ETH zürich

Lecture «Robot Dynamics»: Dynamics 2

151-0851-00 V

lecture:	CAB G11	Tuesday 10:15 – 12:00, every week
exercise:	HG E1.2	Wednesday 8:15 – 10:00, according to schedule (about every 2nd week)
office hour:	LEE H303	Friday 12.15 – 13.00

Marco Hutter, Roland Siegwart, and Thomas Stastny

ETH zürich

	Topic		Title
20.09.2016	Intro and Outline	L1	Course Introduction; Recapitulation Position, Linear Velocity, Transformation
27.09.2016	Kinematics 1	L2	Rotation Representation; Introduction to Multi-body Kinematics
28.09.2016	Exercise 1a	E1a	Kinematics Modeling the ABB arm
04.10.2016	Kinematics 2	L3	Kinematics of Systems of Bodies; Jacobians
05.10.2016	Exercise 1b	L3	Differential Kinematics and Jacobians of the ABB Arm
11.10.2016	Kinematics 3	L4	Kinematic Control Methods: Inverse Differential Kinematics, Inverse Kinematics; Rotation Error; Multi-task Control
12.10.2016	Exercise 1c	E1b	Kinematic Control of the ABB Arm
18.10.2016	Dynamics L1	L5	Multi-body Dynamics
19.10.2016	Exercise 2a	E2a	Dynamic Modeling of the ABB Arm
25.10.2016	Dynamics L2	L6	Dynamic Model Based Control Methods
26.10.2016	Exercise 2b	E2b	Dynamic Control Methods Applied to the ABB arm
01.11.2016	Legged Robots	L7	Case Study and Application of Control Methods
08.11.2016	Rotorcraft 1	L8	Dynamic Modeling of Rotorcraft I
15.11.2016	Rotorcraft 2	L9	Dynamic Modeling of Rotorcraft II & Control
16.11.2016	Exercise 3	E3	Modeling and Control of Multicopter
22.11.2016	Case Studies 2	L10	Rotor Craft Case Study
29.11.2016	Fixed-wing 1	L11	Flight Dynamics; Basics of Aerodynamics; Modeling of Fixed-wing Aircraft
30.11.2016	Exercise 4	E4	Aircraft Aerodynamics / Flight performance / Model derivation
06.12.2016	Fixed-wing 2	L12	Stability, Control and Derivation of a Dynamic Model
07.12.2016	Exercise 5	E5	Fixed-wing Control and Simulation
13.12.2016	Case Studies 3	L13	Fixed-wing Case Study
20.12.2016	Summery and Outlook	L14	Summery; Wrap-up; Exam

ETH zürich

Recapitulation

- We learned how to get the equation of motion in joint space
  - Newton-Euler
  - Projected Newton-Euler
  - Lagrange II
- Introduction to floating base systems

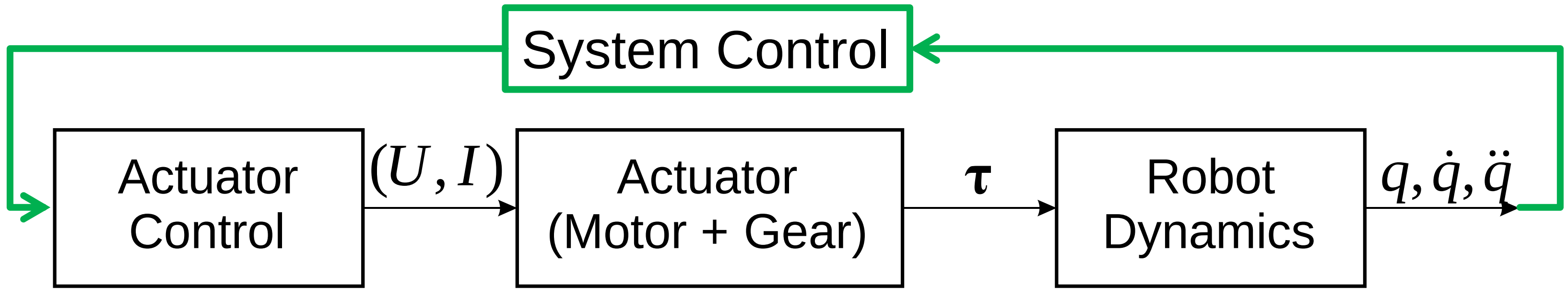
- Today:
  - How can we use this information in order to control the robot

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T\mathbf{F}_c = \boldsymbol{\tau}$$

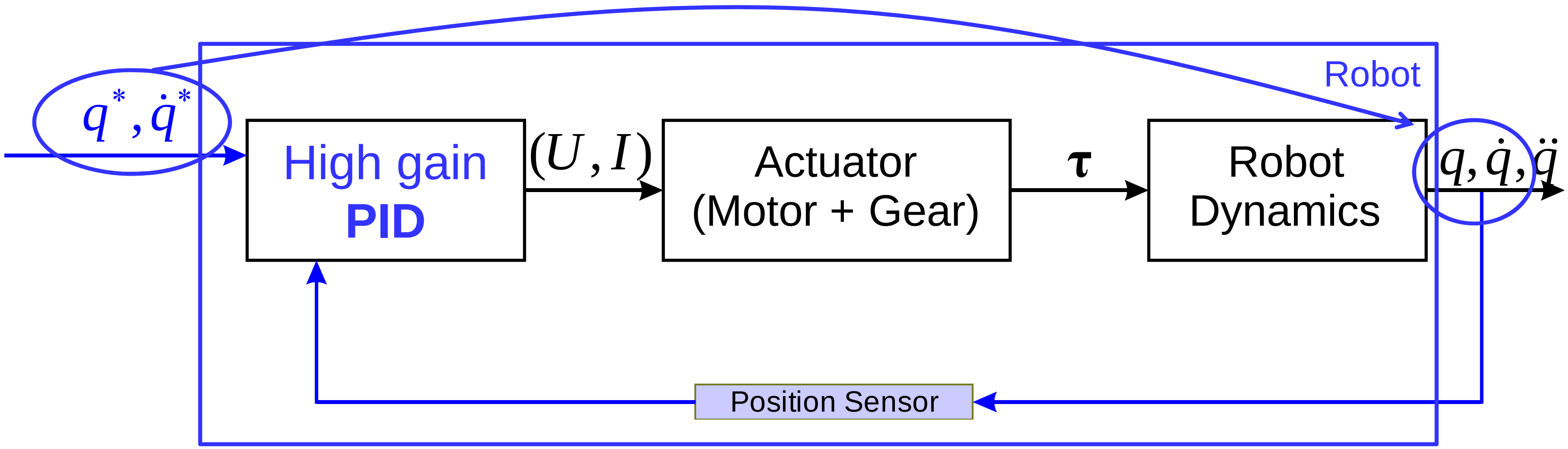
$\ddot{\mathbf{q}}$	Generalized accelerations
$\mathbf{M}(\mathbf{q})$	Mass matrix
$\mathbf{b}(\mathbf{q},\dot{\mathbf{q}})$	Centrifugal and Coriolis forces
$\mathbf{g}(\mathbf{q})$	Gravity forces
$\boldsymbol{\tau}$	Generalized forces
$\mathbf{F}_c$	External forces
$\mathbf{J}_c$	Contact Jacobian

Position vs. Torque Controlled Robot Arms

Setup of a Robot Arm



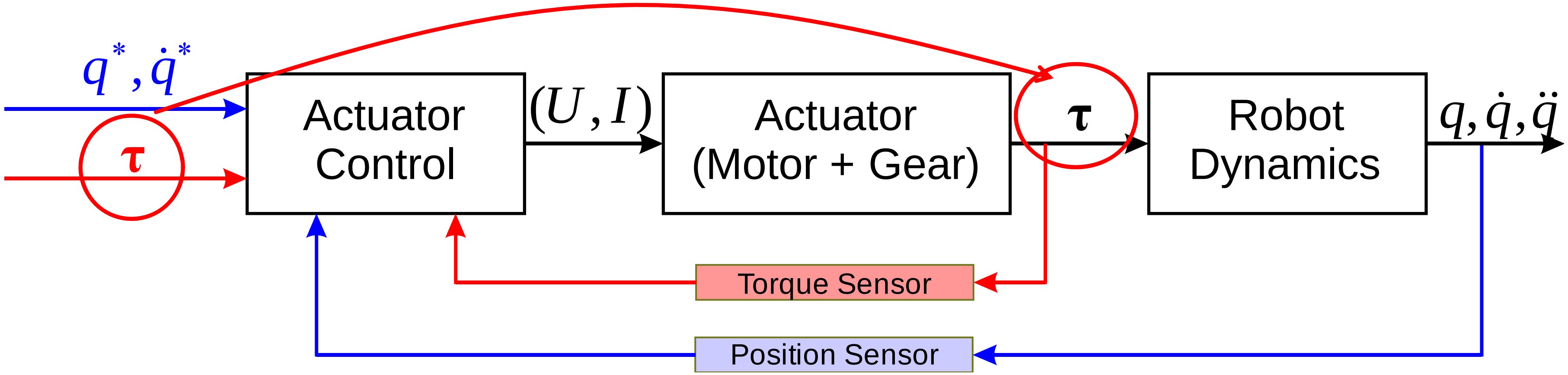
Classical Position Control of a Robot Arm



- Position feedback loop on joint level
  - Classical, position controlled robots don't care about dynamics
  - High-gain PID guarantees good joint level tracking
  - Disturbances (load, etc) are compensated by PID
  - => interaction force can only be controlled with compliant surface

ETH zürich

Joint Torque Control of a Robot Arm



- Integrate force-feedback
  - Active regulation of system dynamics
  - Model-based load compensation
  - Interaction force control

Setup of Modern Robot Arms

- Modern robots have force sensors
  - Dynamic control
  - Interaction control
  - Safety for collaboration

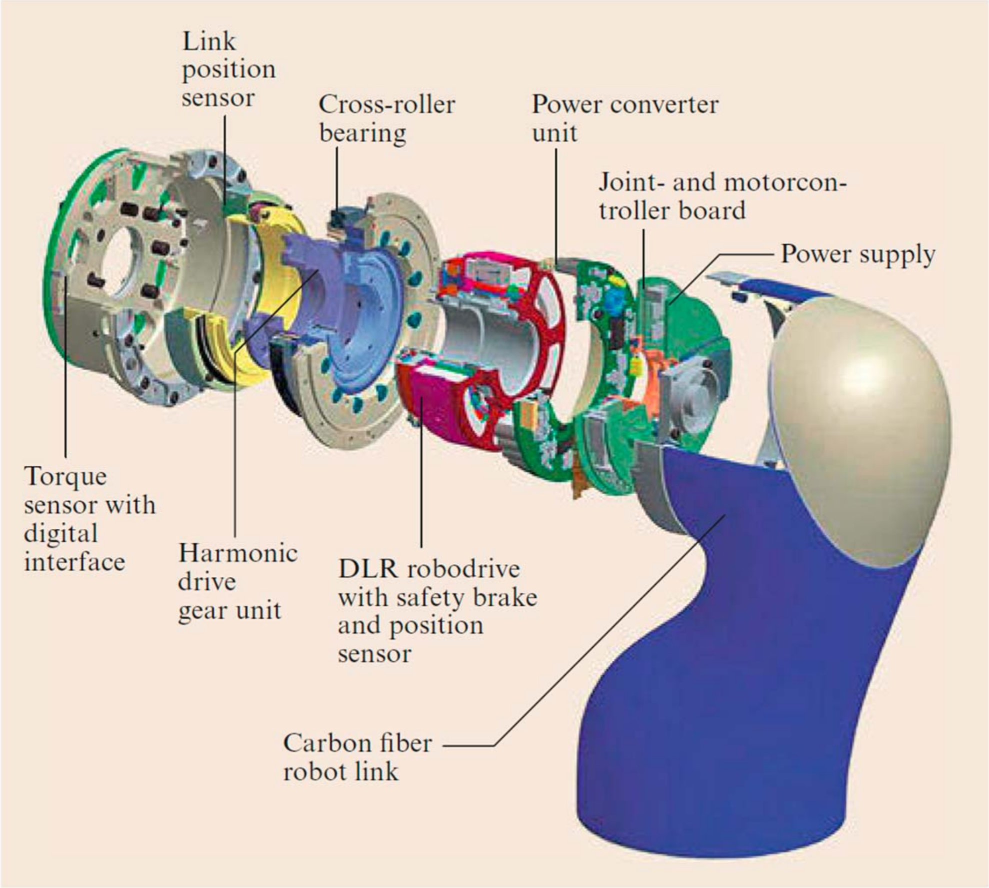


Fig. 11.8 Exploded view of a joint of the *DLR LWR-III* lightweight manipulator and its sensor suite

ANYpulator
An example for a robot that can interact

- Special force controllable actuators
  - Dynamic motion
  - Safe interaction

Joint Impedance Control

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}$$

- Torque as function of position and velocity error $\boldsymbol{\tau}^* = \mathbf{k}_p(\mathbf{q}^* - \mathbf{q}) + \mathbf{k}_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}})$

- Closed loop behavior

$$\cancel{\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}} + \cancel{\mathbf{b}(\mathbf{q},\dot{\mathbf{q}})} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} = \mathbf{k}_p(\mathbf{q}^* - \mathbf{q}) + \mathbf{k}_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}})$$

  - ➢ Static offset due to gravity

- Impedance control and gravity compensation

$$\boldsymbol{\tau}^* = \mathbf{k}_p(\mathbf{q}^* - \mathbf{q}) + \mathbf{k}_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}}) + \hat{\mathbf{g}}(\mathbf{q})$$

Estimated gravity term

Simple setup…
but configuration dependent load

Inverse Dynamics Control

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}$$

- Compensate for system dynamics $\boldsymbol{\tau} = \hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}}^* + \hat{\mathbf{b}}(\mathbf{q},\dot{\mathbf{q}}) + \hat{\mathbf{g}}(\mathbf{q})$

- In case of no modeling errors,
  - the desired dynamics can be perfectly prescribed $\mathbb{I}\ddot{\mathbf{q}} = \ddot{\mathbf{q}}^*$

- PD-control law $\mathbb{I}\ddot{\mathbf{q}} = \ddot{\mathbf{q}}^* = \mathbf{k}_p(\mathbf{q}^* - \mathbf{q}) + \mathbf{k}_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}})$

  - Every joint behaves like a decoupled mass-spring-damper with unitary mass

$$\omega = \sqrt{k_p} \qquad D = \frac{k_d}{2\sqrt{k_p}}$$

Can achieve great performance…
but requires accurate modeling

Inverse Dynamics Control with Multiple Tasks

Motion in joint space is often hard to describe => use task space

- A single task can be written as $\dot{\mathbf{w}}_e = \begin{pmatrix} \ddot{\mathbf{r}} \\ \dot{\boldsymbol{\omega}} \end{pmatrix}_e = \mathbf{J}_e \ddot{\mathbf{q}} + \dot{\mathbf{J}}_e \dot{\mathbf{q}}$
- In complex machines, we want to fulfill multiple tasks
- (As introduced already in for velocity control)
  - Same priority, multi-task inversion $\ddot{\mathbf{q}} = \begin{bmatrix} \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_{n_t} \end{bmatrix}^+ \left(\begin{pmatrix} \dot{\mathbf{w}}_1 \\ \vdots \\ \dot{\mathbf{w}}_{n_t} \end{pmatrix} - \begin{bmatrix} \dot{\mathbf{J}}_1 \\ \vdots \\ \dot{\mathbf{J}}_{n_t} \end{bmatrix} \dot{\mathbf{q}} \right)$
  - Hierarchical $\ddot{\mathbf{q}} = \sum_{i=1}^{n_T} \mathbf{N}_i \ddot{\mathbf{q}}_i, \quad \text{with} \quad \ddot{\mathbf{q}}_i = (\mathbf{J}_i \mathbf{N}_i)^+ \left(\mathbf{w}_i^* - \dot{\mathbf{J}}_i \dot{\mathbf{q}} - \mathbf{J} \sum_{k=1}^{i-1} \mathbf{N}_k \ddot{\mathbf{q}}_k \right)$

Task Space Dynamics

- Joint-space dynamics

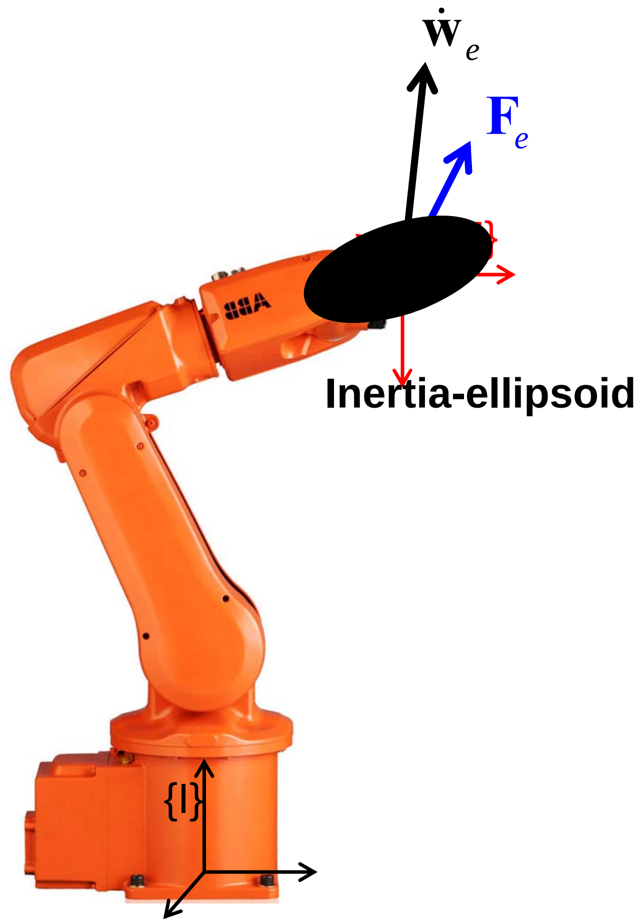
$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}$$

End-effector dynamics

$$\boldsymbol{\Lambda}\dot{\mathbf{w}}_e + \boldsymbol{\mu} + \mathbf{p} = \mathbf{F}_e$$

  - Torque to force mapping $\boldsymbol{\tau} = \mathbf{J}_e^T \mathbf{F}_e$
  - Kinematic relation $\dot{\mathbf{w}}_e = \begin{pmatrix} \ddot{\mathbf{r}} \\ \boldsymbol{\omega} \end{pmatrix}_e = \mathbf{J}_e \ddot{\mathbf{q}} + \dot{\mathbf{J}}_e \dot{\mathbf{q}}$
  - Substitute acceleration $\dot{\mathbf{w}}_e = \mathbf{J}_e \mathbf{M}^{-1}(\boldsymbol{\tau} - \mathbf{b} - \mathbf{g}) + \dot{\mathbf{J}}_e \dot{\mathbf{q}}$

$$\Rightarrow \quad \begin{aligned} \boldsymbol{\Lambda} &= \left(\mathbf{J}_e \mathbf{M}^{-1} \mathbf{J}_e^T\right)^{-1} \\ \boldsymbol{\mu} &= \boldsymbol{\Lambda}\mathbf{J}_e \mathbf{M}^{-1}\mathbf{b} - \boldsymbol{\Lambda}\dot{\mathbf{J}}_e \dot{\mathbf{q}} \\ \mathbf{p} &= \boldsymbol{\Lambda}\mathbf{J}_e \mathbf{M}^{-1}\mathbf{g} \end{aligned}$$



End-effector Motion Control

- Determine a desired end-effector acceleration

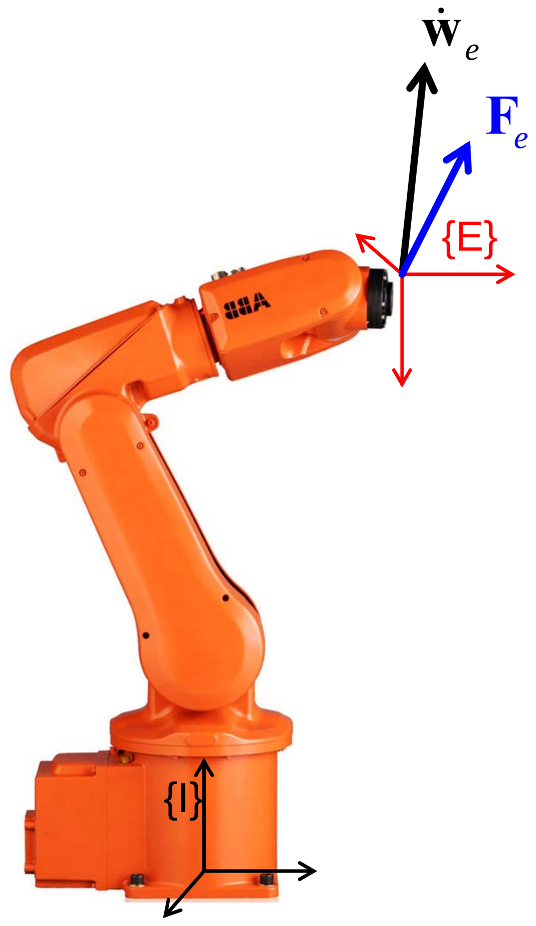
$$\dot{\mathbf{w}}_e^* = \mathbf{k}_p \mathbf{E} (\chi_e^* - \chi_e) + \mathbf{k}_d (\mathbf{w}_e^* - \mathbf{w}_e) + \dot{\mathbf{w}}_e(t)$$

Trajectory control

Note: a rotational error can be related to differenced in representation by $\Delta\phi = E_R(\chi_R)\Delta\chi_R$

- Determine the corresponding joint torque

$$\tau^* = \hat{\mathbf{J}}^T \left(\hat{\Lambda}_e \dot{\mathbf{w}}_e^* + \hat{\mu} + \hat{\mathbf{p}} \right)$$



Robots in Interaction

There is a long history in robots controlling motion and interaction

Operational Space Control

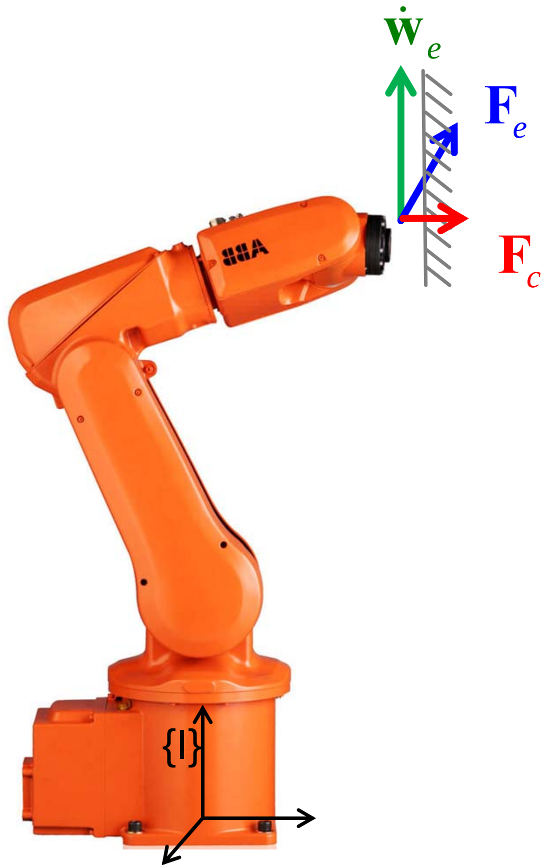
Generalized framework to control motion and force

- Extend end-effector dynamics in contact with contact force

$$\mathbf{F}_c + \boldsymbol{\Lambda}\dot{\mathbf{w}}_e + \boldsymbol{\mu} + \mathbf{p} = \mathbf{F}_e$$

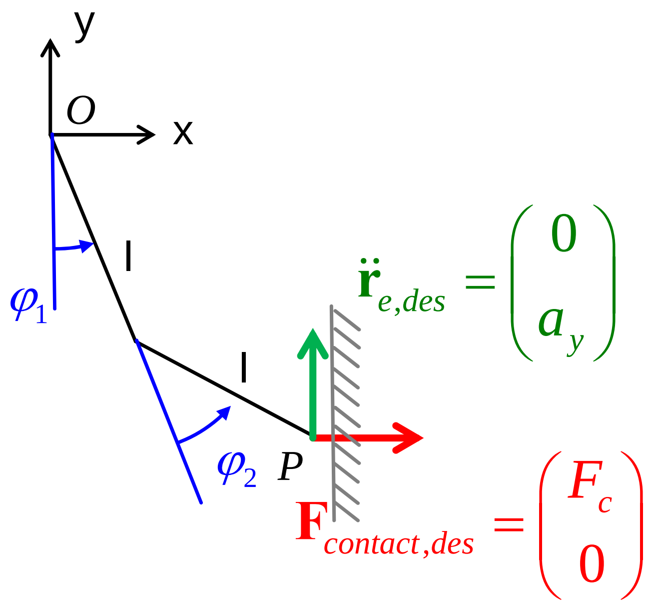
- Introduce selection matrices to separate motion force directions

$$\boldsymbol{\tau}^* = \hat{\mathbf{J}}^T \left(\hat{\boldsymbol{\Lambda}}\mathbf{S}_M\dot{\mathbf{w}}_e + \mathbf{S}_F\mathbf{F}_c + \hat{\tilde{\boldsymbol{\mu}}} + \hat{\mathbf{p}}\right)$$



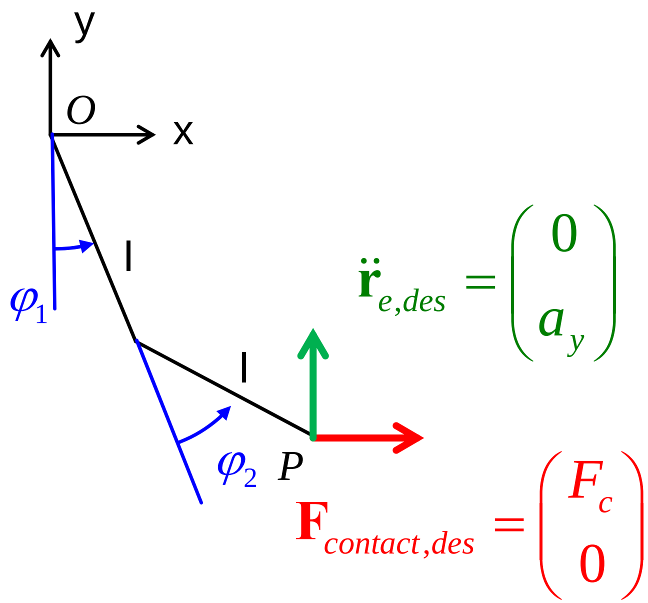
Operational Space Control
2-link example

- Given: $\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}$
- Find $\boldsymbol{\tau}$, s.t. the end-effector
  - accelerates with $\ddot{\mathbf{r}}_{e,des} = \begin{pmatrix} 0 & a_y \end{pmatrix}^T$
  - exerts the contact force $\mathbf{F}_{contact,des} = \begin{pmatrix} F_c & 0 \end{pmatrix}^T$



Operational Space Control
2-link example

- Given: $\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}$
- Find $\boldsymbol{\tau}$, s.t. the end-effector
  - accelerates with $\ddot{\mathbf{r}}_{e,des} = \begin{pmatrix} 0 & a_y \end{pmatrix}^T$
  - exerts the contact force $\mathbf{F}_{contact,des} = \begin{pmatrix} F_c & 0 \end{pmatrix}^T$



- End-effector position and Jacobian

$$\mathbf{r}_E = \begin{pmatrix} ls_1 + ls_{12} \\ -lc_1 - lc_{12} \end{pmatrix} \quad \mathbf{J}_e = \begin{bmatrix} lc_1 + lc_{12} & lc_{12} \\ ls_1 + ls_{12} & ls_{12} \end{bmatrix}$$

$$\boldsymbol{\Lambda} = \left(\mathbf{J}_e \mathbf{M}^{-1} \mathbf{J}_e^T\right)^{-1}$$
$$\boldsymbol{\mu} = \boldsymbol{\Lambda}\mathbf{J}_e\mathbf{M}^{-1}\mathbf{b} - \boldsymbol{\Lambda}\dot{\mathbf{J}}\dot{\mathbf{q}}$$
$$\mathbf{p} = \boldsymbol{\Lambda}\mathbf{J}_e\mathbf{M}^{-1}\mathbf{g}$$

  - Desired end-effector dynamics

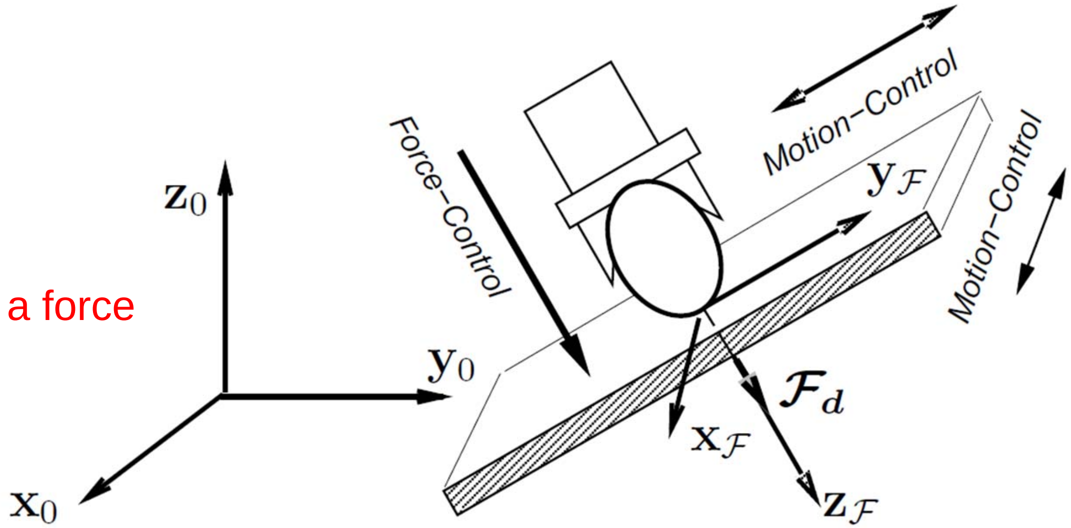
$$\boldsymbol{\tau}^* = \hat{\mathbf{J}}_e^T \left(\hat{\boldsymbol{\Lambda}}\ddot{\mathbf{r}}_{e,des} + \hat{\boldsymbol{\mu}} + \hat{\mathbf{p}} + \mathbf{F}_{contace,des}\right)$$

How to Find a Selection Matrix

- Selection matrix in local frame

$$\Sigma_p = \begin{bmatrix} \sigma_{px} & 0 & 0 \\ 0 & \sigma_{py} & 0 \\ 0 & 0 & \sigma_{pz} \end{bmatrix}$$

1: it can move
0: it can apply a force



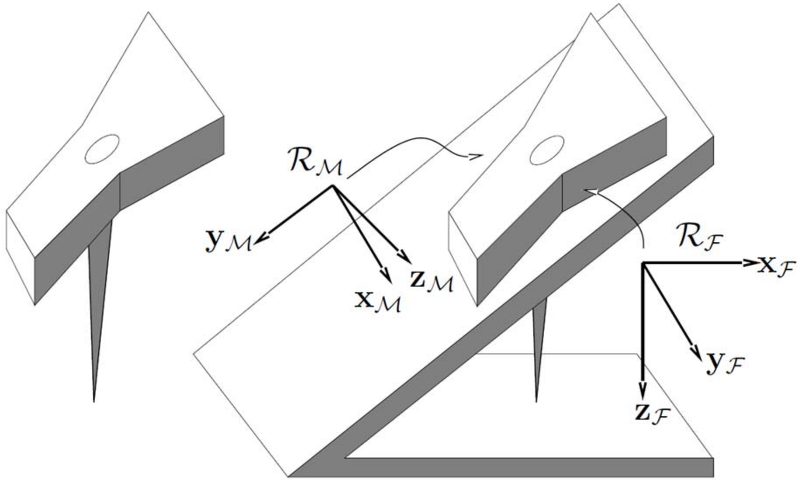
- Rotation between contact force and world frame

$$\mathbf{S}_M = \mathbf{C}^T \Sigma_p \mathbf{C} \qquad \mathbf{S}_F = \mathbf{C}^T \left(\mathbb{I}_3 - \Sigma_p\right) \mathbf{C}$$

How to Find a Selection Matrix

- Selection matrix in local frame

$$\boldsymbol{\Sigma}_p = \begin{bmatrix} \sigma_{px} & 0 & 0 \\ 0 & \sigma_{py} & 0 \\ 0 & 0 & \sigma_{pz} \end{bmatrix} \quad \boldsymbol{\Sigma}_r = \begin{bmatrix} \sigma_{rx} & 0 & 0 \\ 0 & \sigma_{ry} & 0 \\ 0 & 0 & \sigma_{rz} \end{bmatrix}$$



- Rotation between contact force and world frame

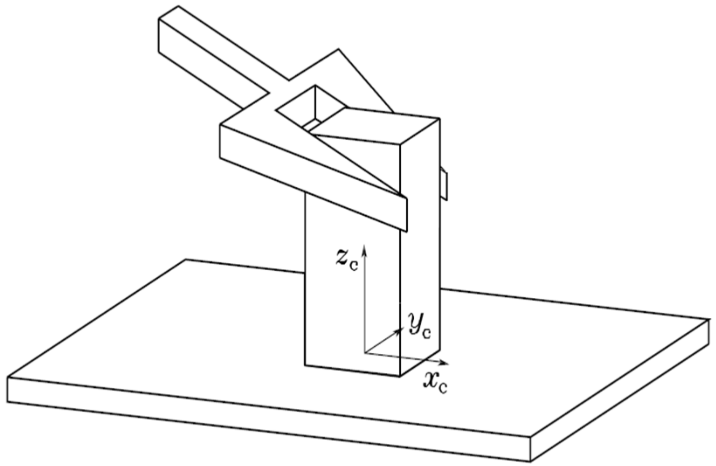
$$\mathbf{S}_M = \begin{bmatrix} \mathbf{C}^T\boldsymbol{\Sigma}_p\mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^T\boldsymbol{\Sigma}_r\mathbf{C} \end{bmatrix} \qquad \mathbf{S}_F = \begin{bmatrix} \mathbf{C}^T\left(\mathbb{I}_3 - \boldsymbol{\Sigma}_p\right)\mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^T\left(\mathbb{I}_3 - \boldsymbol{\Sigma}_r\right)\mathbf{C} \end{bmatrix}$$

Sliding a Prismatic Object Along a Surface

- Assume friction less contact surface

$$\boldsymbol{\Sigma}_{Mp} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{\Sigma}_{Mr} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{\Sigma}_{Fp} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{\Sigma}_{Fr} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

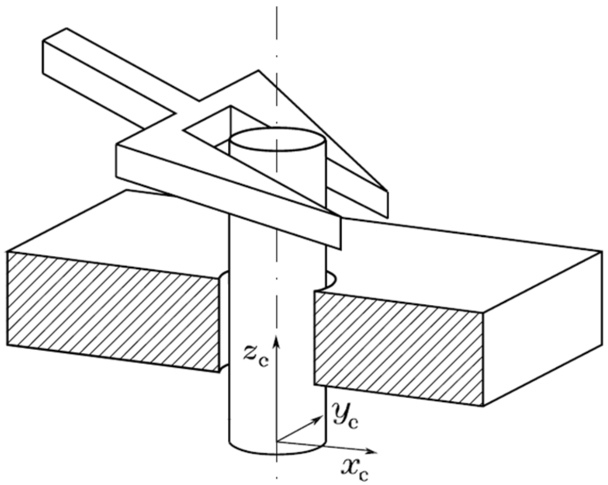


Inserting a Cylindrical Peg in a Hole

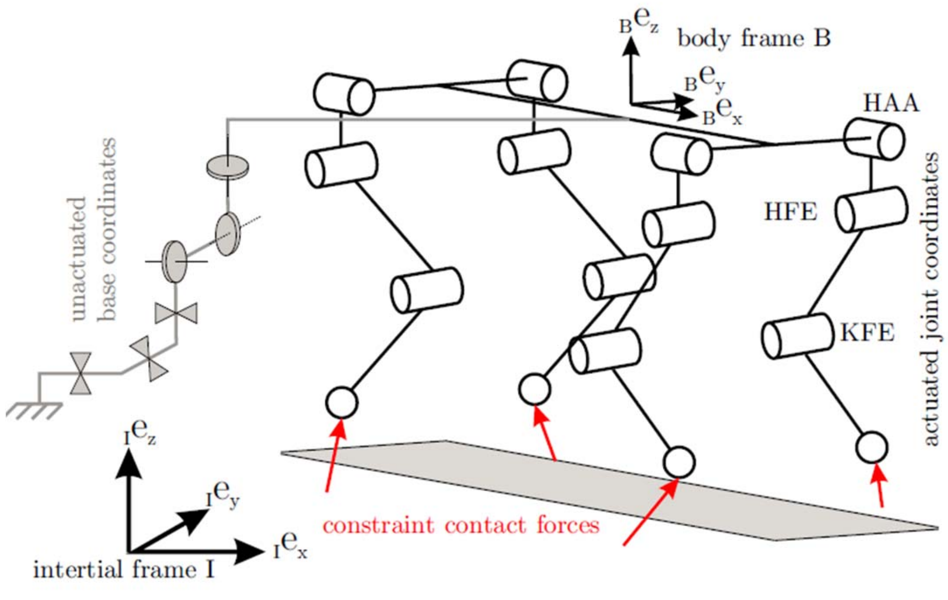
- Find the selection matrix (in local frame)

$$\boldsymbol{\Sigma}_{Mp} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{\Sigma}_{Mr} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

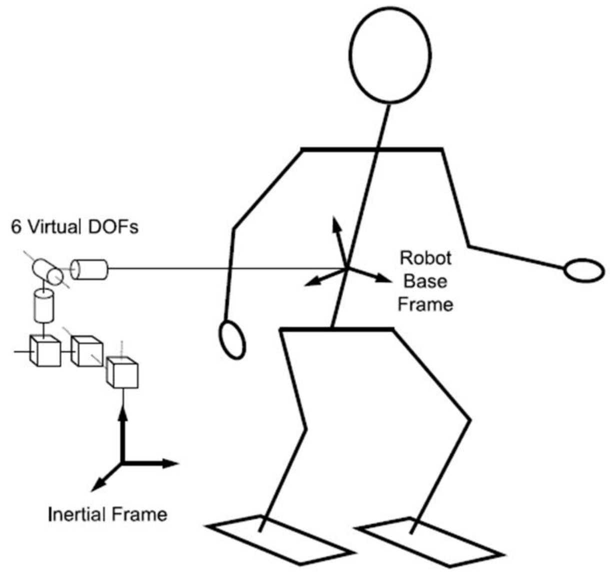
$$\boldsymbol{\Sigma}_{Fp} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{\Sigma}_{Fr} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$



Inverse Dynamics of Floating Base Systems



(a) Quadruped



(b) Humanoid

Recapitulation: Support Consistent Dynamics

- Equation of motion $$\mathbf{M}(\mathbf{q})\dot{\mathbf{u}} + \mathbf{b}(\mathbf{q},\mathbf{u}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T \mathbf{F}_c = \mathbf{S}^T \boldsymbol{\tau}$$
  - Cannot directly be used for control due to the occurrence of contact forces
- Contact constraint $$\ddot{\mathbf{r}}_c = \mathbf{J}_c \dot{\mathbf{u}} + \dot{\mathbf{J}}_c \mathbf{u} = 0$$
- Contact force $$\mathbf{F}_c = \left(\mathbf{J}_c \mathbf{M}^{-1} \mathbf{J}_c^T\right)^{-1} \left(\mathbf{J}_c \mathbf{M}^{-1} \left(\mathbf{S}^T \boldsymbol{\tau} - \mathbf{b} - \mathbf{g}\right) + \dot{\mathbf{J}}_c \mathbf{u}\right)$$
  - Back-substitute in (1), replace $\dot{\mathbf{J}}_s \dot{\mathbf{q}} = -\mathbf{J}_s \ddot{\mathbf{q}}$ and use support null-space projection $$\mathbf{N}_c = \mathbb{I} - \mathbf{M}^{-1} \mathbf{J}_c^T \left(\mathbf{J}_c \mathbf{M}^{-1} \mathbf{J}_c^T\right)^{-1} \mathbf{J}_c$$
- Support consistent dynamics $$\mathbf{N}_c^T \left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g}\right) = \mathbf{N}_c^T \mathbf{S}^T \boldsymbol{\tau}$$

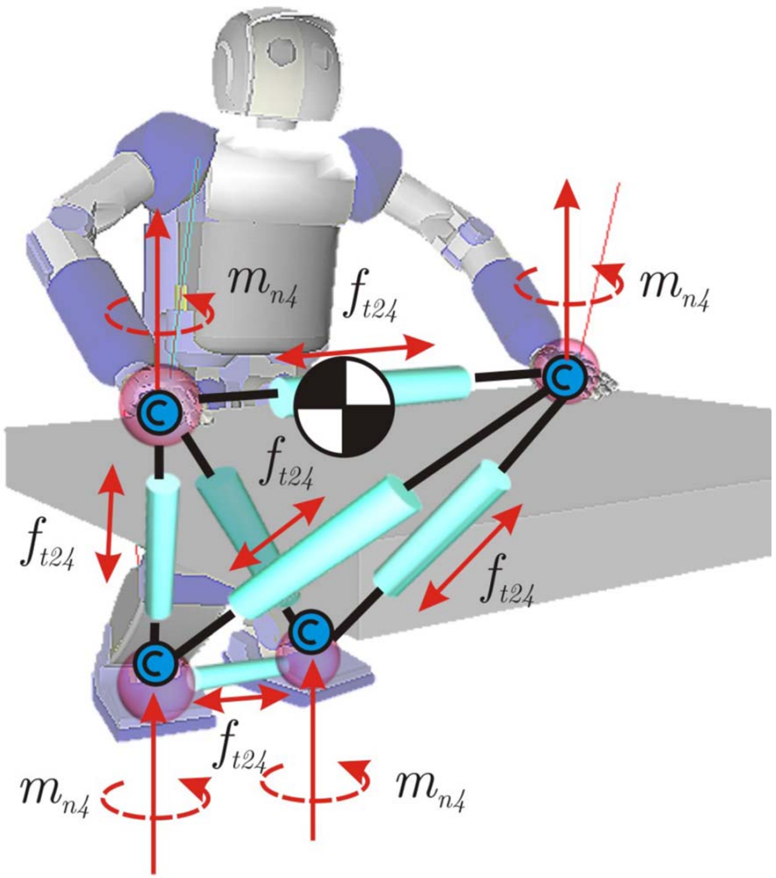
Inverse Dynamics of Floating Base Systems

- Equation of motion of floating base systems

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{u}} + \mathbf{b}(\mathbf{q},\mathbf{u}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T \mathbf{F}_c = \mathbf{S}^T \boldsymbol{\tau}$$

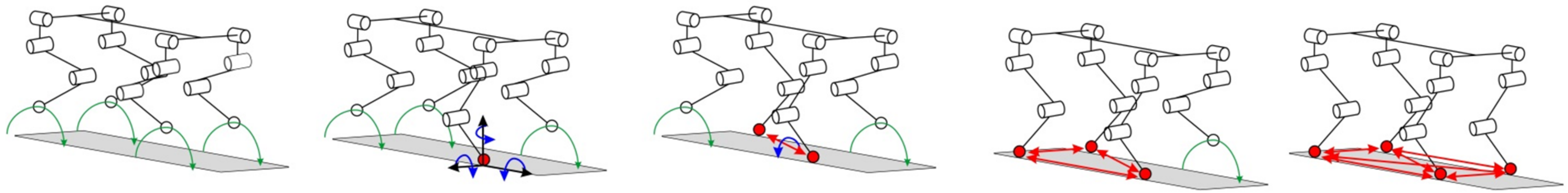
- Support-consistent $$\mathbf{N}_c^T (\mathbf{M}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g}) = \mathbf{N}_c^T \mathbf{S}^T \boldsymbol{\tau}$$
- Inverse-dynamics $$\boldsymbol{\tau}^* = \left(\mathbf{N}_c^T \mathbf{S}^T\right)^+ \mathbf{N}_c^T (\mathbf{M}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g})$$
- Multiple solutions $$\boldsymbol{\tau}^* = \left(\mathbf{N}_c^T \mathbf{S}^T\right)^+ \mathbf{N}_c^T (\mathbf{M}\ddot{\mathbf{q}}^* + \mathbf{b} + \mathbf{g}) + \mathcal{N}\left(\mathbf{N}_c^T \mathbf{S}^T\right) \boldsymbol{\tau}_0^*$$

Some Examples of Using Internal Forces



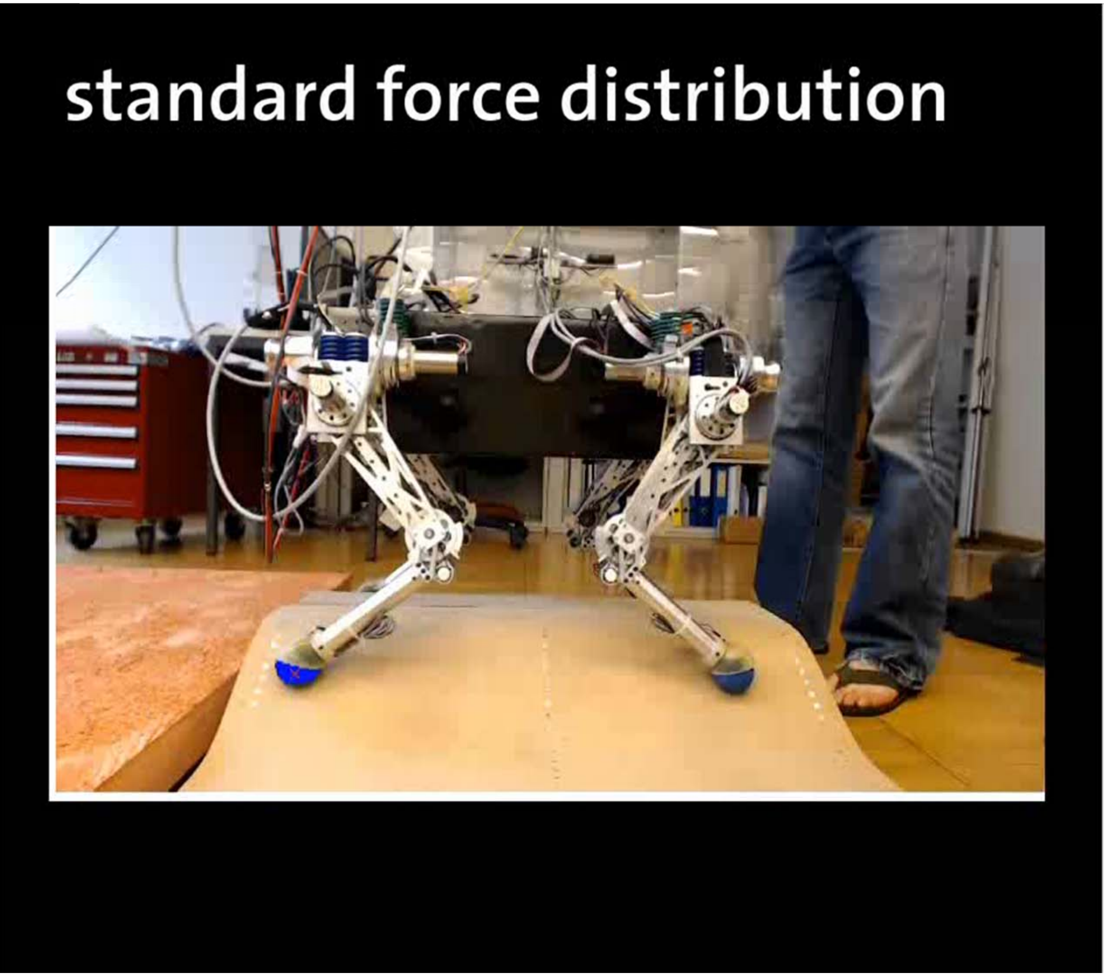
Recapitulation: Quadrupedal Robot with Point Feet

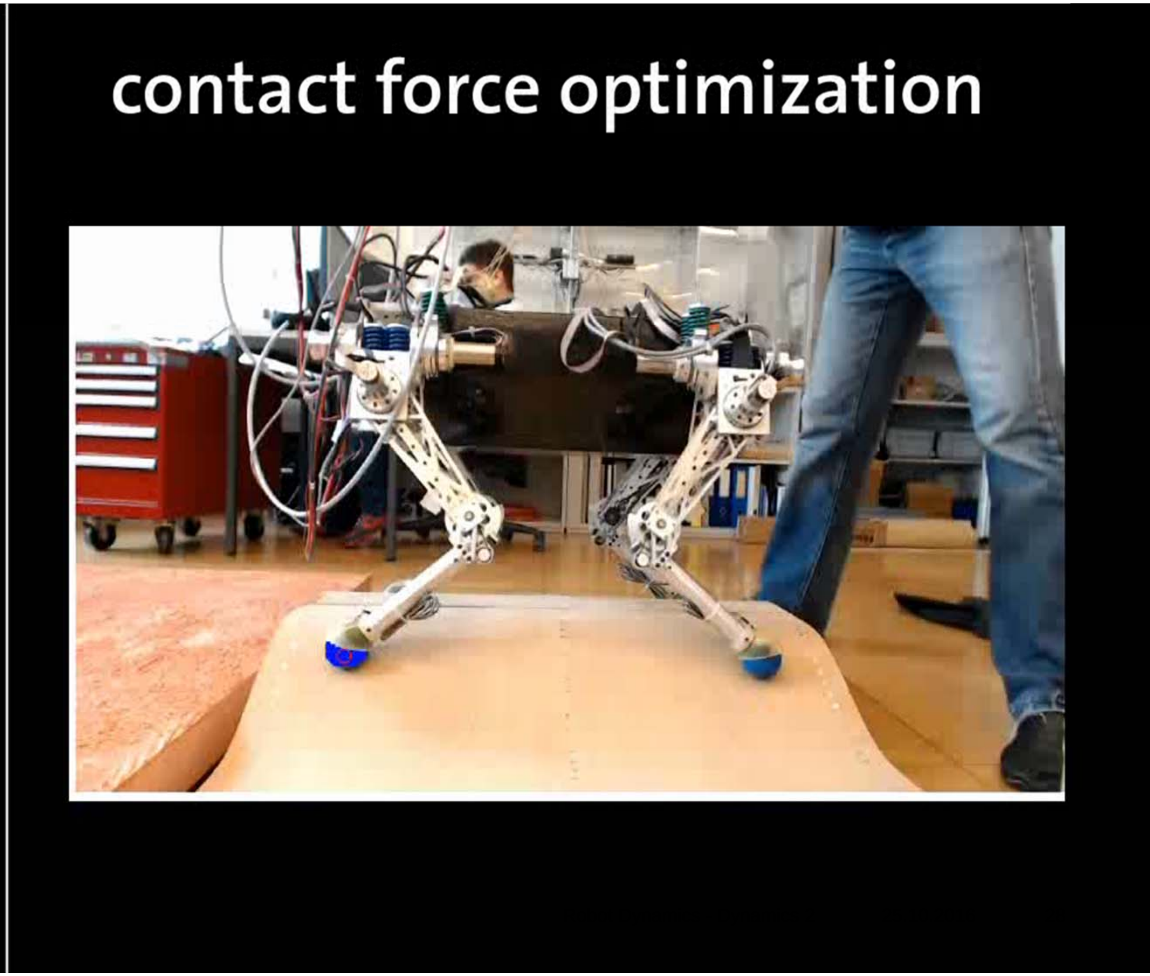
- Floating base system with 12 actuated joint and 6 base coordinates (18DoF)



Total constraints	0	3	6	9	12
Internal constraints	0	0	1	3	6
Uncontrollable DoFs	6	3	1	0	0

Some Examples of Using Internal Forces





Operational Space Control as Quadratic Program

A general problem

$$\min_{\mathbf{x}} \quad \|\mathbf{A}_i \mathbf{x} - \mathbf{b}_i\|_2 \qquad \mathbf{x} = \begin{pmatrix} \dot{\mathbf{u}} \\ \mathbf{F}_c \\ \boldsymbol{\tau} \end{pmatrix}$$

- We search for a solution that fulfills the equation of motion

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{u}} + \mathbf{b}(\mathbf{q},\mathbf{u}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T \mathbf{F}_c = \mathbf{S}^T \boldsymbol{\tau} \Rightarrow \mathbf{A} = \begin{bmatrix} \hat{\mathbf{M}} & \hat{\mathbf{J}}_c^T & -\mathbf{S}^T \end{bmatrix} \qquad \mathbf{b} = -\hat{\mathbf{b}} - \hat{\mathbf{g}}$$

  - Motion tasks: $\mathbf{J}\dot{\mathbf{u}} + \dot{\mathbf{J}}\mathbf{u} = \dot{\mathbf{w}}^*$ $\Rightarrow \mathbf{A} = \begin{bmatrix} \hat{\mathbf{J}}_i & \mathbf{0} & \mathbf{0} \end{bmatrix} \qquad \mathbf{b} = \dot{\mathbf{w}}^* - \hat{\dot{\mathbf{J}}}_i \mathbf{u}$
  - Force tasks: $\mathbf{F}_i = \mathbf{F}_i^*$ $\Rightarrow \mathbf{A} = \begin{bmatrix} \mathbf{0} & \hat{\mathbf{J}}_i^T & \mathbf{0} \end{bmatrix} \qquad \mathbf{b} = \mathbf{F}_i^*$
  - Torque min: $\min \|\boldsymbol{\tau}\|_2$ $\Rightarrow \mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbb{I} \end{bmatrix} \qquad \mathbf{b} = \mathbf{0}$

Solving a Set of QPs

- QPs need different priority!!
- Exploit Null-space of tasks with higher priority
- Every step = quadratic problem with constraints
- Use iterative null-space projection *(formula in script)*

$$\min_{\mathbf{x}} \quad \|\mathbf{A}_i\mathbf{x} - \mathbf{b}_i\|_2$$
$$s.t. \quad \underbrace{\begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{i-1} \end{bmatrix}}_{\hat{\mathbf{A}}_{i-1}} \mathbf{x} - \underbrace{\begin{pmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{i-1} \end{pmatrix}}_{\hat{\mathbf{b}}_{i-1}} = \mathbf{c}$$

n_T = Number of Tasks

$\mathbf{x} = \mathbf{0}$ ▷ initial optimal solution

$\mathbf{N}_1 = \mathbb{I}$ ▷ initial null-space projector

for $i = 1 \to n_T$ **do**

  $\mathbf{x}_i = (\mathbf{A}_i\mathbf{N}_i)^+ (\mathbf{b}_i - \mathbf{A}_i\mathbf{x})$

  $\mathbf{x} = \mathbf{x} + \mathbf{N}_i\mathbf{x}_i$

  $\mathbf{N}_{i+1} = \mathcal{N}\left(\begin{bmatrix}\mathbf{A}_1^T & \dots & \mathbf{A}_i^T\end{bmatrix}^T\right)$

end for

Behavior as Multiple Tasks