ETH zürich

Lecture «Robot Dynamics»: Dynamics and Control

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)

Marco Hutter, Roland Siegwart, and Thomas Stastny

19.09.2017	Intro and Outline	Course Introduction; Recapitulation Position, Linear Velocity			
26.09.2017	Kinematics 1	Rotation and Angular Velocity; Rigid Body Formulation, Transformation	26.09.2017	Exercise 1a	Kinematics Modeling the ABB arm
03.10.2017	Kinematics 2	Kinematics of Systems of Bodies; Jacobians	03.10.2017	Exercise 1b	Differential Kinematics of the ABB arm
10.10.2017	Kinematics 3	Kinematic Control Methods: Inverse Differential Kinematics, Inverse Kinematics; Rotation Error; Multi-task Control	10.10.2017	Exercise 1c	Kinematic Control of the ABB Arm
17.10.2017	Dynamics L1	Multi-body Dynamics	17.10.2017	Exercise 2a	Dynamic Modeling of the ABB Arm
24.10.2017	Dynamics L2	Floating Base Dynamics	24.10.2017		
31.10.2017	Dynamics L3	Dynamic Model Based Control Methods	31.10.2017	Exercise 2b	Dynamic Control Methods Applied to the ABB arm
07.11.2017	Legged Robot	Dynamic Modeling of Legged Robots & Control	07.11.2017	Exercise 3	Legged robot
14.11.2017	Case Studies 1	Legged Robotics Case Study	14.11.2017		
21.11.2017	Rotorcraft	Dynamic Modeling of Rotorcraft & Control	21.11.2017	Exercise 4	Modeling and Control of Multicopter
28.11.2017	Case Studies 2	Rotor Craft Case Study	28.11.2017		
05.12.2017	Fixed-wing	Dynamic Modeling of Fixed-wing & Control	05.12.2017	Exercise 5	Fixed-wing Control and Simulation
12.12.2017	Case Studies 3	Fixed-wing Case Study (Solar-powered UAVs - AtlantikSolar, Vertical Take-off and Landing UAVs – Wingtra)			
19.12.2017	Summery and Outlook	Summery; Wrap-up; Exam			

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

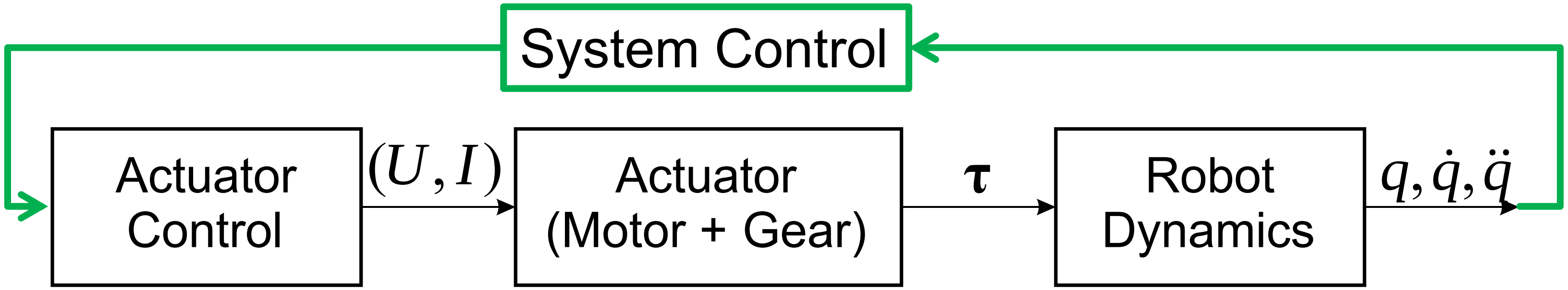
- Today:
  - How can we use this information in order to control the robot

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

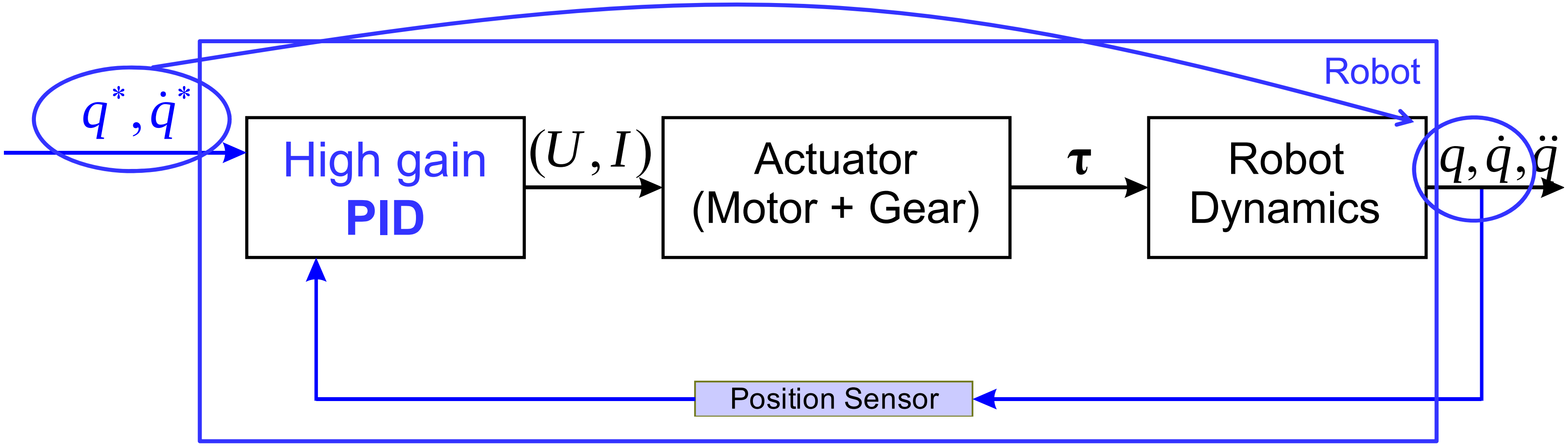
$\ddot{\mathbf{q}}$	Generalized coordinates
$\mathbf{M}(\mathbf{q})$	Mass matrix
$\mathbf{b}(\mathbf{q},\dot{\mathbf{q}})$	Centrifugal and Coriolis forces
$\mathbf{g}(\mathbf{q})$	Gravity forces
$\boldsymbol{\tau}$	Generalized forces
$\mathbf{S}_\tau$	Selection matrix/Jacobian
$\mathbf{F}_c$	External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

Position vs. Torque Controlled Robot Arms

Setup of a Robot Arm

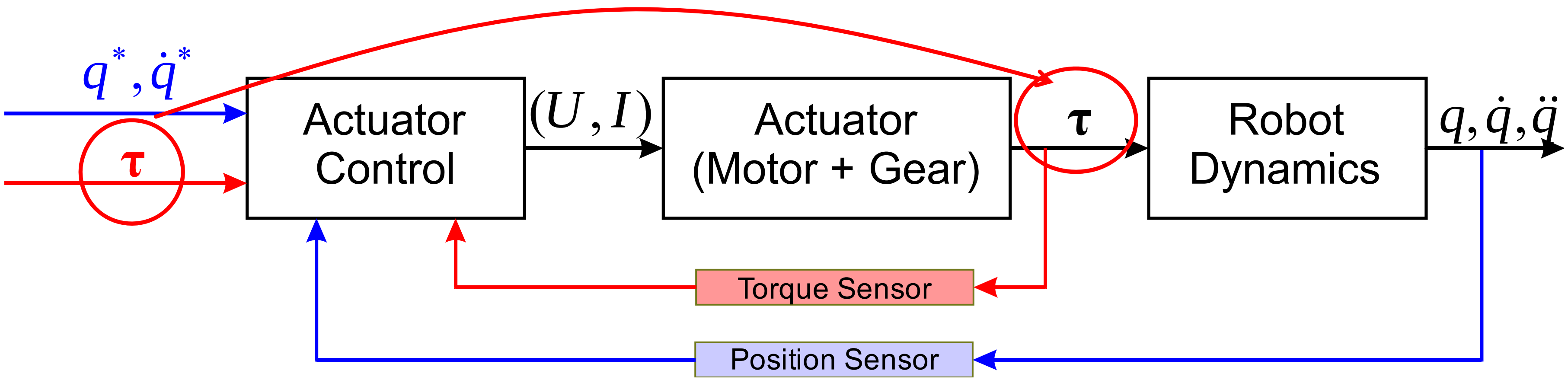


Classical Position Control of a Robot Arm



- 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

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

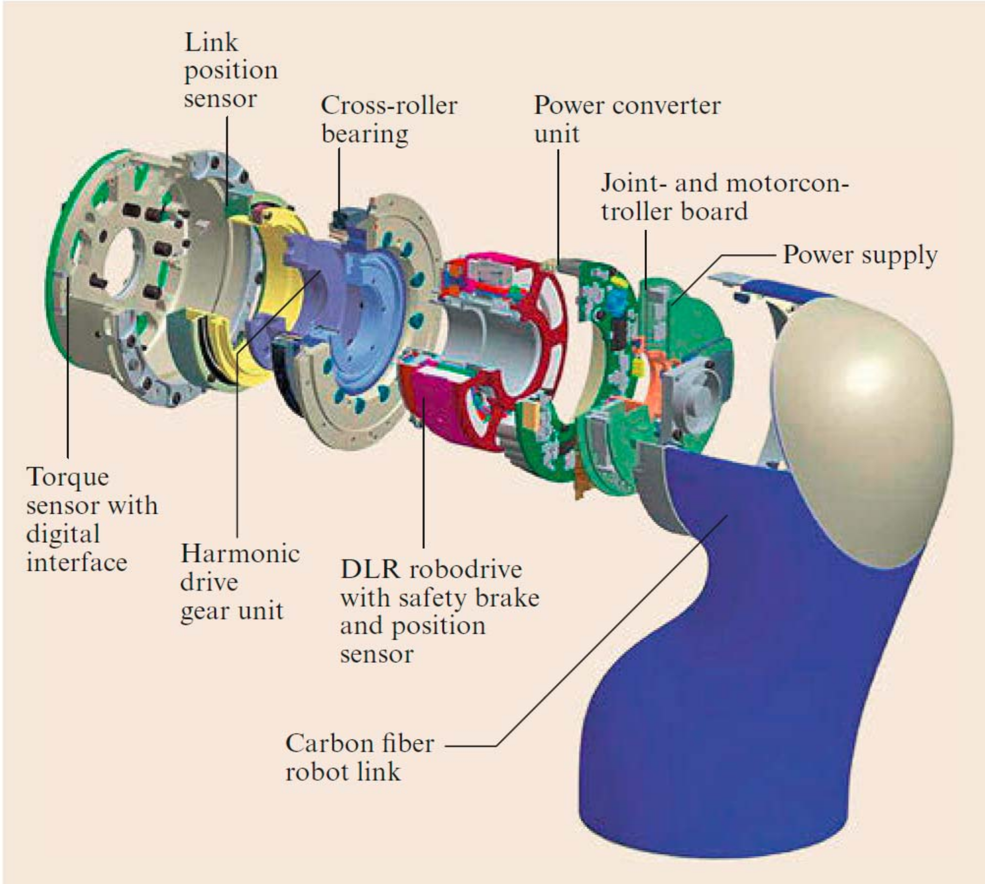


Fig. 11.8 Exploded view of a joint of the *DLR LWR-III* lightweight manipulator and its sensor suite

FRANKA – an example of a force controllable robot arm

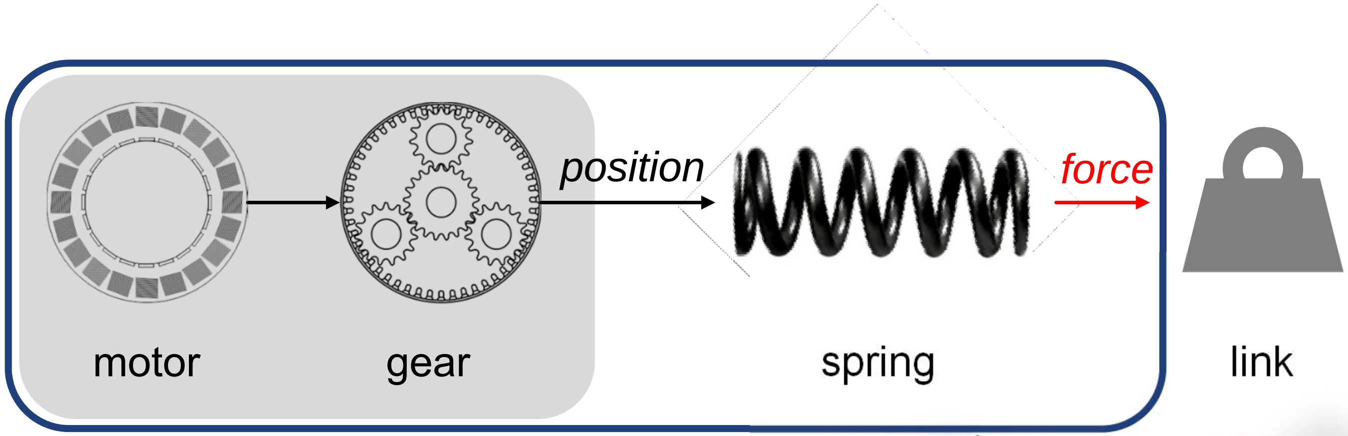
CHAPTER I

THIS IS FRANKA

ANYpulator

An example for a robot that can interact

- Special force controllable actuators
  - Dynamic motion
  - Safe interaction



Series Elastic Actuator

Joint Impedance Control

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

- 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

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

- 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

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

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 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 $\ddot{\mathbf{q}} = \sum_{i=1}^{n_T} \mathbf{N}_i \ddot{\mathbf{q}}_i, \quad \text{with} \quad \ddot{\mathbf{q}}_i = (\mathbf{J}_i \mathbf{N}_i)^+ \left(\mathbf{w}_i^* - \dot{\mathbf{J}}_i \dot{\mathbf{q}} - \mathbf{J} \sum_{k=1}^{i-1} \mathbf{N}_k \dot{\mathbf{q}}_k \right)$

Task Space Dynamics

- Joint-space dynamics

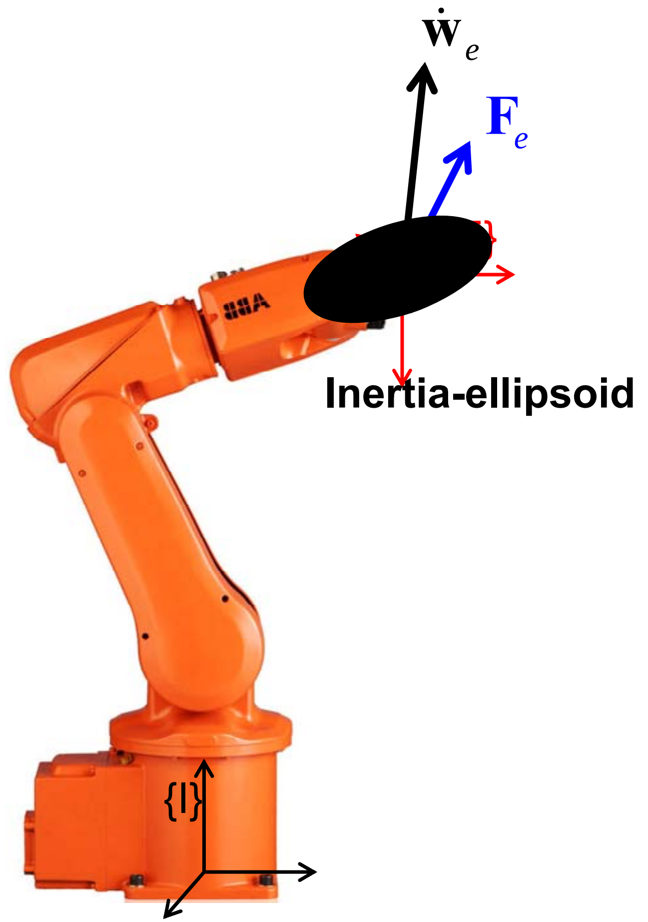
$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}$$

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 $\dot{\mathbf{w}}_e = \mathbf{J}_e\mathbf{M}^{-1}(\boldsymbol{\tau} - \mathbf{b} - \mathbf{g}) + \dot{\mathbf{J}}_e\dot{\mathbf{q}}$

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



End-effector Motion Control

- Determine a desired end-effector acceleration

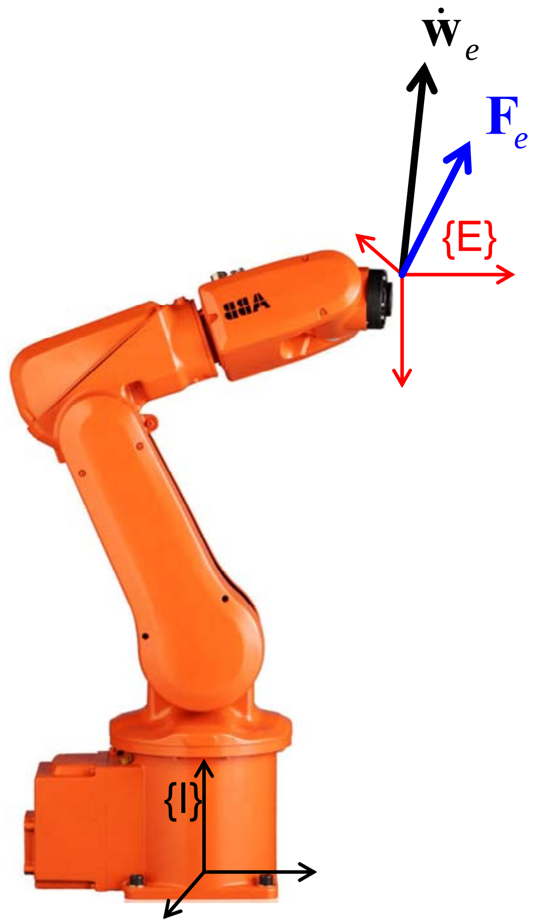
$$\dot{\mathbf{w}}_e^* = \mathbf{k}_p \mathbf{E}\left(\chi_e^* - \chi_e\right) + \mathbf{k}_d\left(\mathbf{w}_e^* - \mathbf{w}_e\right) + \dot{\mathbf{w}}_e(t)$$

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

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



Robots in Interaction

There is a long history in robots controlling motion and interaction

Operational Space Control

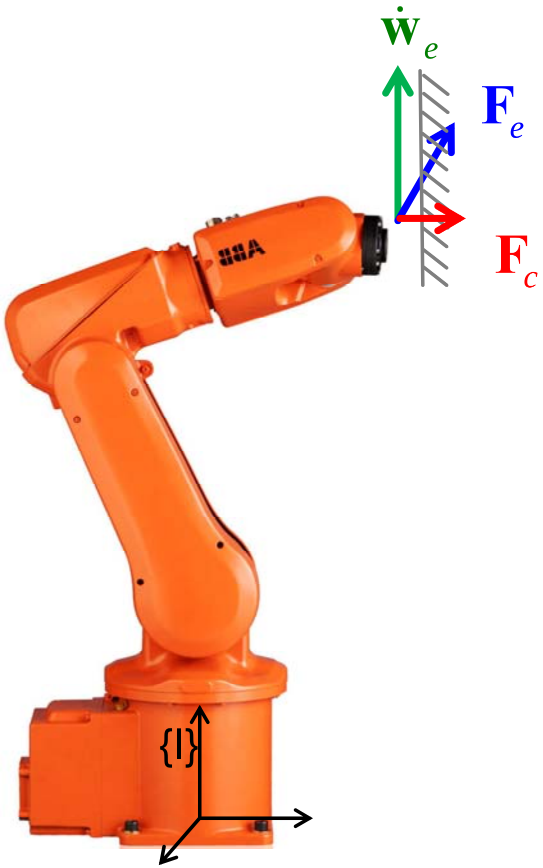
Generalized framework to control motion and force

- Extend end-effector dynamics in contact with contact force

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

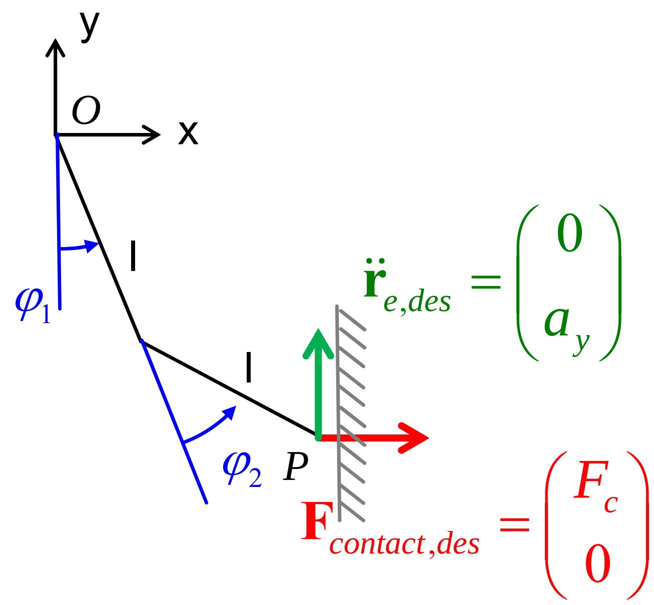
- Introduce selection matrices to separate motion force directions

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



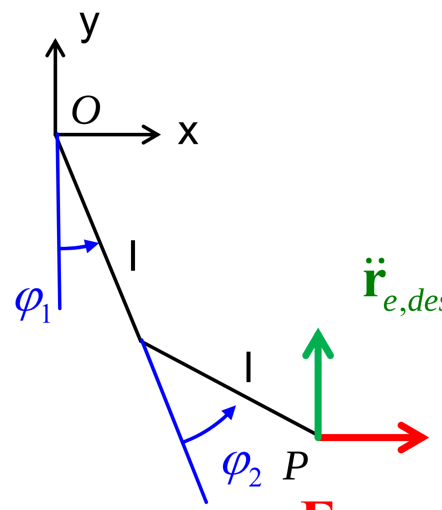
Operational Space Control
2-link example

- 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 $\mathbf{F}_{contact,des} = \begin{pmatrix} F_c & 0 \end{pmatrix}^T$



$$\ddot{\mathbf{r}}_{e,des} = \begin{pmatrix} 0 \\ a_y \end{pmatrix}$$

$$\mathbf{F}_{contact,des} = \begin{pmatrix} F_c \\ 0 \end{pmatrix}$$

- End-effector position and Jacobian

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

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

  - Desired end-effector dynamics

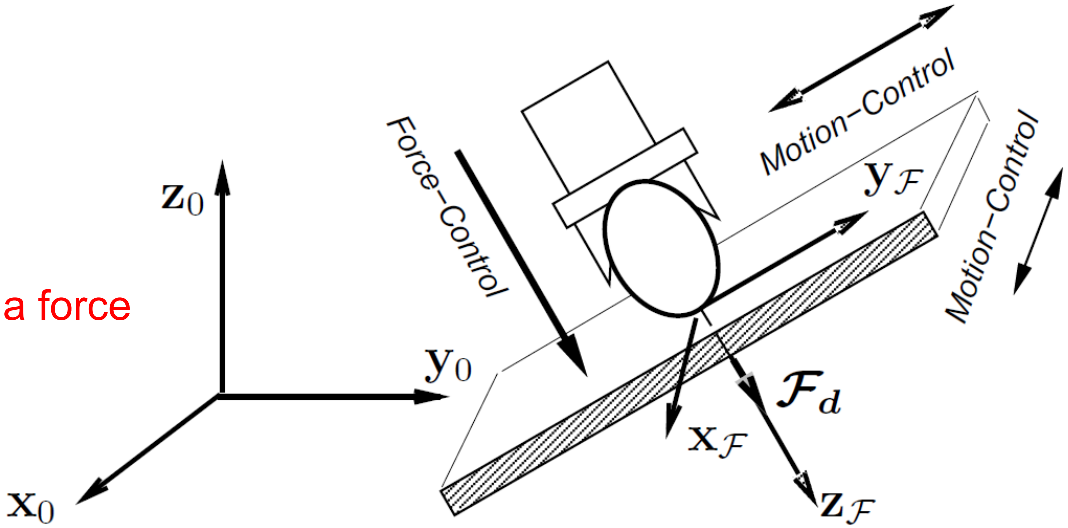
$$\boldsymbol{\tau}^* = \hat{\mathbf{J}}_e^T \left(\hat{\boldsymbol{\Lambda}}\ddot{\mathbf{r}}_{e,des} + \hat{\boldsymbol{\mu}} + \hat{\mathbf{p}} + \mathbf{F}_{contace,des} \right)$$

How to Find a Selection Matrix

- Selection matrix in local frame

$$\boldsymbol{\Sigma}_p = \begin{bmatrix} \sigma_{px} & 0 & 0 \\ 0 & \sigma_{py} & 0 \\ 0 & 0 & \sigma_{pz} \end{bmatrix}$$

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



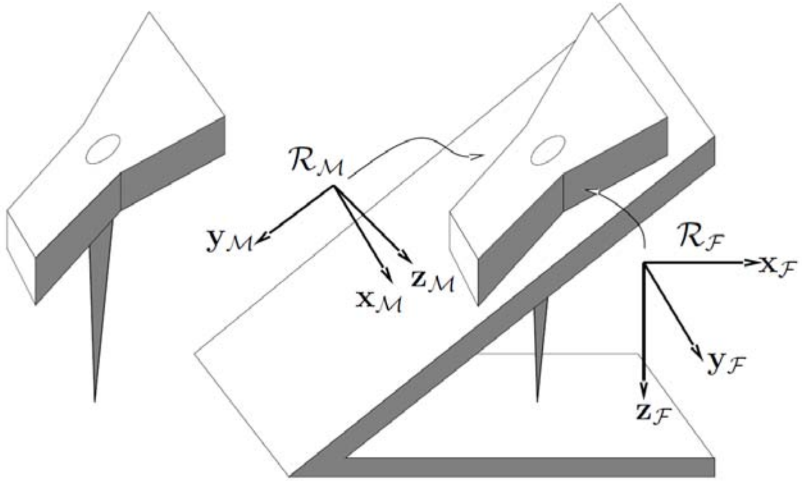
- Rotation between contact force and world frame

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

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

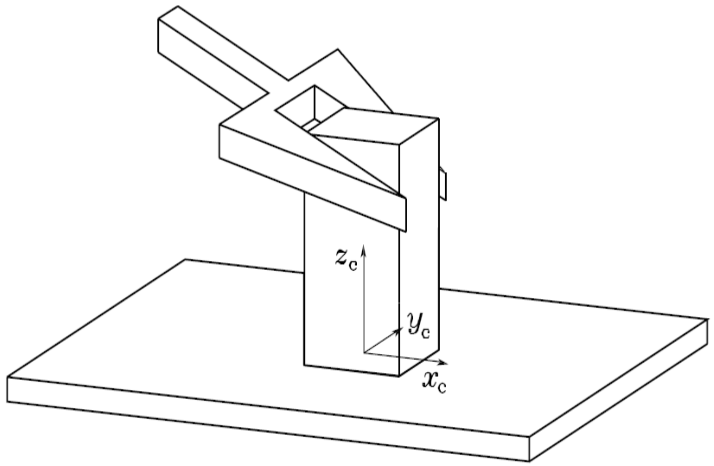
$$\mathbf{S}_M = \begin{bmatrix} \mathbf{C}^T \boldsymbol{\Sigma}_p \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^T \boldsymbol{\Sigma}_r \mathbf{C} \end{bmatrix} \qquad \mathbf{S}_F = \begin{bmatrix} \mathbf{C}^T \left(\mathbb{I}_3 - \boldsymbol{\Sigma}_p\right) \mathbf{C} & \mathbf{0} \\ \mathbf{0}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

Sliding a Prismatic Object Along a Surface

- Assume friction less contact surface

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

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

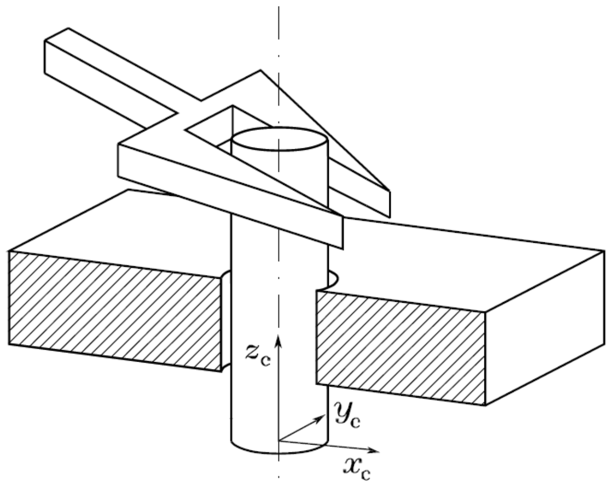


Inserting a Cylindrical Peg in a Hole

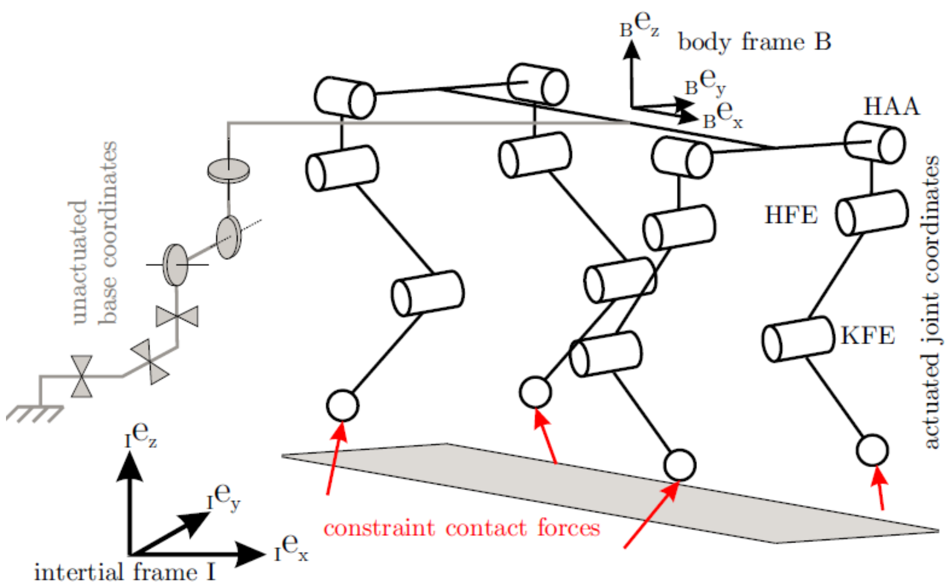
- Find the selection matrix (in local frame)

$$\mathbf{\Sigma}_{Mp} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{\Sigma}_{Mr} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

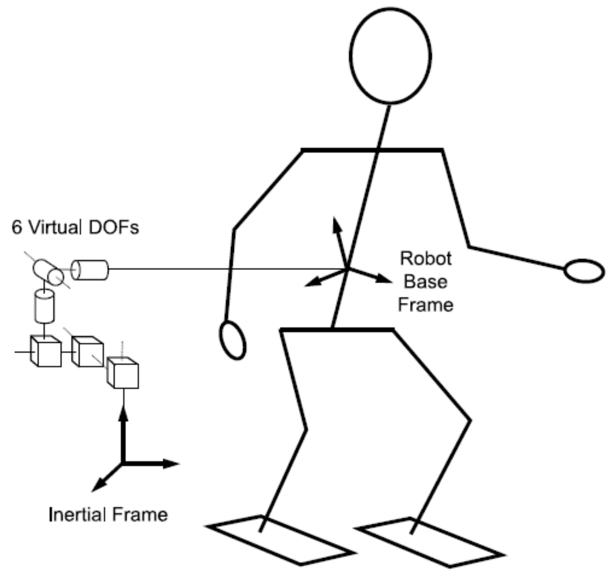
$$\mathbf{\Sigma}_{Fp} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \mathbf{\Sigma}_{Fr} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$



Inverse Dynamics of Floating Base Systems



(a) Quadruped

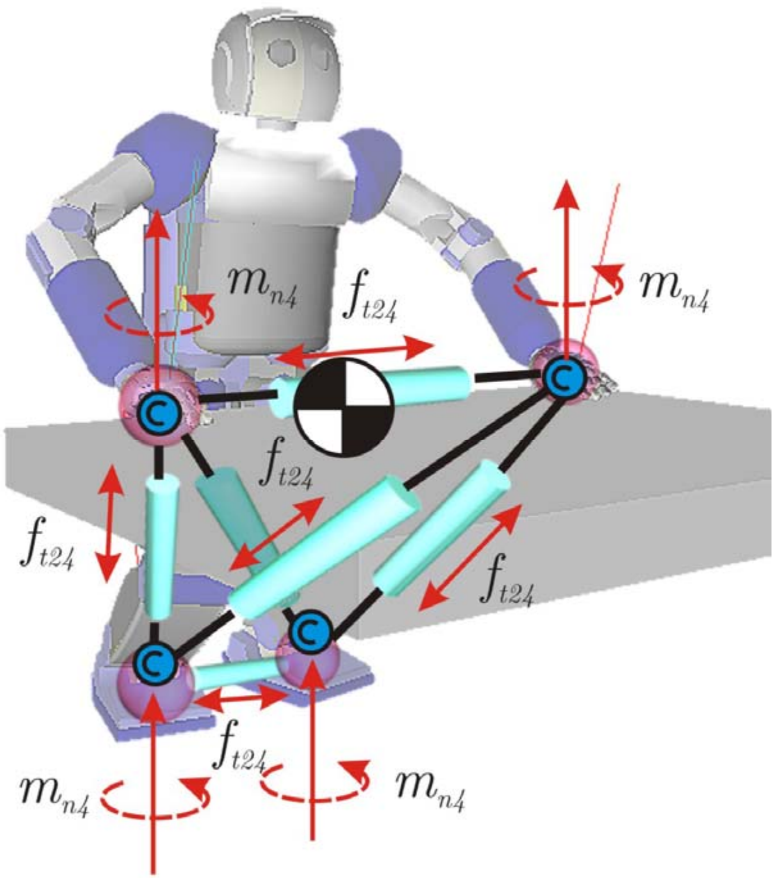


(b) Humanoid

Recapitulation: Support Consistent Dynamics

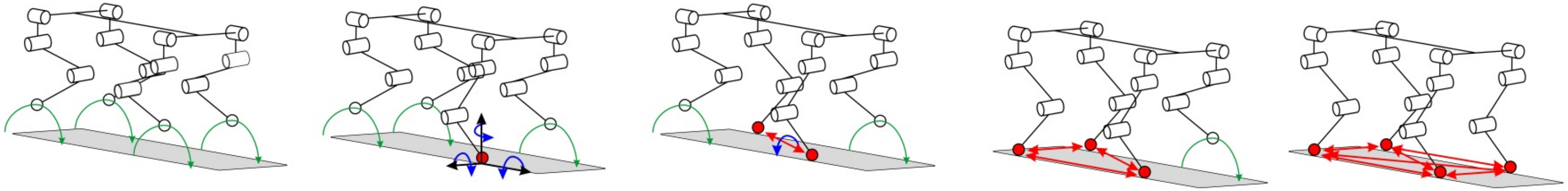
- Equation of motion $$\mathbf{M}(\mathbf{q})\dot{\mathbf{u}} + \mathbf{b}(\mathbf{q},\mathbf{u}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T\mathbf{F}_c = \mathbf{S}^T\boldsymbol{\tau}$$
  - Cannot directly be used for control due to the occurrence of contact forces
- Contact constraint $$\ddot{\mathbf{r}}_c = \mathbf{J}_c\dot{\mathbf{u}} + \dot{\mathbf{J}}_c\mathbf{u} = 0$$
- 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 $$\boldsymbol{\tau}^* = \left(\mathbf{N}_c^T\mathbf{S}^T\right)^+\mathbf{N}_c^T\left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g}\right)$$
- Multiple solutions $$\boldsymbol{\tau}^* = \left(\mathbf{N}_c^T\mathbf{S}^T\right)^+\mathbf{N}_c^T\left(\mathbf{M}\ddot{\mathbf{q}}^* + \mathbf{b} + \mathbf{g}\right) + \mathcal{N}\left(\mathbf{N}_c^T\mathbf{S}^T\right)\boldsymbol{\tau}_0^*$$

Some Examples of Using Internal Forces



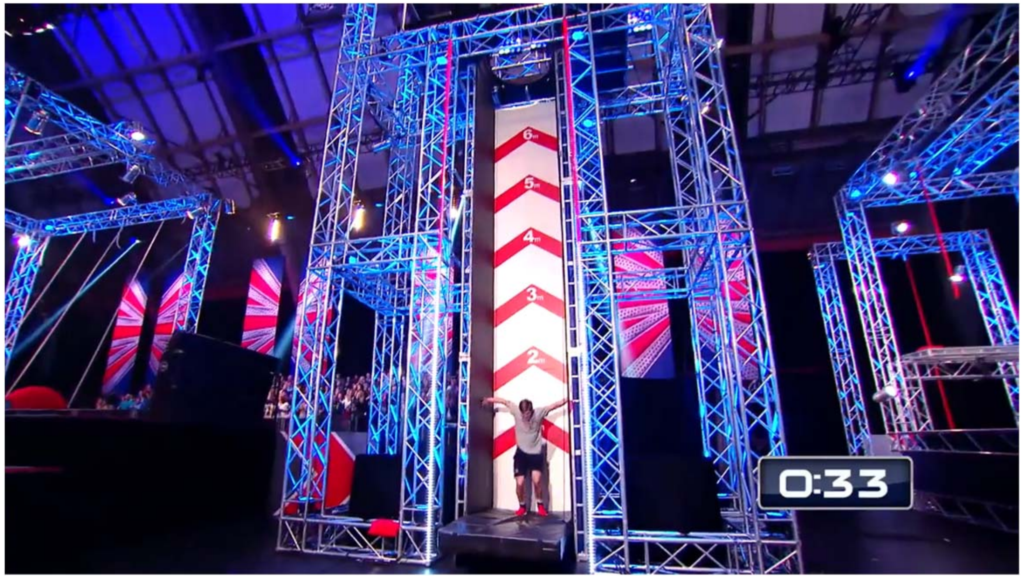
Recapitulation: Quadrupedal Robot with Point Feet

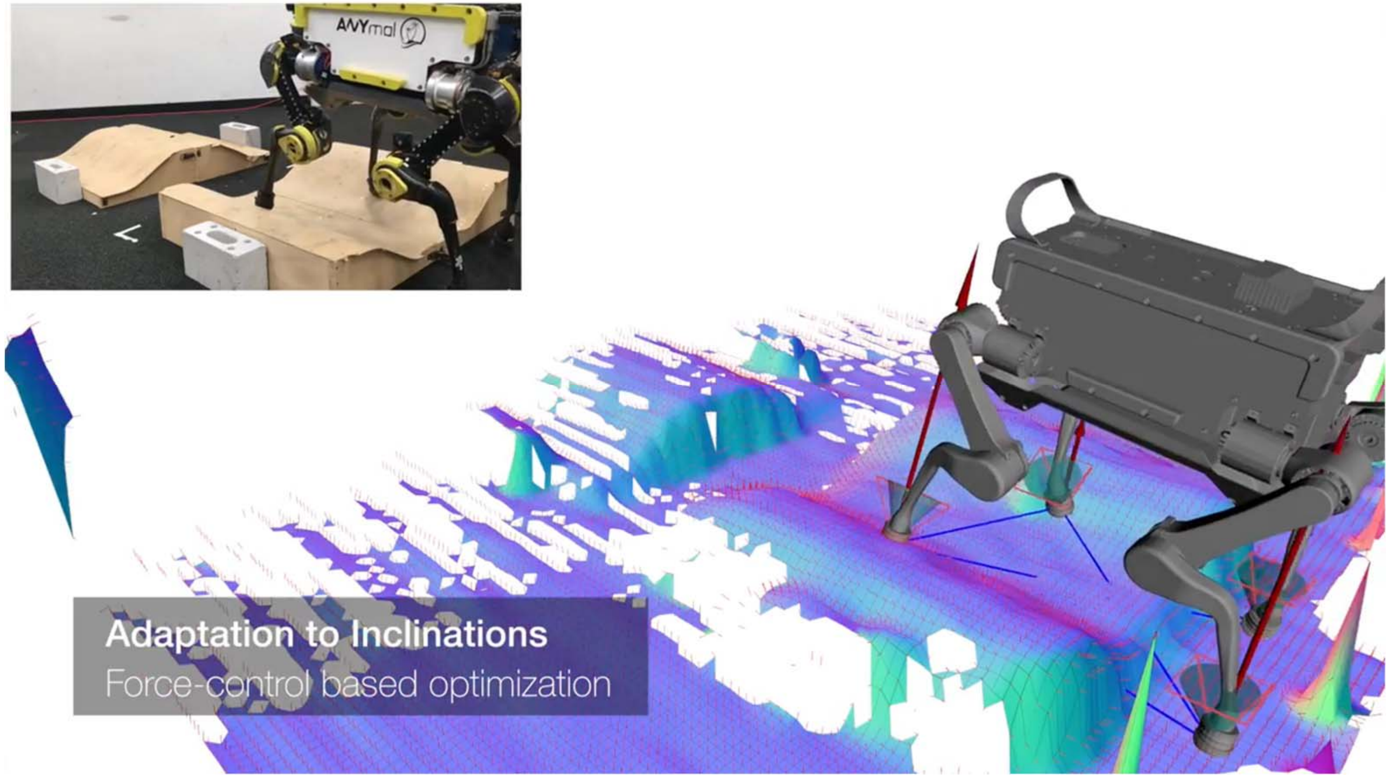
- Floating base system with 12 actuated joint and 6 base coordinates (18DoF)



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

Internal Forces
extreme example




ANYmal
Adaptation to Inclinations
Force-control based optimization

Least Square Optimization
some notes on quadratic optimization

$$\mathbf{A}\mathbf{x}-\mathbf{b}=\mathbf{0} \longrightarrow \mathbf{x}=\mathbf{A}^{+}\mathbf{b} \qquad \min_{\mathbf{x}}\left\|\mathbf{A}\mathbf{x}-\mathbf{b}\right\|_2 \qquad \min\left\|\mathbf{x}\right\|_2$$

$$\mathbf{A}_1\mathbf{x}_1-\mathbf{b}=\mathbf{A}_2\mathbf{x}_2 \longrightarrow \begin{pmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{pmatrix}=\begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}^{+}\mathbf{b} \qquad \min_{\mathbf{x}_1,\mathbf{x}_2}\left\|\begin{bmatrix}\mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}\begin{pmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{pmatrix}-\mathbf{b}\right\|_2 \qquad \min\left\|\begin{pmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{pmatrix}\right\|_2$$

$$\begin{matrix}\mathbf{A}_1\mathbf{x}-\mathbf{b}_1=\mathbf{0}\\ \mathbf{A}_2\mathbf{x}-\mathbf{b}_2=\mathbf{0}\end{matrix} \xrightarrow{\text{Equal priority}} \mathbf{x}=\begin{bmatrix}\mathbf{A}_1\\ \mathbf{A}_2\end{bmatrix}^{+}\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\end{pmatrix} \qquad \min_{\mathbf{x}}\left\|\begin{bmatrix}\mathbf{A}_1\\ \mathbf{A}_2\end{bmatrix}\mathbf{x}-\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\end{pmatrix}\right\|_2 \qquad \min\left\|\mathbf{x}\right\|_2$$

Hierarchy

$$\mathbf{x}=\mathbf{A}_1^{+}\mathbf{b}_1+\mathcal{N}\left(\mathbf{A}_1\right)\mathbf{x}_0 \qquad \min_{\mathbf{x}}\left\|\mathbf{A}_1\mathbf{x}-\mathbf{b}_1\right\|_2$$

$$\left.\begin{matrix}\mathbf{A}_2\mathbf{x}-\mathbf{b}_2=\mathbf{A}_2\left(\mathbf{A}_1^{+}\mathbf{b}_1+\mathcal{N}\left(\mathbf{A}_1\right)\mathbf{x}_0\right)-\mathbf{b}_2=\mathbf{0}\\ \mathbf{x}_0=\left(\mathbf{A}_2\mathcal{N}\left(\mathbf{A}_1\right)\right)^{+}\left(\mathbf{b}_2-\mathbf{A}_2\mathbf{A}_1^{+}\mathbf{b}_1\right)\end{matrix}\right\} \qquad \left\{\begin{matrix}\min_{\mathbf{x}}\left\|\mathbf{A}_2\mathbf{x}-\mathbf{b}_2\right\|_2\\ s.t.\left\|\mathbf{A}_1\mathbf{x}-\mathbf{b}_1\right\|=c_1\end{matrix}\right.$$

Least Square Optimization
Application to Inverse Dynamics

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{b} + \mathbf{g} = \boldsymbol{\tau}$$
$$\mathbf{J}_e\ddot{\mathbf{q}} + \dot{\mathbf{J}}_e\dot{\mathbf{q}} = \dot{\mathbf{w}}_e^*$$

$$\left.\begin{aligned} \begin{bmatrix}\mathbf{M} & -\mathbf{I}\end{bmatrix}\begin{pmatrix}\ddot{\mathbf{q}}\\ \boldsymbol{\tau}\end{pmatrix} + \mathbf{b} + \mathbf{g} = \mathbf{0} \\ \begin{bmatrix}\mathbf{J}_e & \mathbf{0}\end{bmatrix}\begin{pmatrix}\ddot{\mathbf{q}}\\ \boldsymbol{\tau}\end{pmatrix} + \dot{\mathbf{J}}\dot{\mathbf{q}} = \dot{\mathbf{w}}_e^* \end{aligned}\right\}$$

Single task $\min_{\ddot{\mathbf{q}},\boldsymbol{\tau}} \left\| \begin{bmatrix}\mathbf{M} & -\mathbf{I}\\ \mathbf{J}_e & \mathbf{0}\end{bmatrix}\begin{pmatrix}\ddot{\mathbf{q}}\\ \boldsymbol{\tau}\end{pmatrix} + \begin{pmatrix}\mathbf{b}+\mathbf{g}\\ \dot{\mathbf{J}}\dot{\mathbf{q}} - \dot{\mathbf{w}}_e^*\end{pmatrix} \right\|_2$

Priority $\begin{cases} \min_{\ddot{\mathbf{q}},\boldsymbol{\tau}} \left\| \begin{bmatrix}\mathbf{J}_e & \mathbf{0}\end{bmatrix}\begin{pmatrix}\ddot{\mathbf{q}}\\ \boldsymbol{\tau}\end{pmatrix} + \dot{\mathbf{J}}\dot{\mathbf{q}} - \dot{\mathbf{w}}_e^* \right\|_2 \\ s.t. \begin{bmatrix}\mathbf{M} & -\mathbf{I}\end{bmatrix}\begin{pmatrix}\ddot{\mathbf{q}}\\ \boldsymbol{\tau}\end{pmatrix} + \mathbf{b} + \mathbf{g} = \mathbf{0} \end{cases}$

Operational Space Control as Quadratic Program

A general problem

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

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

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

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

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 Ta

$\mathbf{x} = \mathbf{0}$ ... l optimal solution

$\mathbf{N}_1 = \mathbb{I}$ ... ll-space projector

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

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

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

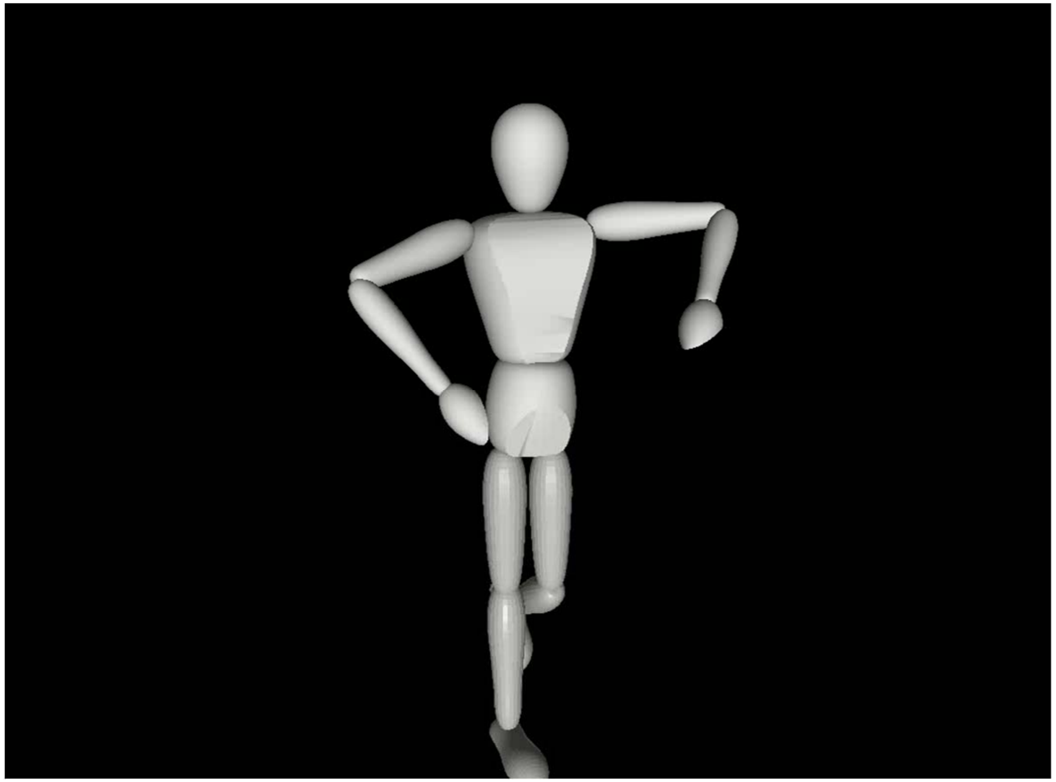
$\mathbf{N}_{i+1} = \mathcal{N}\left(\left[\mathbf{A}_1^T \quad \ldots \quad {}_i\right]\right)$

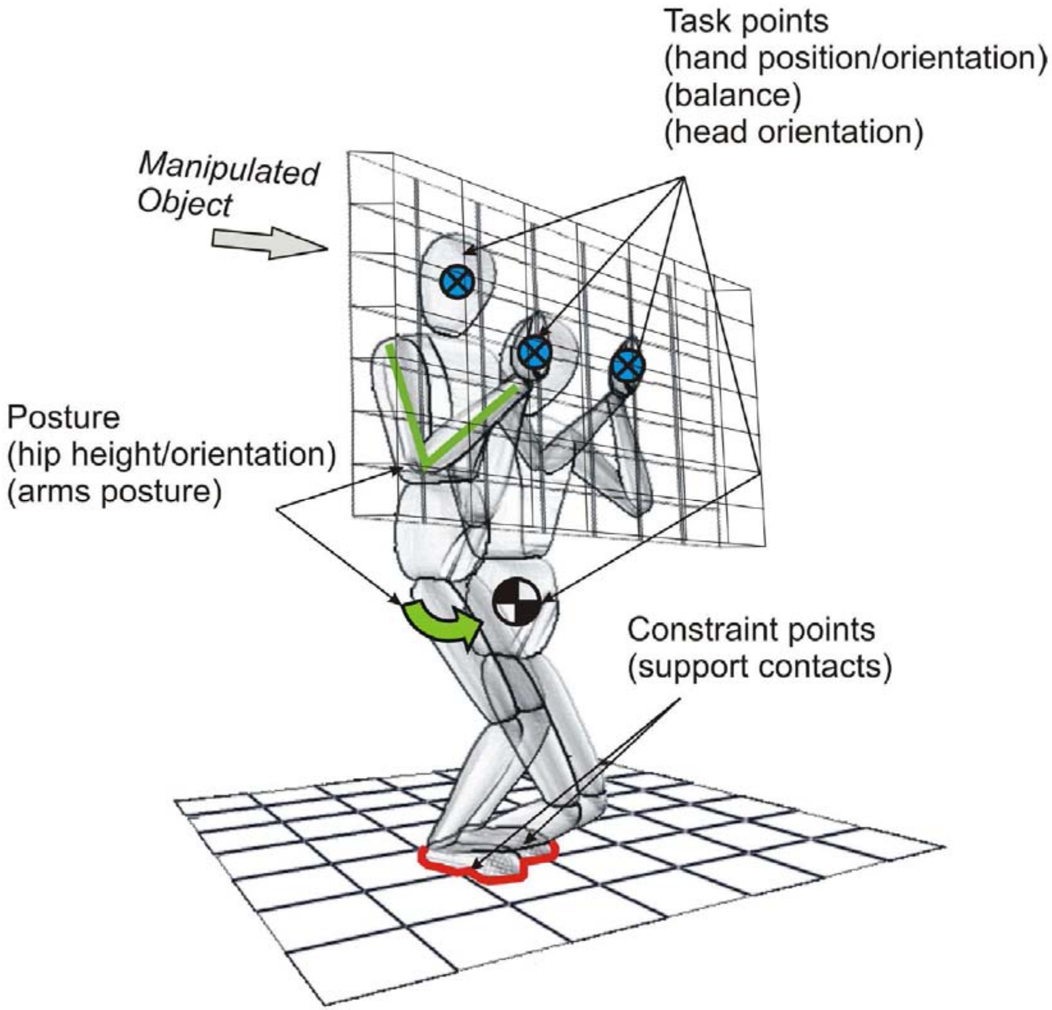
end for

Use a **numeric solver**

- e.g. quadprog, OOCP, …
- quadratic optimization
  - equality constraints
  - inequality constraints

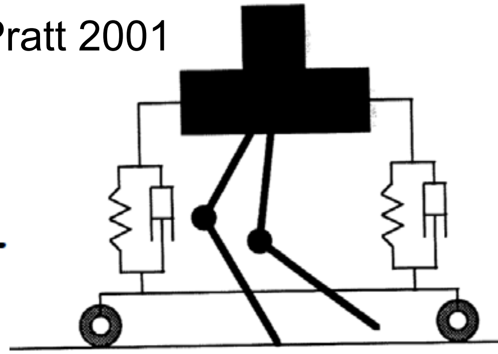
Behavior as Multiple Tasks

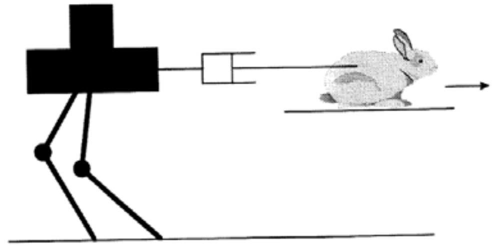




Quasi-static: Virtual Model Control

Pratt 2001





- No dynamic effects $\mathbf{M}(\cancel{\mathbf{q}})\dot{\mathbf{u}} + \mathbf{b}(\cancel{\mathbf{q}, \mathbf{u}}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T \mathbf{F}_c = \mathbf{S}^T \boldsymbol{\tau}$
- Add virtual external forces to pull/support the robot
- Static equilibrium of forces and moments
  - From principle of virtual work it follows directly that

$$0 = \sum_i \mathbf{F}_{p_i}, \qquad 0 = \sum_i \mathbf{r}_{bp_i} \times \mathbf{F}_{p_i} \quad \Rightarrow \quad \begin{pmatrix} \mathbf{F}_{c_1} \\ \vdots \\ \mathbf{F}_{c_{n_c}} \end{pmatrix} = \begin{bmatrix} \mathbb{I} & \cdots & \mathbb{I} \\ [\mathbf{r}_{c_1}]_\times & \cdots & [\mathbf{r}_{c_{n_c}}]_\times \end{bmatrix}^+ \begin{bmatrix} \sum \mathbf{F}_{g_i} - \sum \mathbf{F}_{vi} \\ \sum \mathbf{r}_{g_i} \times \mathbf{F}_{g_i} - \sum \mathbf{r}_{vi} \times \mathbf{F}_{vi} \end{bmatrix}$$

$$0 = \boldsymbol{\tau} + \sum_i \mathbf{J}_{bp_i}^T \mathbf{F}_{p_i} \quad \Rightarrow \quad \boldsymbol{\tau} = -\sum_i \mathbf{J}_{bg_i}^T \mathbf{F}_{g_i} + \sum_i \mathbf{J}_{bv_i}^T \mathbf{F}_{v_i} + \sum_i \mathbf{J}_{bc_i}^T \mathbf{F}_{c_i}$$

Next Time

- Application of this technique for locomotion control of legged robots