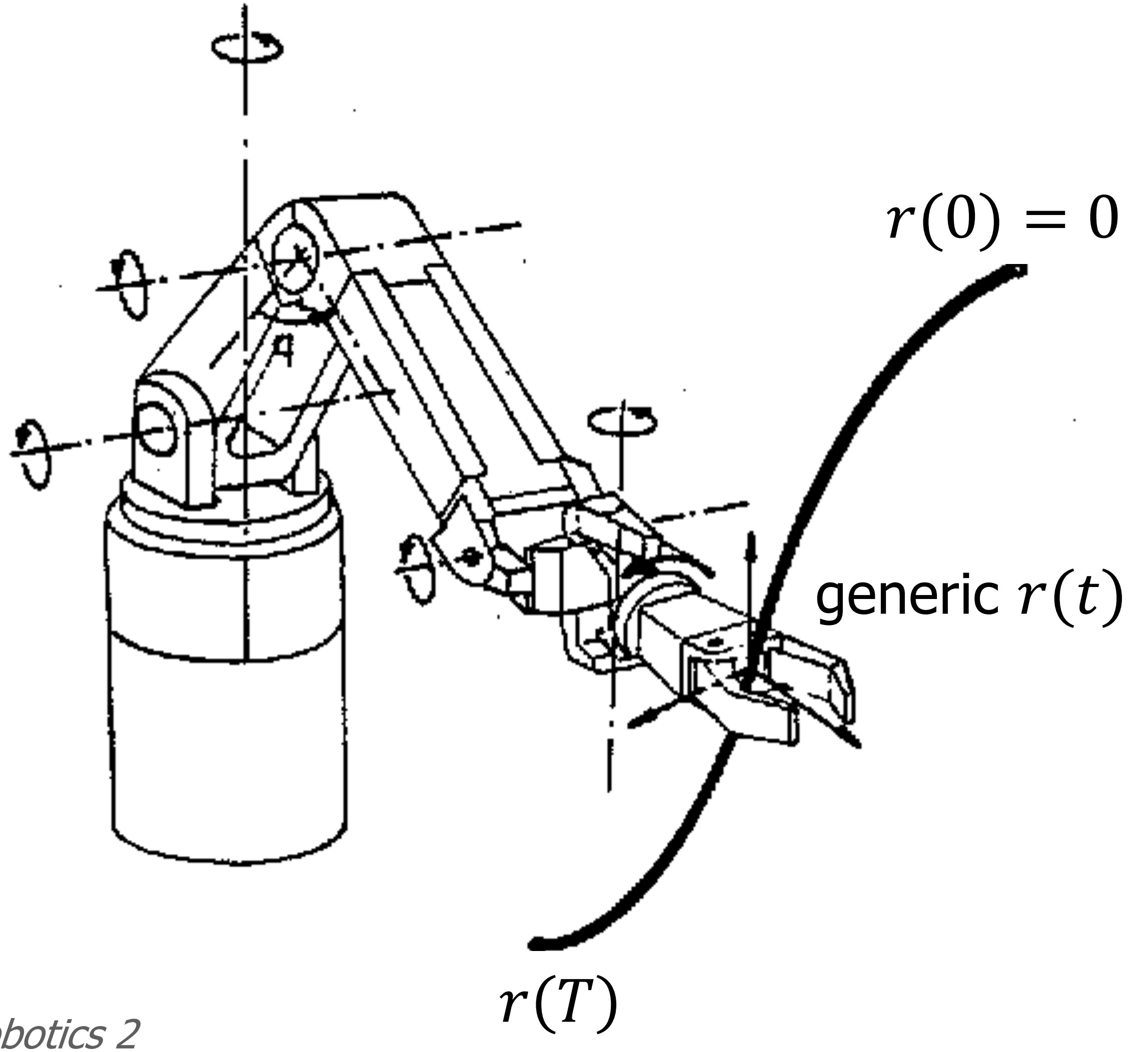
} for $M = N$, these terms are identical to slide #27

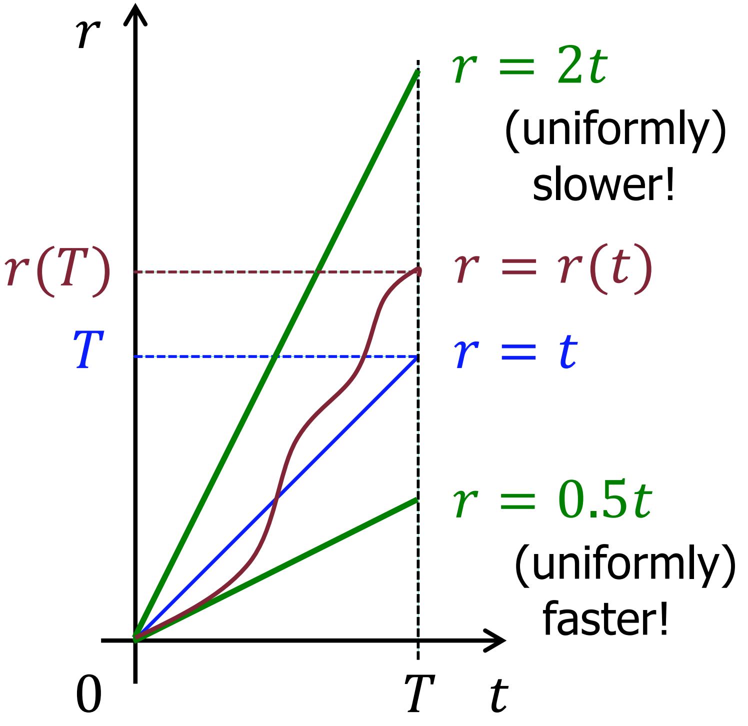
7) an additional $(N - M)$-dimensional second-order dynamics is needed to describe the full robot!

Dynamic scaling of trajectories

uniform time scaling of motion

- given a smooth original trajectory $q_d(t)$ of motion for $t \in [0, T]$
  - suppose to rescale time as $t \to r(t)$ (a strictly **increasing** function of t)





Dynamic scaling of trajectories

uniform time scaling of motion

- in the new time scale, the scaled trajectory $q_s(r)$ satisfies

$$q_d(t) = q_s(r(t)) \quad \Rightarrow \quad \dot{q}_d(t) = \frac{dq_d}{dt} = \frac{dq_s}{dr}\frac{dr}{dt} = q_s'(r)\,\dot{r}(t)$$

same path executed
(at different instants of time)

$$\Downarrow$$

$$\ddot{q}_d(t) = \frac{d\dot{q}_d}{dt} = \left(\frac{dq_s'}{dr}\frac{dr}{dt}\right)\dot{r} + q_s'\frac{d\dot{r}}{dt} = q_s''(r)\dot{r}^2(t) + q_s'(r)\ddot{r}(t)$$

- uniform scaling of the trajectory occurs when $r(t) = kt$

$$\dot{q}_d(t) = kq_s'(kt) \qquad \ddot{q}_d(t) = k^2 q_s''(kt)$$

Q: what is the new input torque needed to execute the scaled trajectory?
(suppose dissipative terms can be neglected)

Dynamic scaling of trajectories

inverse dynamics under uniform time scaling

- the new torque could be recomputed through the inverse dynamics, for every $r = kt \in [0, T_s] = [0, kT]$ along the scaled trajectory, as

$$\tau_s(kt) = M(q_s)q_s'' + c(q_s, q_s') + g(q_s)$$

- however, being the dynamic model linear in the acceleration and quadratic in the velocity, it is

$$\begin{aligned}\tau_d(t) &= M(q_d)\enclose{circle}{\ddot{q}_d} + c(q_d, \enclose{circle}{\dot{q}_d}) + g(q_d) = M(q_s)k^2 q_s'' + c(q_s, kq_s') + g(q_s)\\ &= k^2\big(M(q_s)q_s'' + c(q_s, q_s')\big) + g(q_s) = k^2\big(\tau_s(kt) - g(q_s)\big) + g(q_s)\end{aligned}$$

- thus, saving separately the total torque $\tau_d(t)$ and gravity torque $g_d(t)$ in the inverse dynamics computation along the original trajectory, the new input torque is obtained directly as

$$\boxed{\tau_s(kt) = \frac{1}{k^2}\big(\tau_d(t) - g(q_d(t))\big) + g(q_d(t))}$$

$k > 1$: slow down
⇒ reduce torque

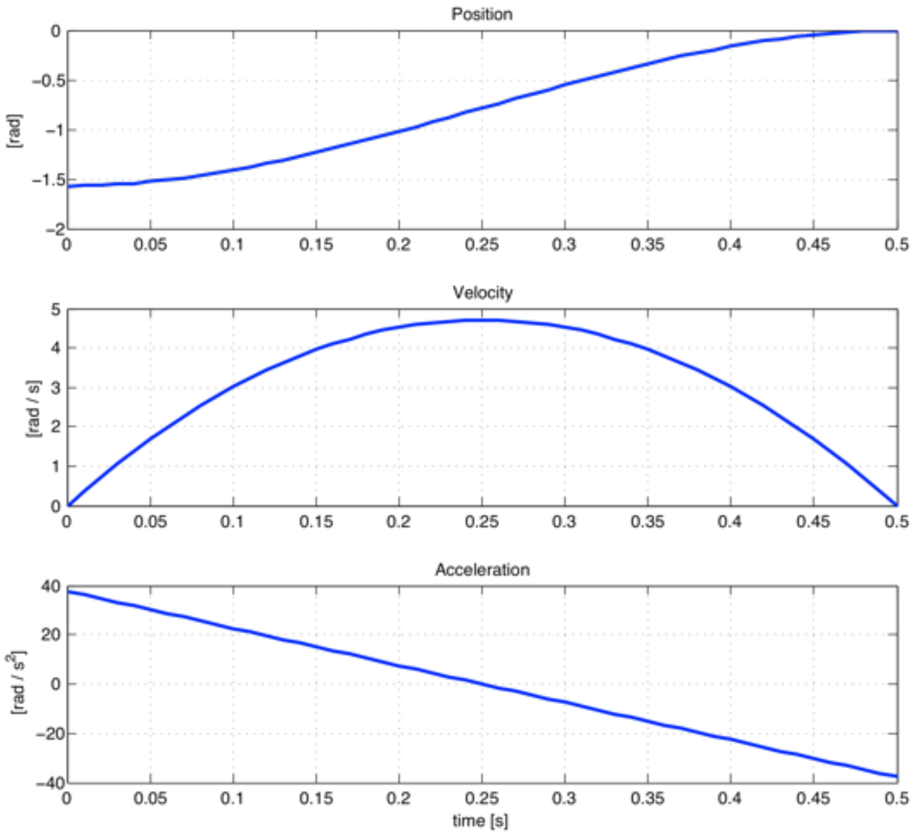
$k < 1$: speed up
⇒ increase torque

gravity term (only position-dependent): does NOT scale!

Dynamic scaling of trajectories

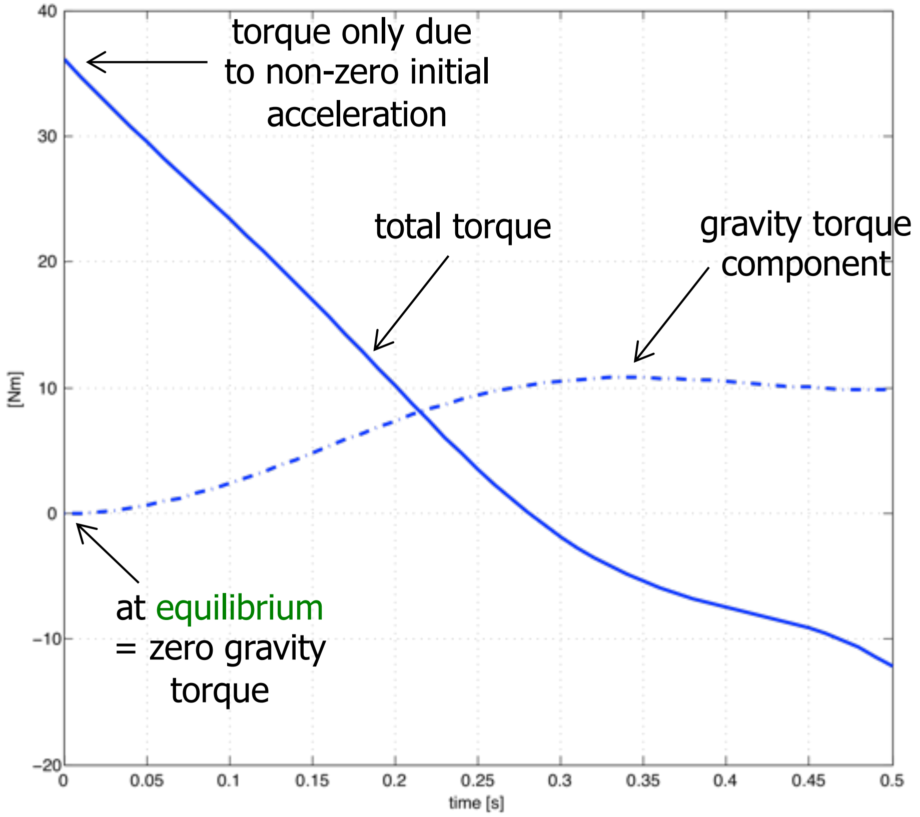
numerical example

- rest-to-rest motion with cubic polynomials for planar 2R robot under gravity (from downward equilibrium to horizontal link 1 & upward vertical link 2)
- original trajectory lasts $T = 0.5$ s (but say, it violates the torque limit at joint 1)



for both joints

only joint 1 torque is shown



torque only due to non-zero initial acceleration

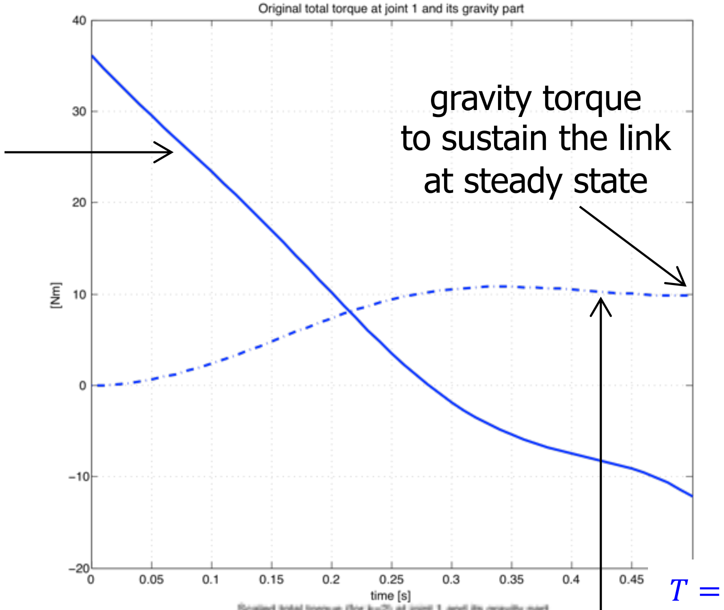
total torque

gravity torque component

at equilibrium = zero gravity torque

Dynamic scaling of trajectories

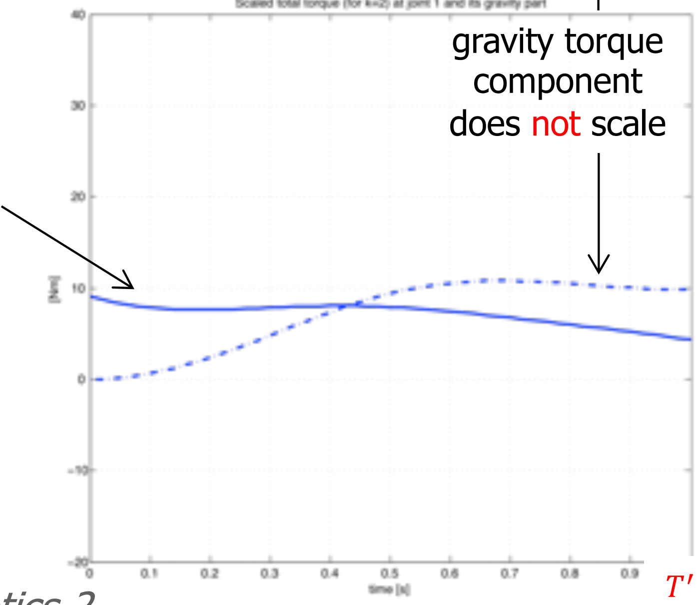
numerical example



original total torque

gravity torque to sustain the link at steady state

$T = 0.5$ s

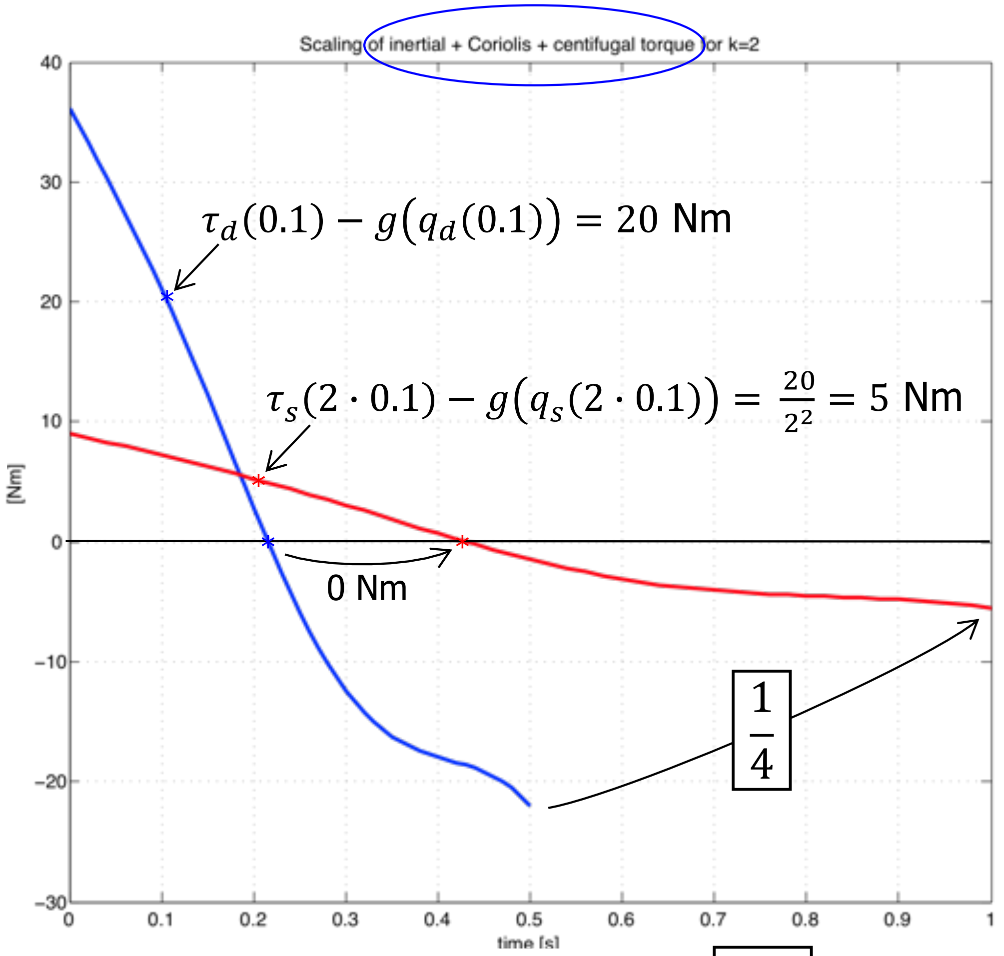


gravity torque component does not scale

scaled total torque

$T' = 1$ s

- scaling with $k = 2$ (slower) → $T' = 1$ s



$\tau_d(0.1) - g\big(q_d(0.1)\big) = 20$ Nm

$\tau_s(2 \cdot 0.1) - g\big(q_s(2 \cdot 0.1)\big) = \frac{20}{2^2} = 5$ Nm

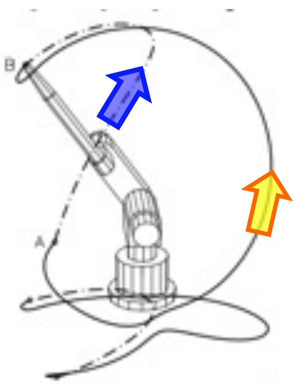
0 Nm

$\boxed{\frac{1}{4}}$

$$T = 0.5 \text{ s} \longrightarrow \boxed{k = 2} \longrightarrow T = 1 \text{ s}$$

Optimal point-to-point robot motion

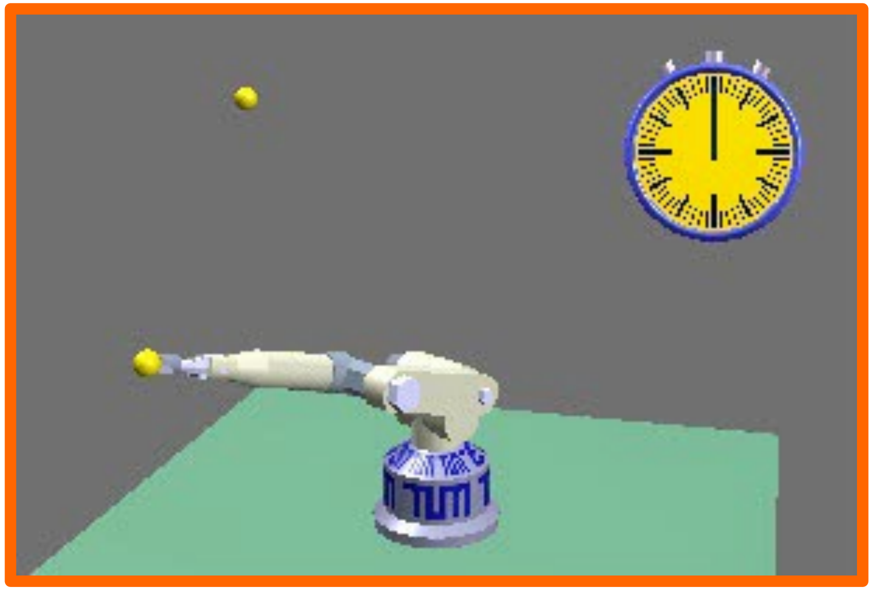
considering the dynamic model



- given the initial (⇒ A) and final (⇒ B) robot configurations (at rest) and the actuator torque bounds, find
  - the minimum-time T_{min} motion
  - the (global/integral) minimum-energy E_{min} motion

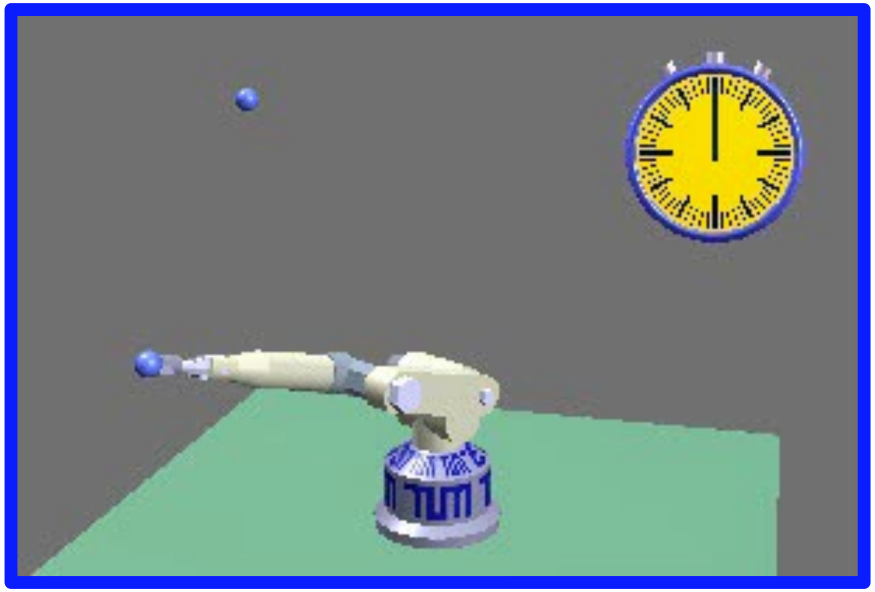
  and the associated command torques needed to execute them
- a complex nonlinear optimization problem solved numerically

video



T_{min}= 1.32 s, E = 306

video



T = 1.60 s, E_{min} = 6.14