G0-Robot Control

PID for manipulators

Control of Robots

- What do we mean by “Control of Robots”?
  - Analysis and synthesis of the robot’s **control laws**

- What is a robot control law?
  - A control law finds the time behavior of the forces and torques to be delivered by the joint actuators to ensure the execution of a desired task

- Robot control tasks can be described as “high-level” commands (e.g., move from A to B, change the position of an object, detect an object with a camera)

- Each high-level description of the control task, can be expressed by means of lower-level (simpler) sub–tasks
  - In material handling tasks, it is sufficient to assign only the pick-up and release locations of an object (point–to–point motion)
  - In machining tasks, instead, the end–effector has to follow a desired trajectory (path motion). In such a case, the trajectory planner is generating the timing laws for the relevant variables (joint or end–effector indifferently)
  - Teleoperated robots, i.e., a robot remotely controlled by humans using a kind of joystick, fall into the case of trajectory following.

Hierarchical internal structure

Different levels of robot control:

- successfully complete a task or work program (load a truck, inspect a building)
- accurate execution of a motion trajectory (go from A to B through a path)
- zeroing a positioning error (go from A to B)

More intelligence

Decision making

High-level control

Plan trajectories

Maintain balance

Motor control

Low-level control

Fast and accurate

Direct control of actuators

- Different models can be exploited at the various control layers
- Pure planning (open loop) is not enough in face of **disturbances**

Evaluation of control performance

Quality of execution:

- Accuracy
- Velocity of task execution
- Energy requirements

Models can improve these!

Robustness in perturbed/uncertain conditions:

- Change of parameters (e.g., changes in payload)
- Modeling errors (e.g., a simpler model was used to described the plant)
- Uncertainties (e.g., imperfections in measuring devices)
- External disturbances (e.g., a push)

Control schemes and uncertainty

- **Classic feedback control:** tolerates mild disturbances and small parameter variation

- **Robust control**: tolerates large uncertainties in a known range

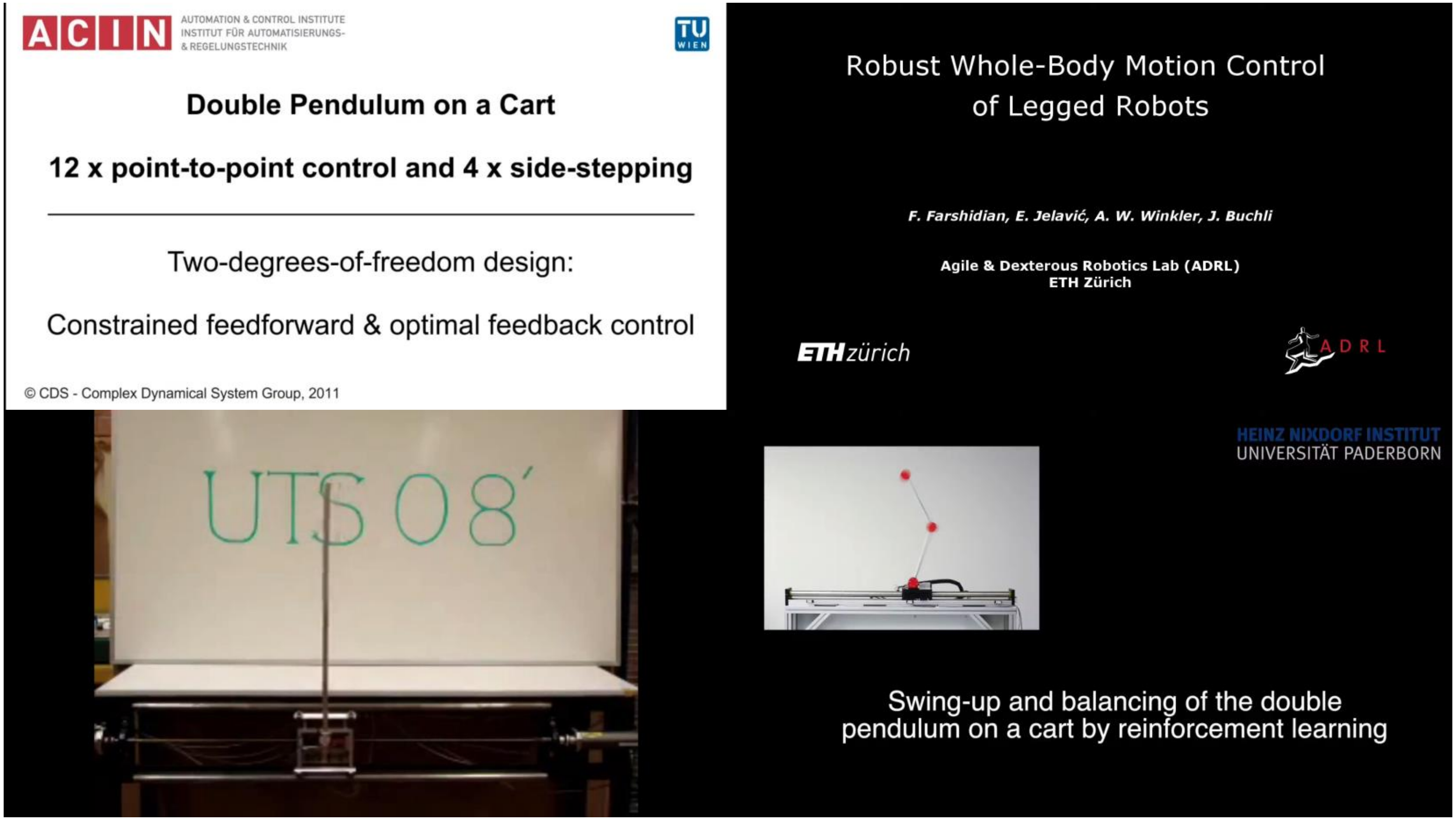
- **Adaptive control:** The control law has no fixed structure but is adaptable to an unknown range of uncertainties

- **Intelligent control/learning**: the control structure is independent from the physic domain (e.g., neural network). It includes tasks you have never seen. There is no **formal** guarantee of performance and may require a lot of data.

Control schemes and uncertainty

Feedback

Robust



Adaptive

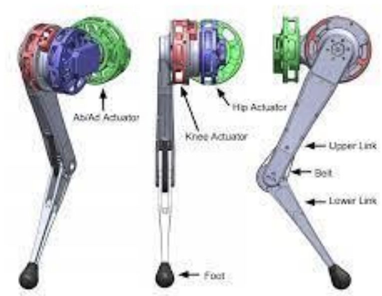
Learning

Challenges in control of industrial robots

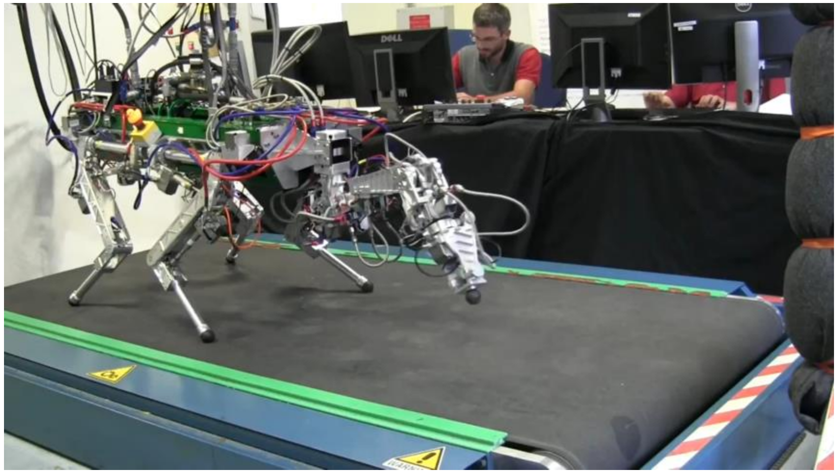
- Hard real-time operation
- Low-rate reading in exteroceptive sensors (i.e., vision)
- Dynamic accuracy on fast motion (difficult to track force/torque)
- Compliance (in the structure/ transmissions)
  - Heavy structure (resonance increases, no longer excited) -> limited speed
  - Inserted on purpose, for human-robot interaction
- Dry friction and backlash
  - Generally solved by using direct-drive actuators

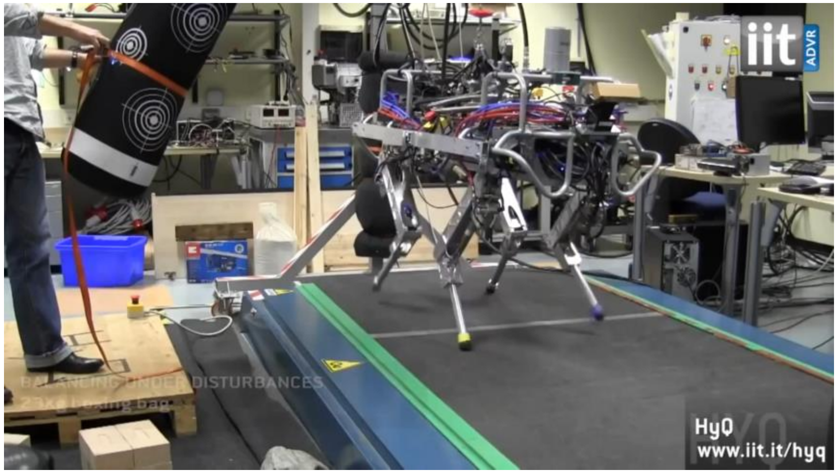
Take home messages

- A good hardware design is better than a fancy control

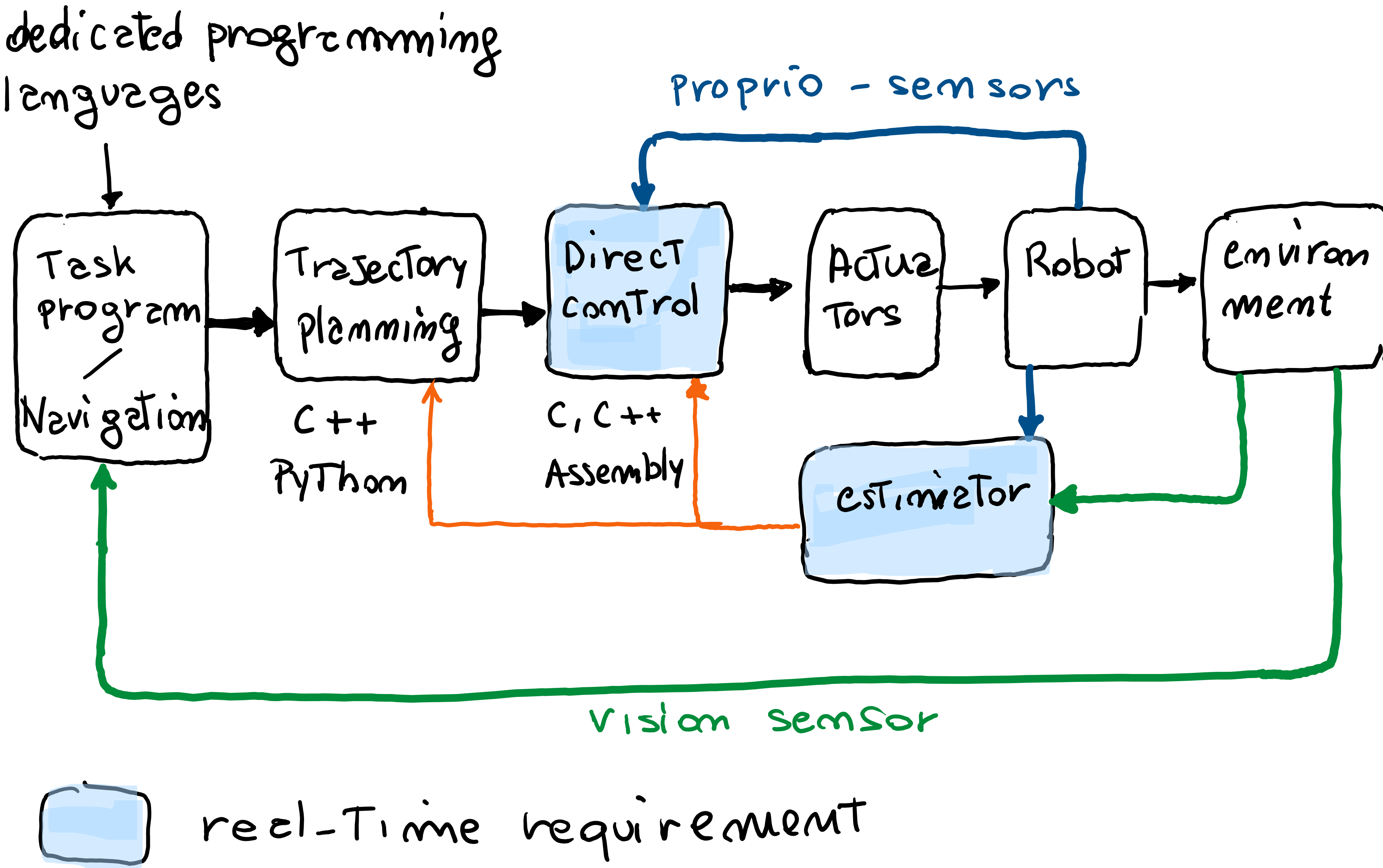


- Models can improve control performance
- Feedback is used to reject disturbances





Control pipeline block diagram



Software

- **Middleware** :
  - *Robot Operating System (ROS)*
  - Yet Another Robot Platform (YARP)
  - Open Robot Control Software (Orocos)

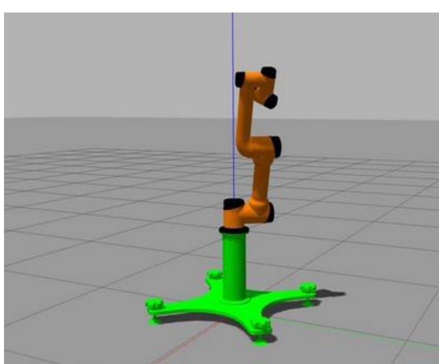
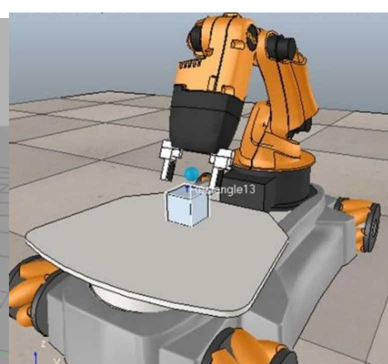
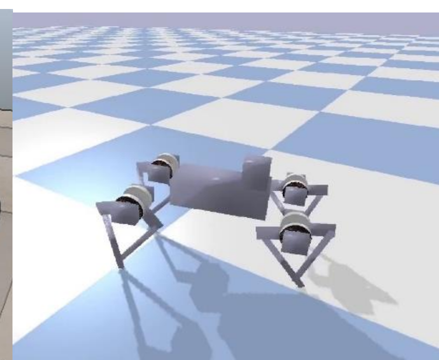
- **Rigid body kinematics and dynamics libraries**:
  - Rigid Body Dynamics Library (RBDL)
  - Kinematics and Dynamics Library (KDL)
  - *Robotics Code Generator (RobCoGen)*
  - *Pinocchio*

- **Drivers**: APIs (application programming interfaces) to communicate with sensors/actuators (usually C++ implementation, most efficient)

Software

- **Simulators**:
  - *Gazebo*
  - CoppeliaSim (former V-REP)
  - PyBullet
  - MuJoCo

- **Physics engines**:
  - Open Dynamics Engine (ODE)
  - Bullet
  - MuJoCo


PANORAMIC VIEW

assume control commands are always joint torques

type of task \ definition of error		JOINT SPACE (ref. desired configuration)	TASK SPACE (reference desired pose)
free motion	Regulator (initial/final)	P, PD, PID gravity compensation	PD + gravity compensation
	Traj. Tracking	feedback linearization (JSID)	feedback linearization (TSID)
motion in contact			impedance / admittance control (with variants)

↓ constraint motion
can exchange forces with environment

EQUILIBRIUM STATE OF A ROBOT

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

if we organize the dynamics equation in a state-space form:

$$x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \Rightarrow \dot{x} = \begin{bmatrix} x_2 \\ -M^{-1}(x_1)\left[R(x_1, x_2)\right] \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}(x_1) \end{bmatrix} u$$

$\Downarrow$

2n non-linear 1° order differential equations (ODE)

$$\dot{x} = \underbrace{f_x(x_1, x_2)}_{\text{drift term}} + \underbrace{f_u(x_1)}_{\text{forcing term}} u$$

(A) $\bar{x}$ unforced equilibrium $\rightarrow f_u(\bar{x})u = 0$
(no command $u = 0$)

$$\begin{cases} \boxed{\bar{x}_2 = 0} \\ -M^{-1}(\bar{x}_1)^{-1} R(\bar{x}_1, 0) = 0 \end{cases} \quad \boxed{g(\bar{x}_1) = 0}$$

gravity is balanced by structure

(B) $\bar{x}$ forced equilibrium $u = u(x)$

$$\Rightarrow \begin{cases} \boxed{\bar{x}_2 = 0} \\ -M^{-1} \underbrace{h(\bar{x}_1, 0)}_{g(\bar{x}_1)} + M^{-1} u(\bar{x}) = 0 \end{cases} \Rightarrow \boxed{u(\bar{x}) = g(\bar{x}_1)}$$

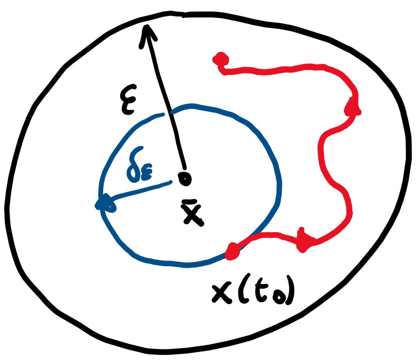
Joint Torques balance gravity

STABILITY

an equilibrium point $\bar{x}$ is stable if:

$\forall \varepsilon > 0 \quad \exists \delta_\varepsilon > 0 \; : \; \| x(t_0) - \bar{x} \| < \delta_\varepsilon \Rightarrow \| x(t) - \bar{x} \| < \varepsilon \quad \forall t > t_0$

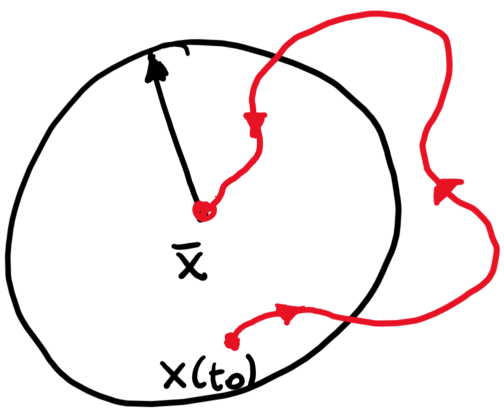
$\Rightarrow$ small perturbations from equilibrium will remain bounded (neighbourhood of $\bar{x}$)

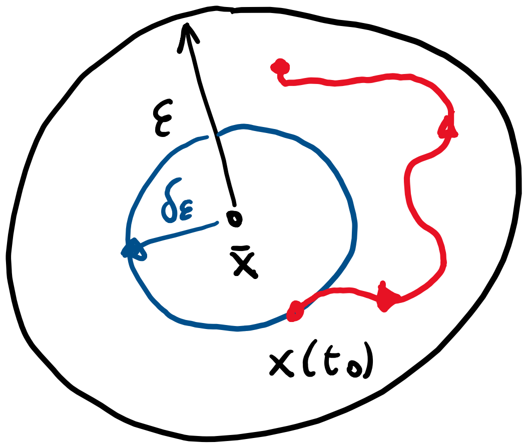


ASYMPTOTIC STABILITY

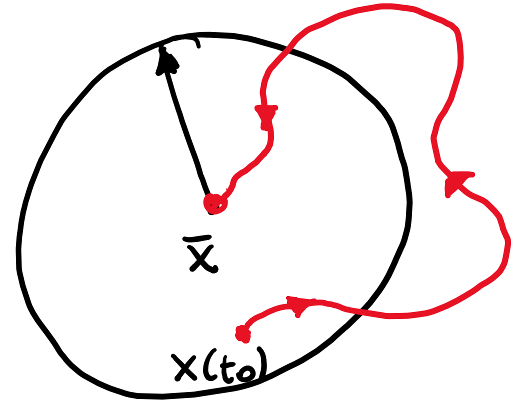
$\exists\, \delta > 0 : \| x(t_0) - \bar{x} \| < \delta \implies \| x(t) - \bar{x} \| \xrightarrow[t \to \infty]{} 0$ asymptotic stability

↳ $\forall\, \delta > 0$ global asymptotic stability





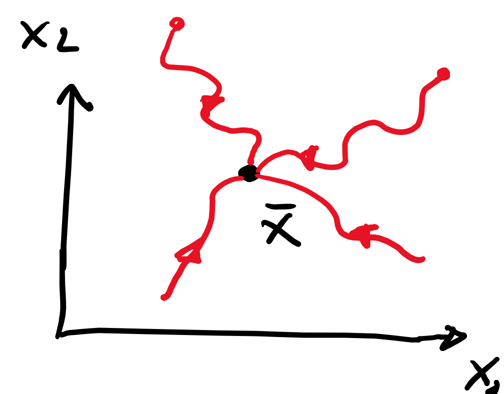
Stability



Asymptotic stability

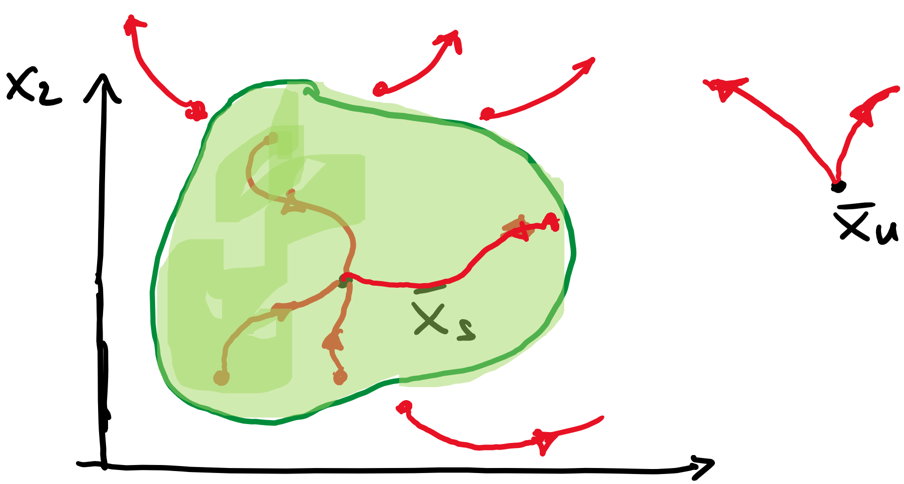
STABILITY OF LINEAR SYSTEMS

- $\dot{x} = Ax$
- equilibrium point is UNIQUE
- if eigenvalues of A have $Re(\lambda) < 0$
  The equilibrium is globally asymptotically stable



STABILITY OF NON-LINEAR SYSTEMS

- non-linear systems can have more than one equilibrium point (eg. pendulum) stable / unstable
- for the stable ones the basin of attraction is the set of initial conditions for which the state trajectory converges to $\bar{x}$.



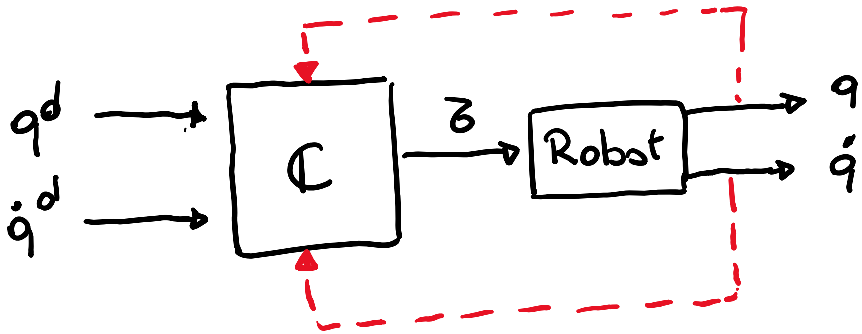
PANORAMIC VIEW

assume control commands are always Joint Torques

definition / type of error / of Task		JOINT SPACE (ref. desired configuration)	TASK SPACE (reference desired pose)
free motion	Regulator (initial/final)	P, PD, PID gravity compensation	PD + gravity compensation
	Traj. Tracking	feed back linearization (JSID)	feedback linearization (TSID)
motion in contact			impedance / admittance control (with variants)

PD CONTROL

Proportional + derivative action



--- feedback

INPUTS:

- Joint reference trajectories: $q^d(t)$, $\dot{q}^d(t)$
  - ↳ Regulation problem: $q^d = \text{const}$, $\dot{q}^d = \emptyset$
- Join positions/velocities (sensor measurements)

CONTROL LAW:

$$\boxed{u(t) = K_p\,(q^d(t) - q(t)) + K_d\,(\dot{q}^d(t) - \dot{q}(t))}$$

$q \in \mathbb{R}^n$

$K_p, K_d \in \mathbb{R}^{n \times n}$

$K_p > 0$, $K_d > 0$

pos. def.
gain
matrix

if we replace in dynamics

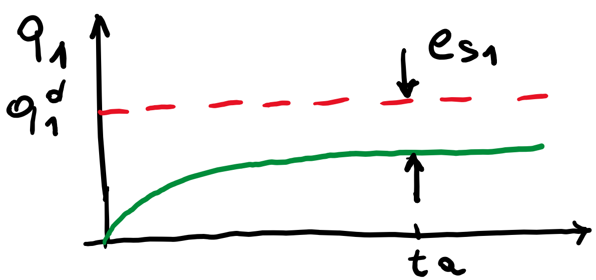
$$M(q)\,\ddot{q} + c(q, \dot{q}) + g(q) = u$$

at equilibrium $(\dot q = 0, \ddot q = 0)$ with constant reference $(q^d = \text{const}, \dot q^d = 0)$

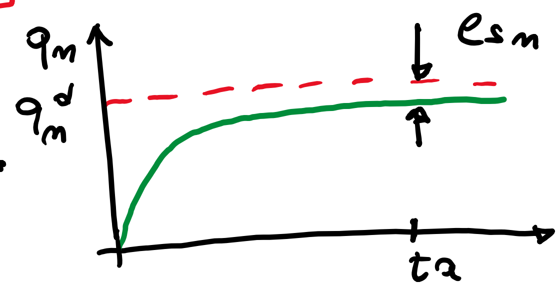
$$\cancel{M\ddot q} + c(\cancel{q, \dot q}) + g(q) = K_p(q^d - q) - \cancel{K_d(\dot q^d - \dot q)}$$

steady state error e_s at equilibrium:

$$\boxed{q^d - q = e_s = K_p^{-1}\, g(q)}$$



...



distal joint has smaller error because it supports less weight

- if we chose K_p, K_d positive definite the equilibrium is asymptotically stable (if gravity $= \emptyset$ is globally a. s.)
- Typically

$K_p = \text{diag}(K_{pi})$ $\quad\quad$ $K_d = \text{diag}(K_{di})$

$\Rightarrow$ decentralized linear control (local to each joint)

COMMENTS ON PD CONTROL

- gains K_P, K_d affect the robot evolution during transient and the settling time t_a
- When viscous friction is present the derivative action is not strictly necessary:

  $$-F_v \dot{q} \simeq -K_d \dot{q}$$

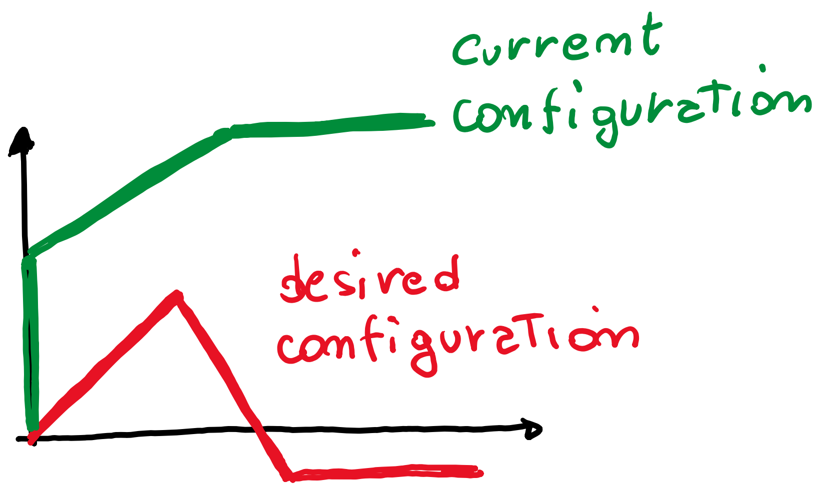
⊖ hard to define optimal values of K_P, K_d for the whole workspace.

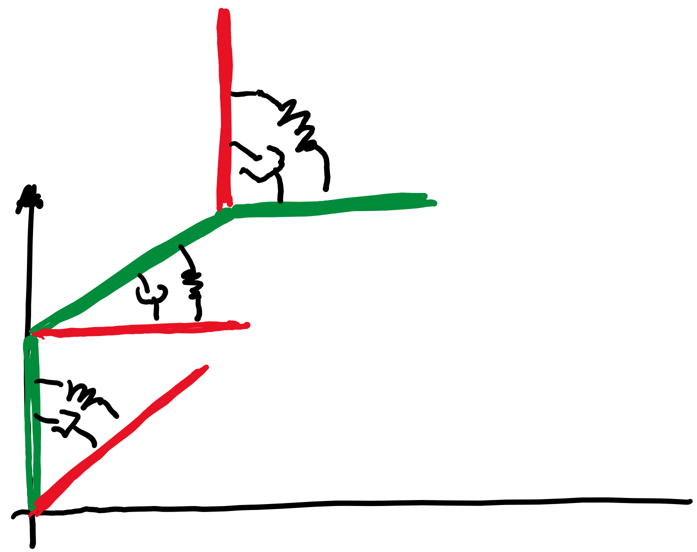
MECHANICAL INTERPRETATION

K_p, K_d correspond to stiffness of virtual springs and viscosity of virtual dampers.

—/\/\/\— stiffness $K_{P_i} > 0$

—]— damping $K_{d_i} > 0$





NB q_i^d is defined relative to the previous link

PD + GRAVITY COMPENSATION

- Idea: add a non-linear cancellation of gravity in the control law

$$\boxed{u = K_p(q^d - q) + K_d(\dot{q}^d - \dot{q}) + g(q)}$$

at equilibrium $\dot{q}, \ddot{q} = 0$

$$\cancel{M\ddot{q}} + \cancel{C(q,\dot{q})} + \cancel{g(q)} = K_p(q^d - q) + \cancel{K_d(\dot{q}^d} - \dot{q}) + \cancel{g(q)}$$

$$K_p e_s = 0 \quad \Rightarrow \quad \boxed{e_s = 0}$$

⊕ at equilibrium $q \to q^d$, $\dot{q} \to 0$, no steady-state error

⊕ global asymptotic stability (ensured if $K_p > 0$, $K_d > 0$)

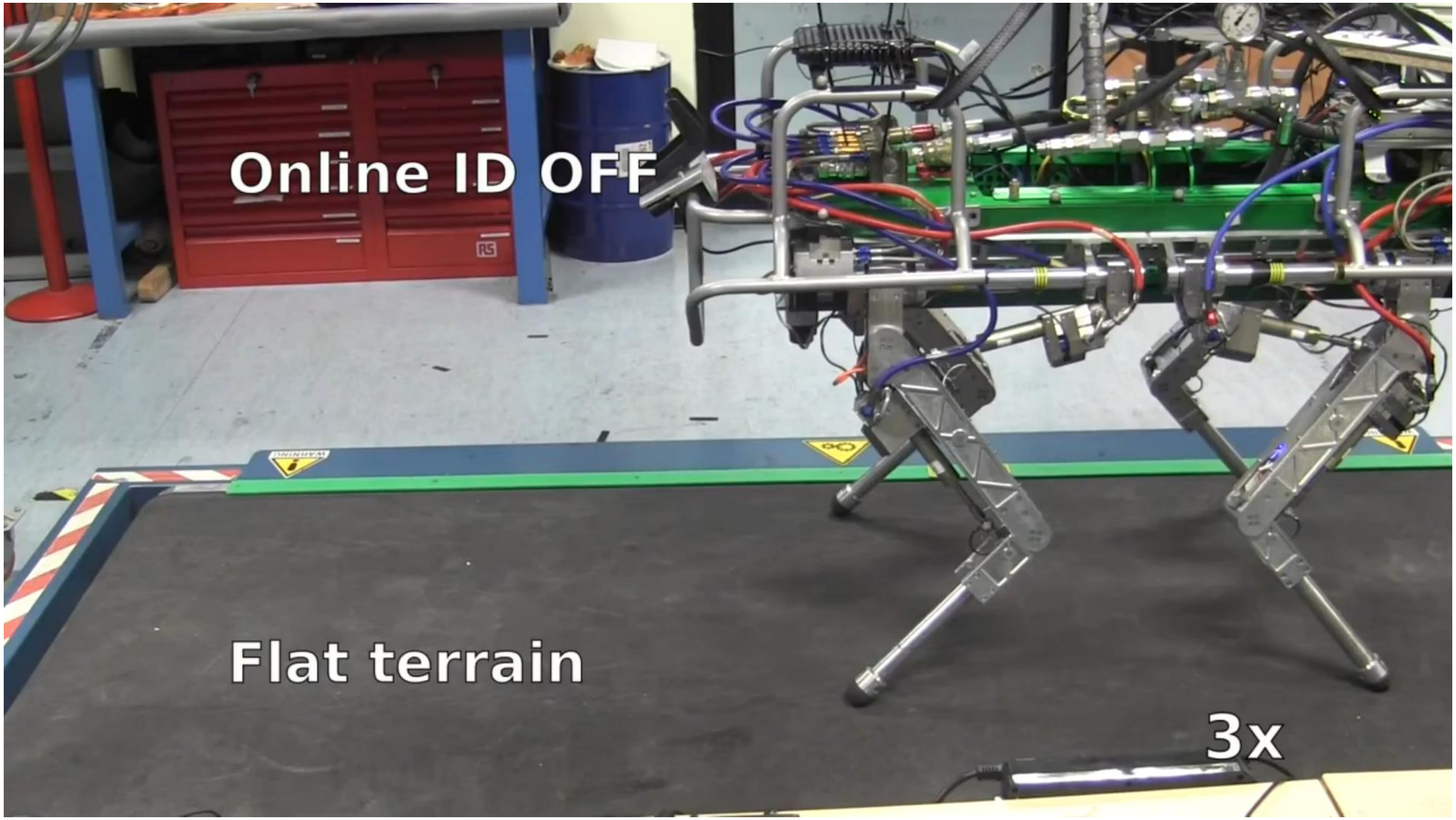
⊖ we need to identify $g(q)$ (what about unknown payload?)

⊖ if $g(q)$ is cancelled approximately the steady state error is not zero and $q \to q^* \neq q^d$

$\Rightarrow$ if you increase k_p, $q \to q^d$

$\Rightarrow$ in real system $k_p \uparrow\uparrow \Rightarrow$ instabilities

Application example



PID CONTROL

- we can see gravity as a constant disturbance acting on the system
- in linear systems the addition of an integral control action is used to eliminate a constant error in the face of unknown constant disturbances (i.e. inaccurate / absent gravity compensation / unknown payloads)

CONTROL LAW: PID

$$\boxed{u(t) = K_p\,(q^d(t) - q(t)) + K_d\,(\dot q^d + \dot q) + \underbrace{K_i \int_0^t (q^d - q)\, d\tau}_{u_I}}$$

The integral component grows magnifying the error till the disturbance is not compensated

$e(t) > 0 \qquad u_I \nearrow$

$e(t) < 0 \qquad u_I \searrow$

$u_I \propto$ mean value of e

at equilibrium $\dot{q}, \ddot{q} = 0$ we will have

$$K_I \int_0^t (q^d - q)\, d\tau = g(q)$$

integral action compensates gravity

and $e_s = 0$

COMMENTS ON PID

⊕ is independent from any robot model

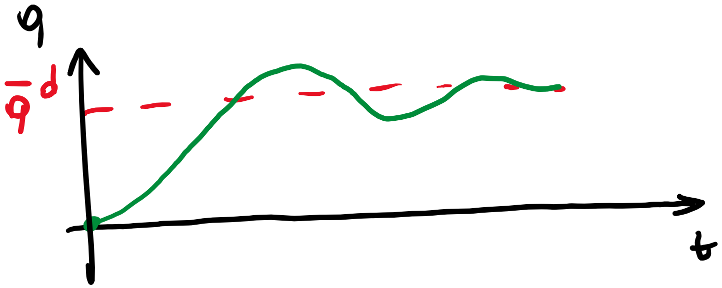
⊖ Tuning of gains requires experience / trial & error.

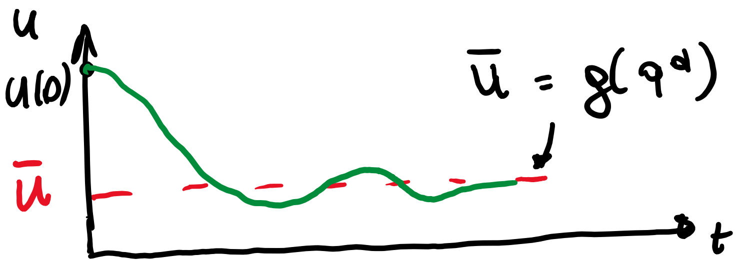
⊖ Anti-windup scheme is necessary to stop integration when the actuator saturates

⊖ Can create oscillations in presence of coulomb friction

REGULATORS VS TRAJECTORY PLANNING

$\bar{q}^d(t) = \text{const}, \quad \dot{q}^d = 0$

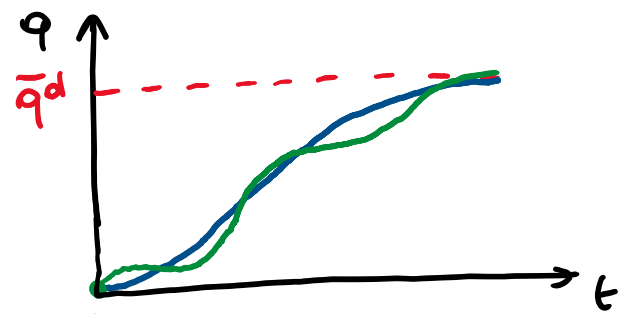


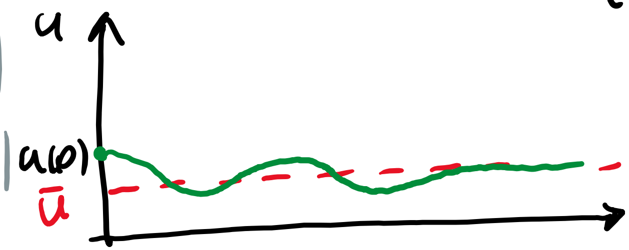


⊖ control effort can be high at $t=0$ (saturation)

with a planner that generates a time varying reference trajectory that interpolates actual to desired position

$q^d(0) = q(0), \quad q^d(T) = \bar{q}^d$





⊕ position error is ∅ at $t=0$

⊕ motion stays in the vicinity of ref. trajectory (can set larger gains)