



Robotics 2

Introduction to Control

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

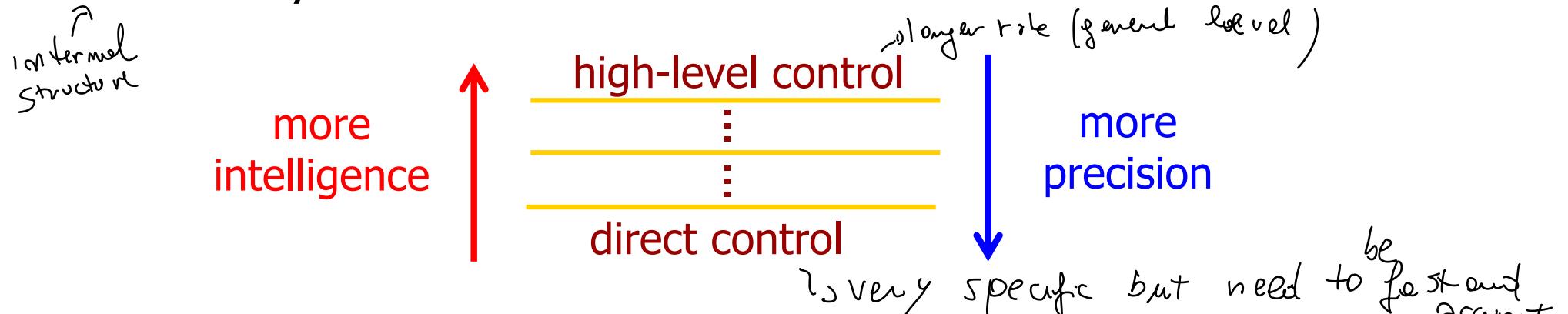




What do we mean by robot control?

- different level of definitions may be given to robot control
 - successfully complete a task or work program → usually solving subtask
 - accurate execution of a motion trajectory → bring to & position error at given time
 - zeroing a positioning error

⇒ control system unit has a hierarchical internal structure



- different time scales in the various control levels: lowest $\leq 1 \text{ ms}$, higher levels up to seconds
- different but cooperating models, objectives, methods



robot not ideal as we consider "prior"

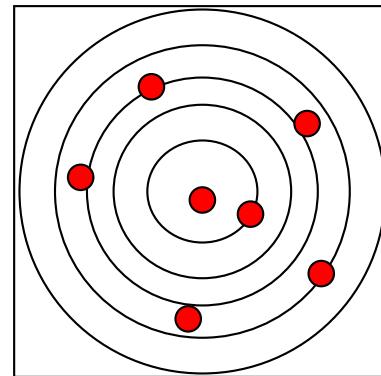
Evaluation of control performance

- **quality** of execution in nominal conditions
 - velocity/speed of task completion
 - accuracy/repeatability (in **static** and **dynamic** terms)
 - energy requirements
 - ⇒ improvements also thanks to **models** (software!)
- **robustness** in **perturbed/uncertain** conditions
 - adaptation to changing environments
 - high repeatability despite disturbances, changes of parameters,
uncertainties, modeling errors
 - ⇒ can be improved by a generalized use of **feedback**,
using more **sensor information**
 - ⇒ **learn** through repeated robot trials/human experience

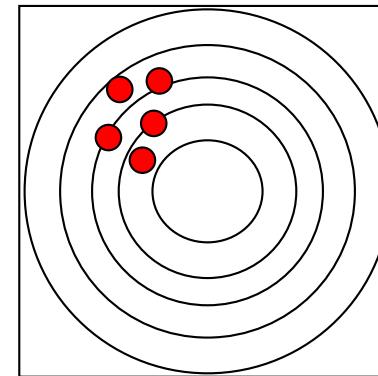


Static positioning accuracy and repeatability

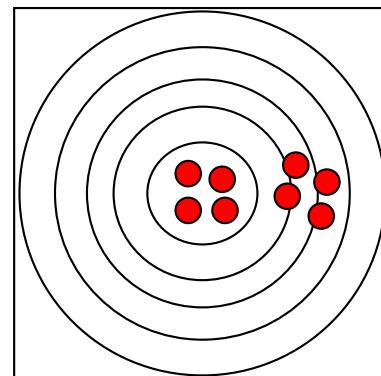
poor accuracy
poor repeatability



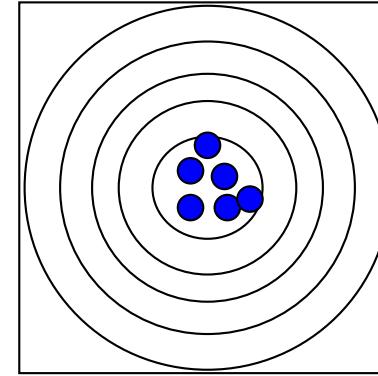
poor accuracy
good repeatability



good accuracy
poor repeatability



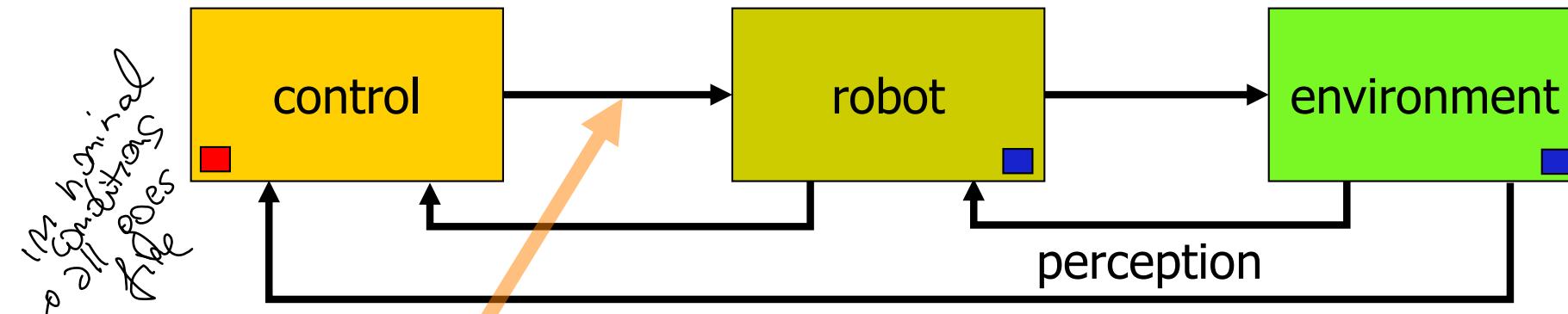
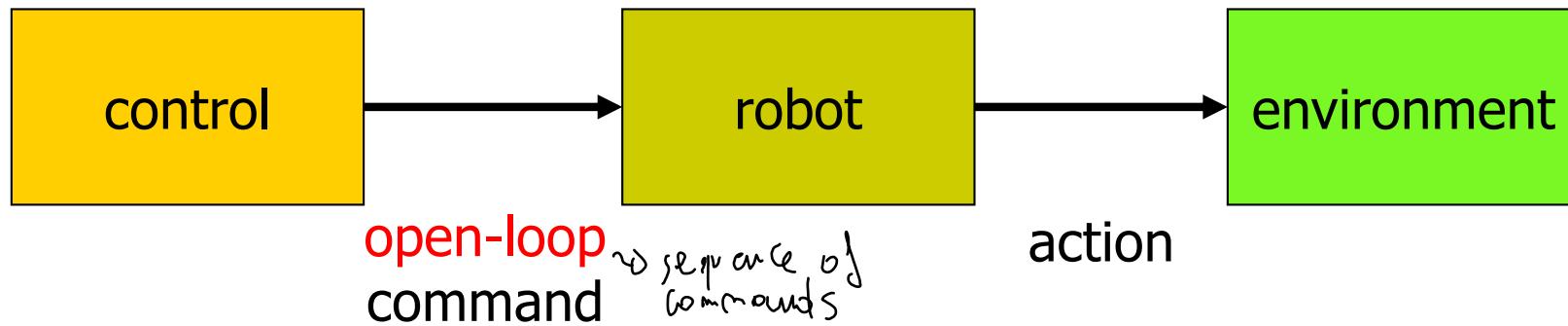
good accuracy
good repeatability



what about “dynamic” accuracy on (test or selected) motion trajectories?



Basic control schemes



compensate for error from ideal situation

best performance

robustness

or learned by experience

or learned & stored

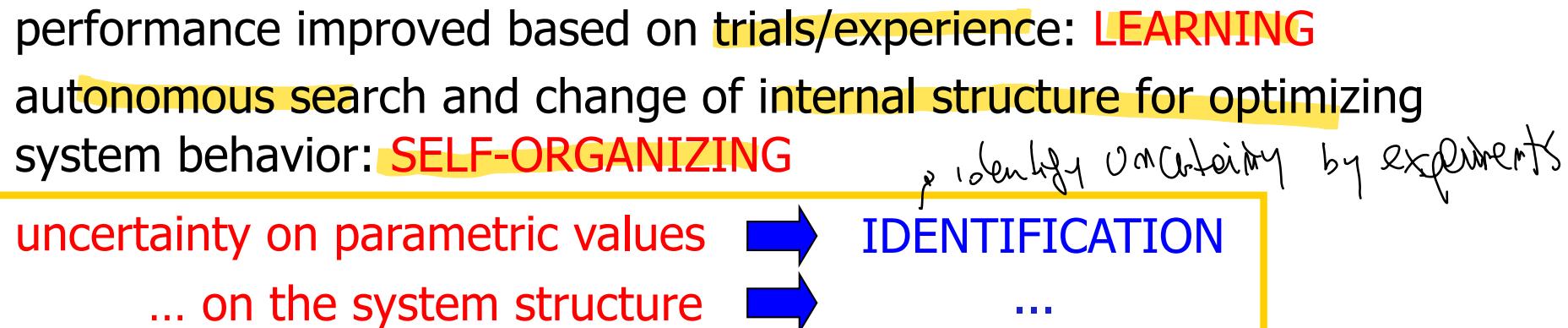
Kim, dynamic environment models

5



Control schemes and uncertainty

- **feedback control** ↪ driving the system from desired and current (error)
 - insensitivity to mild disturbances, small variations of parameters, and different initial conditions
- **robust control**
 - tolerates relatively large uncertainties of known range
- **adaptive control**
 - improves performance online, adapting the control law to unknown range of uncertainties and/or large (but slow) parameter variations
- **intelligent control**





Limits in control of industrial robots - 1

- from a **functional** viewpoint
 - “closed” control architectures, relatively difficult to interface with external programs and sensing devices for **hard real-time** operation
 - need of some expertise for programming and handling of exceptions
 - ⇒ introducing easy/more intuitive user (multi-modal) interfaces
- at the **higher** level
 - open-loop task command generation
 - ⇒ exteroceptive sensory feedback absent or very loose, with low capability of autonomous reasoning
- at the **intermediate** level
 - limited consideration of advanced kinematic and dynamic issues
 - ⇒ e.g., singularity robustness: solved on a case-by-case basis
 - ⇒ task redundancy: no automatic use of the extra degrees of freedom



Limits in control of industrial robots - 2

- at the **lower (direct)** level
 - reduced execution speed ("control bandwidth")
 - ⇒ typically, heavy mechanical structures
 - reduced dynamic accuracy on fast motion trajectories
 - ⇒ standard: use of kinematic control + PID only
 - problems with dry friction and backlash at the joints
 - compliance in the robot structure
 - ⇒ flexible transmissions
(belts, harmonic drives, long shafts)
 - ⇒ large structures or relatively lightweight links

now **desired**
for safe
physical
Human-Robot
Interaction

- need to include better **dynamic models** and model-based **control laws**
- handled, e.g., using **direct-drive** actuators or online friction **compensation**



Example of robot positioning

- low damped vibrations due to joint elasticity



without modeling
and explicit
control of
joint elasticity

- 6R KUKA KR-15/2 robot (235 kg), with 15 kg payload



We want to study
new better control laws

Advanced robot control laws

- deeper mathematical/physical analysis and modeling of robot components (**model-based approach**)
- schemes using various control loops at different/multiple hierarchical levels (**feedback**) and with additional sensors
 - visual servoing *for example using cameras*
 - force/torque sensors for interaction control
 - ...
- “new” methods
 - integration of (open-loop/feedforward) **motion planning** and **feedback control** aspects (e.g., sensor-based planning)
 - fast (sensor-based) re-planning
 - model predictive control (with preview)
 - **learning** (iterative, by imitation, skill transfer, ...)
 - ...



Example of visual-based control

- human-obstacle collision avoidance



video

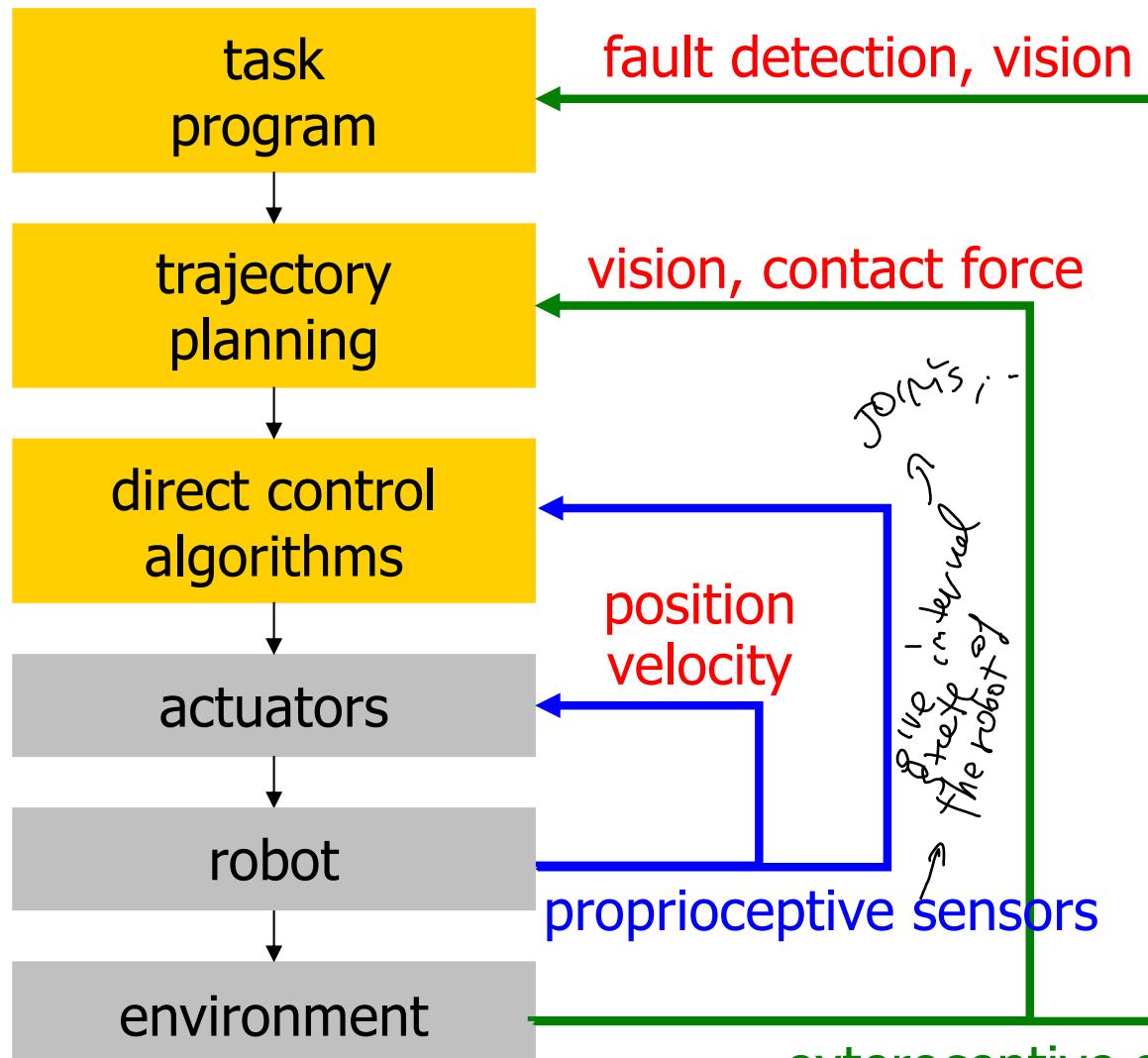
- 3R SoftArm prototype with McKibben actuators (Univ. of Pisa) using **repulsive force field** built from stereo camera information



Functional structure of a control unit

typical workflow:

sensor measurements



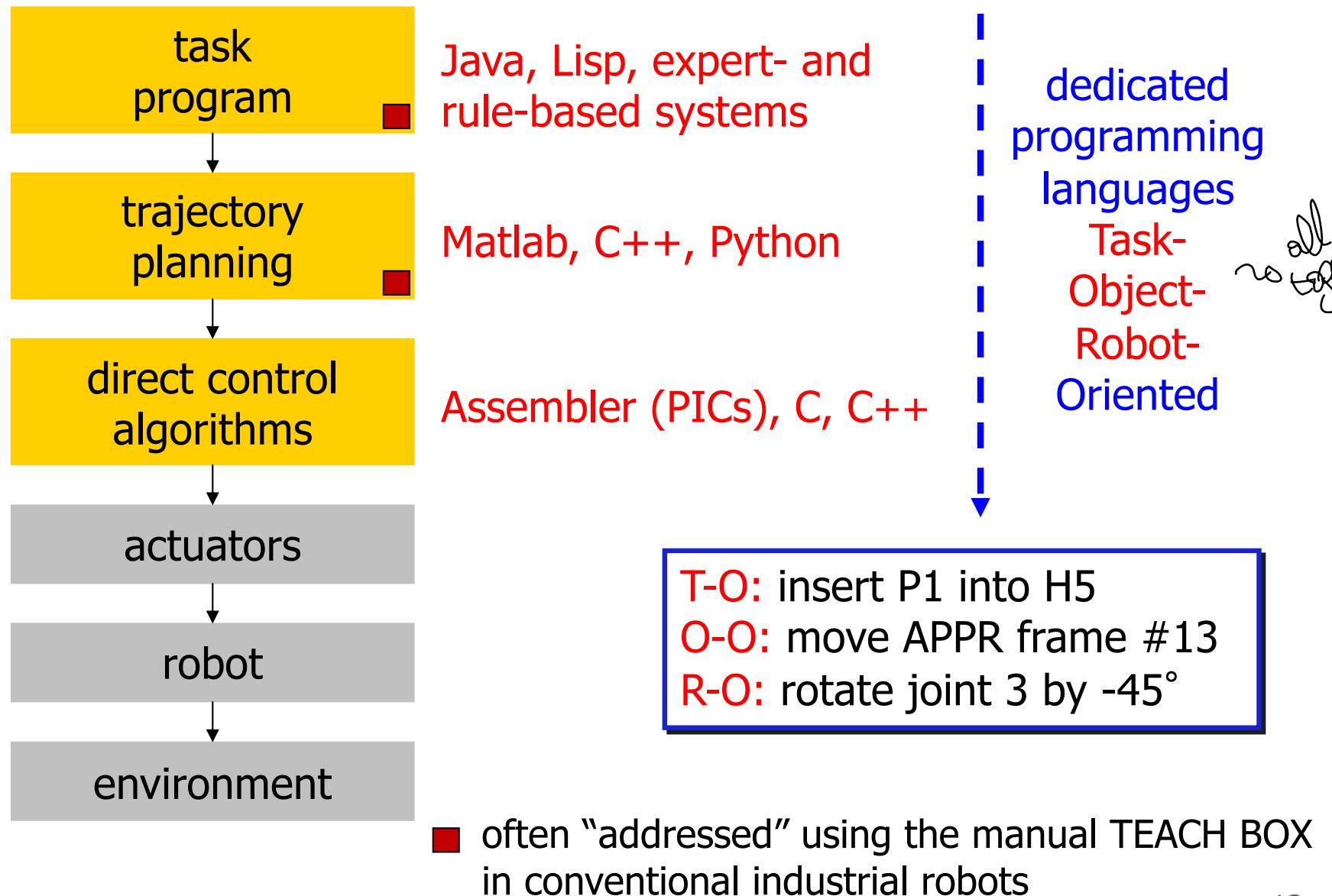
SENSORS:
optical encoders,
velocity tachos,
strain gauges,
joint or wrist
F/T sensors, IMUs,
tactile sensors,
micro-switches,
range/depth sensors,
laser, CCD/CMOS and
stereo cameras,
RGB-D cameras,

...



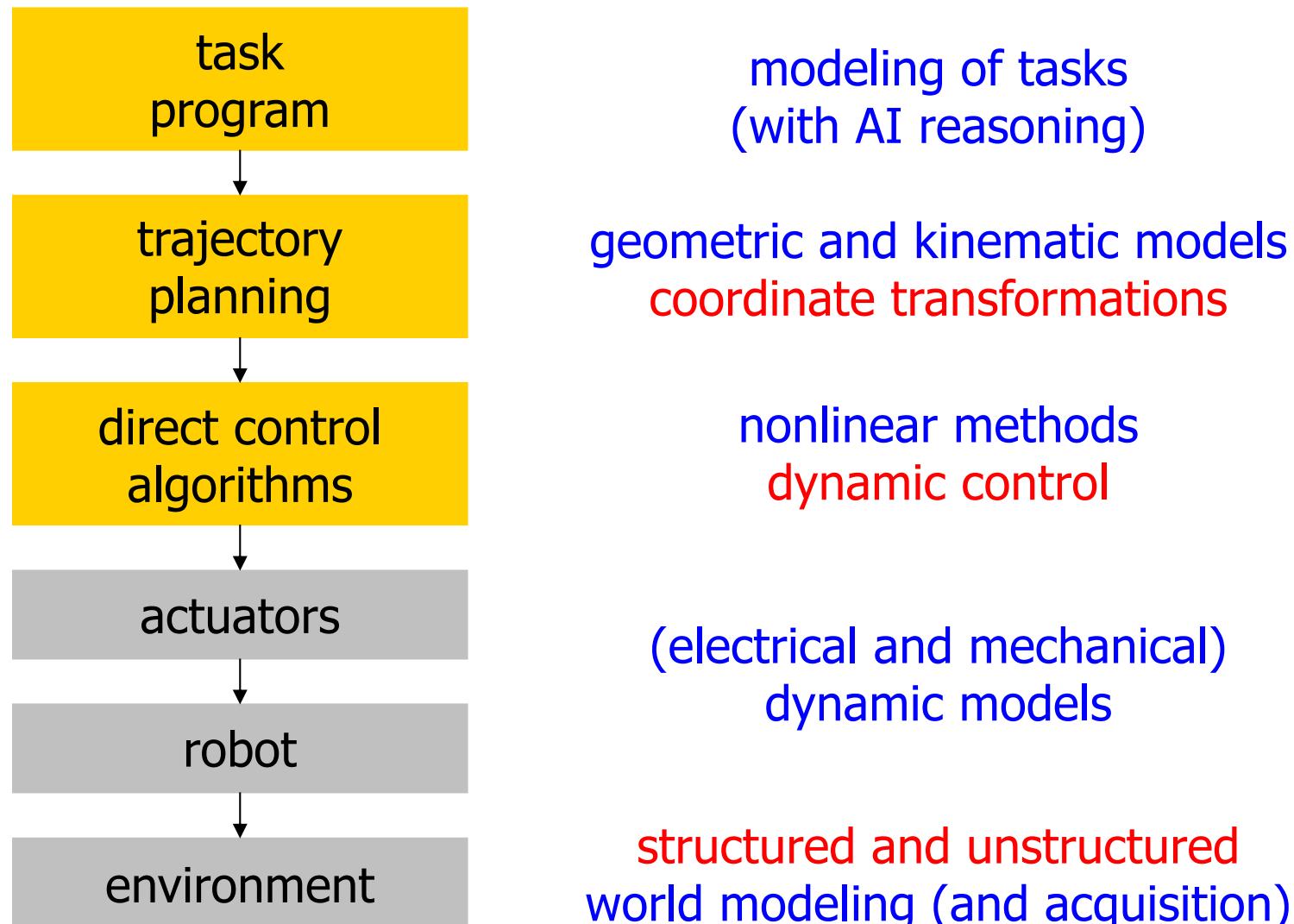
Functional structure of a control unit

programming languages



Functional structure of a control unit

modeling issues





Industrial robot programming languages

- ABB Rapid
- COMAU PDL2
- FANUC Karel
- KUKA KRL
- MITSUBISHI MELFA
- UNIVERSAL ROBOTS RoboDK
- ...

ABB



FANUC

KUKA

MITSUBISHI

 **UNIVERSAL ROBOTS**



Robot control/research software

(last updated in April 2024)

- a (partial) list of open source robot software
 - for simulation and/or real-time control
 - for interfacing with devices and sensors
 - research oriented

Player/Stage playerstage.sourceforge.net ⇒ github.com/rtv/stage

works
with
others

- **Stage:** in origin, a networked Linux/MacOS X robotics server acting as abstraction layer to support a variety of hardware ⇒ now a 2(.5)D mobile robot standalone simulation environment
- **Gazebo:** 3D robot simulator (**ODE** physics engine and **OpenGL** rendering), now an independent project ⇒ gazebosim.org



GAZEBO

CoppeliaSim (was V-REP; edu version available) www.coppeliarobotics.com

interfaces
with other
objects

- each object/model controlled via an embedded script, a plugin, a ROS node, a remote API client, or a custom solution
- controllers written in C/C++, Python, Java, Matlab, ...





Robot control/research software (cont'd)

Robotics System Toolbox (license for Sapienza)



- tools/algorithms for simulation of kinematics, dynamics, trajectory planning, control of serial manipulators, mobile robots and humanoids
- library of robots, scene and map creation, Gazebo interface ...

QUT Robot Academy petercorke.com



- free software for robotics and for vision; includes the Robotics Toolbox ([release 10](#)) and the Machine Vision Toolbox (release 4) for MATLAB

ROS (Robot Operating System) ros.org



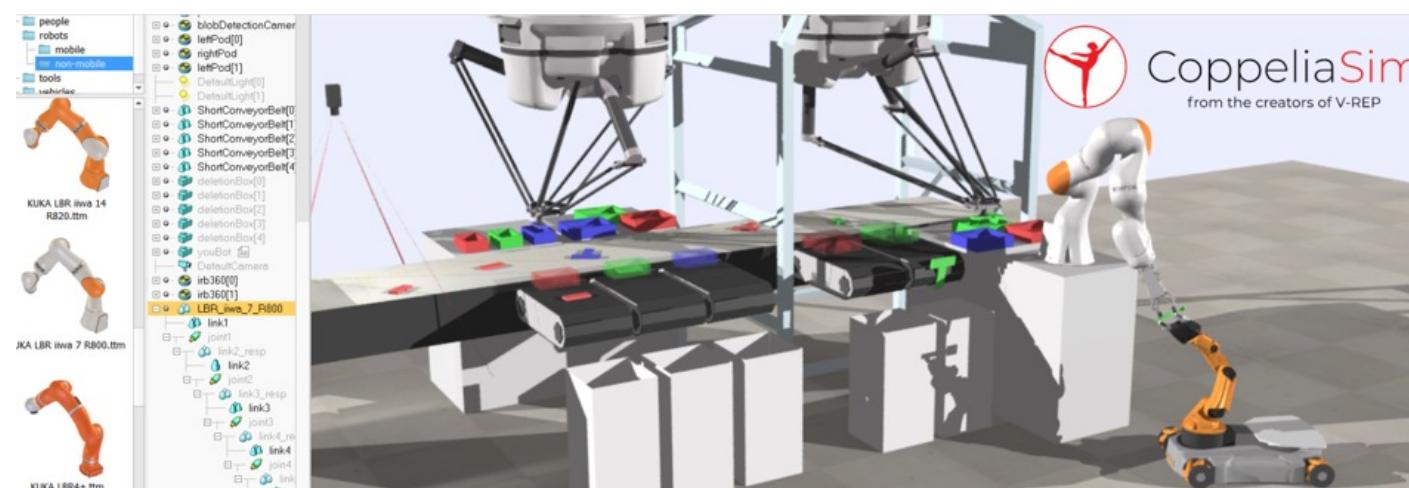
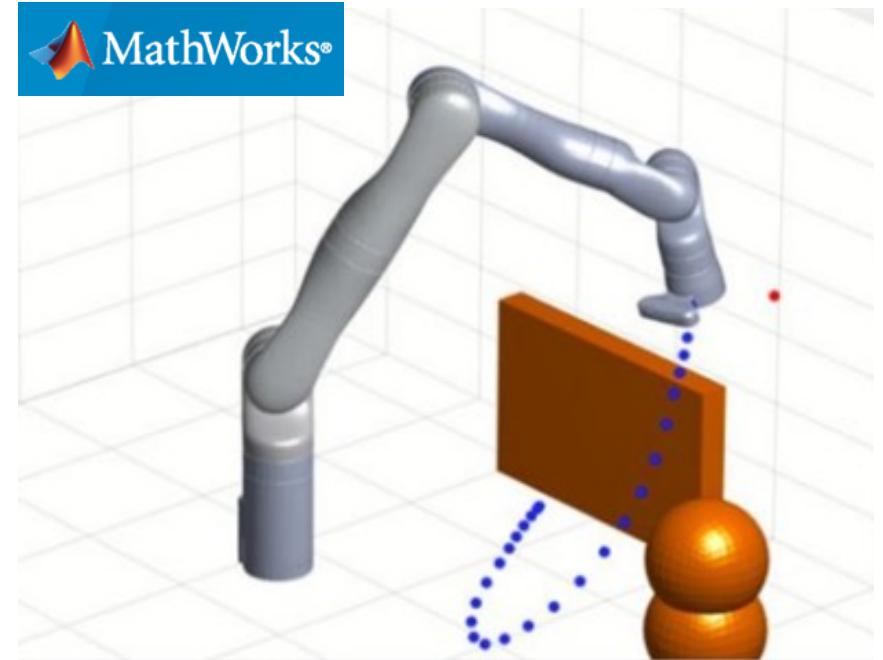
- **middleware** with hardware abstraction, device drivers, libraries, visualizers, message-passing, package management (now **ROS 2**)
- “nodes”: executable code (in Python, C++) running with a publish/subscribe communication style
- drivers, tools, state-of-the-art algorithms ... (all open source)

PyRobotics (Python API) pypi.org/project/pyRobotics (v1.8 in 2015)





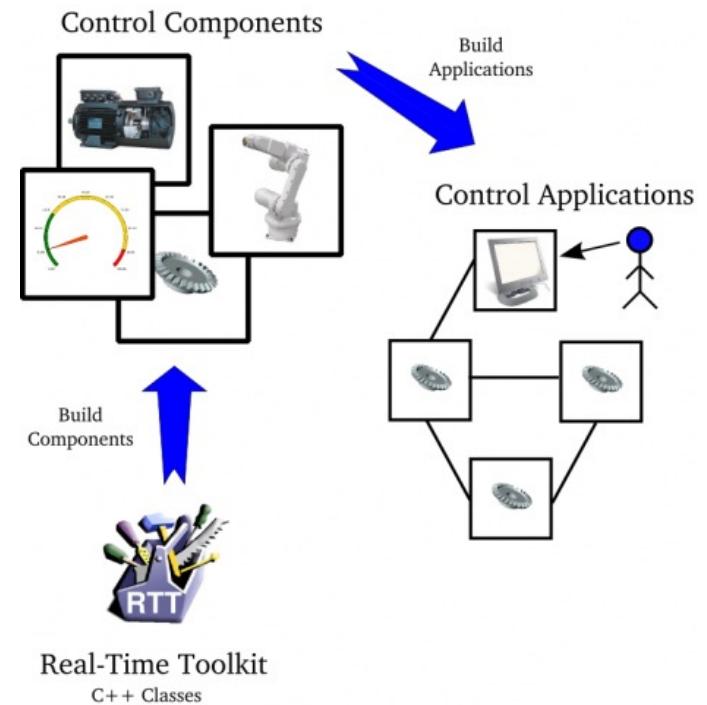
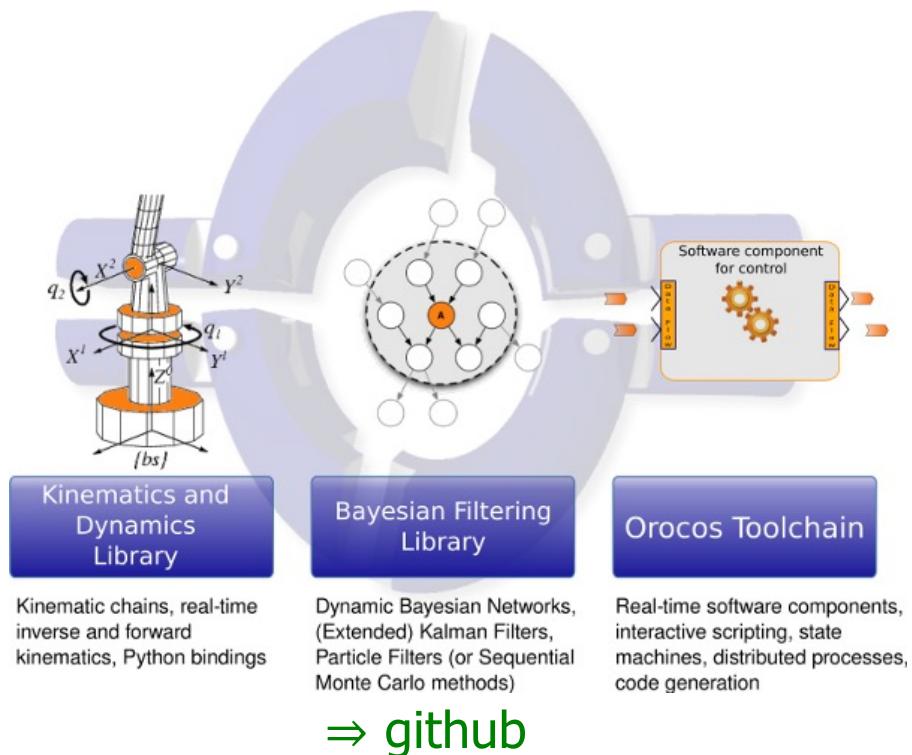
Robot control/research software





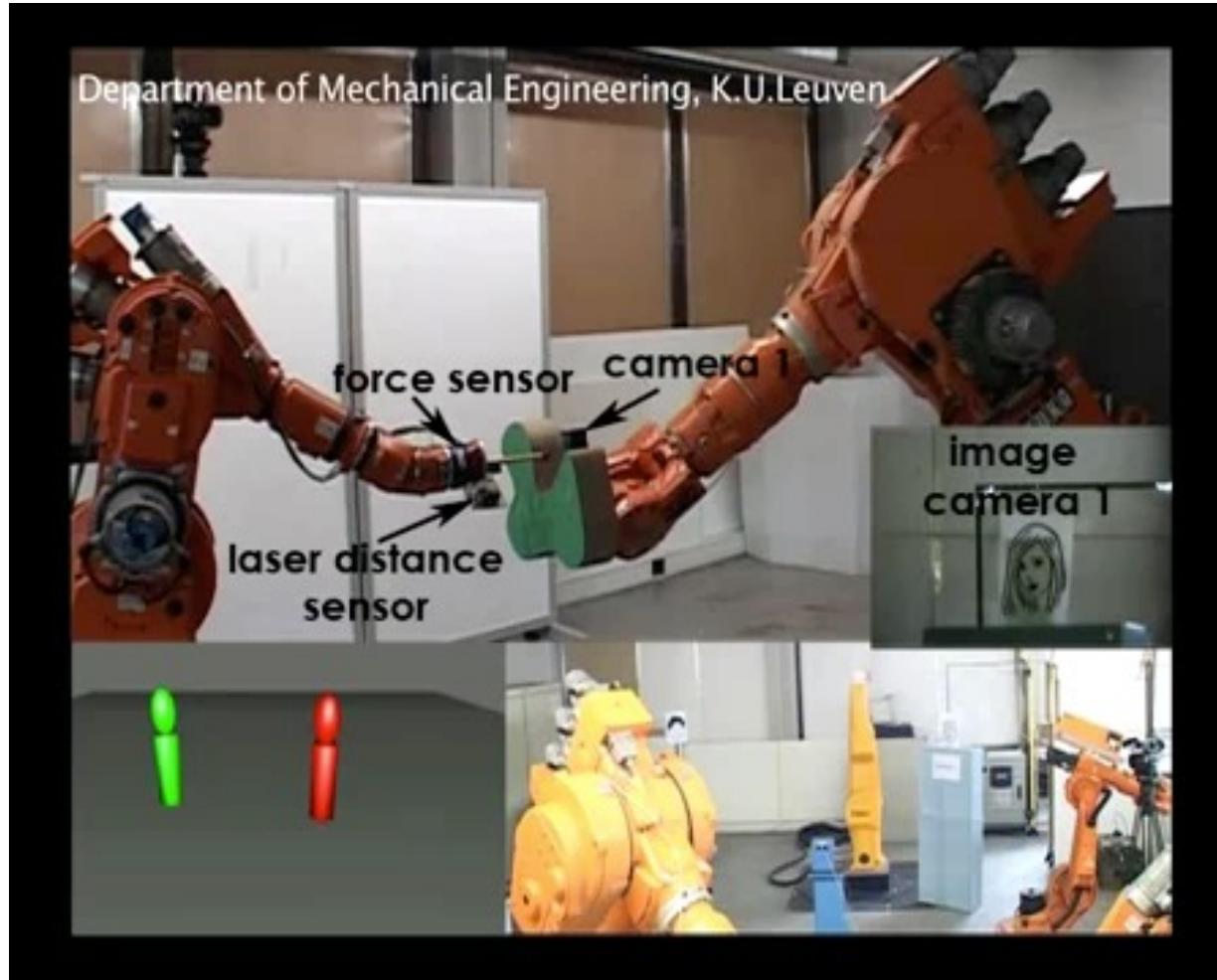
OROCOS control software

- OROCOS (Open RObot COntrol Software) orocos.org
 - open-source, portable C++ libraries for robot control
 - Real-Time Toolkit (for Linux, MacOS X, Windows Visual Studio)
 - supports CORBA for distributed network computing and ROS interface
 - (user-defined) application libraries





Example application using OROCOS



multi-sensor fusion for multi-robot manipulation
in a human populated environment (KU Leuven)



Summarizing ...

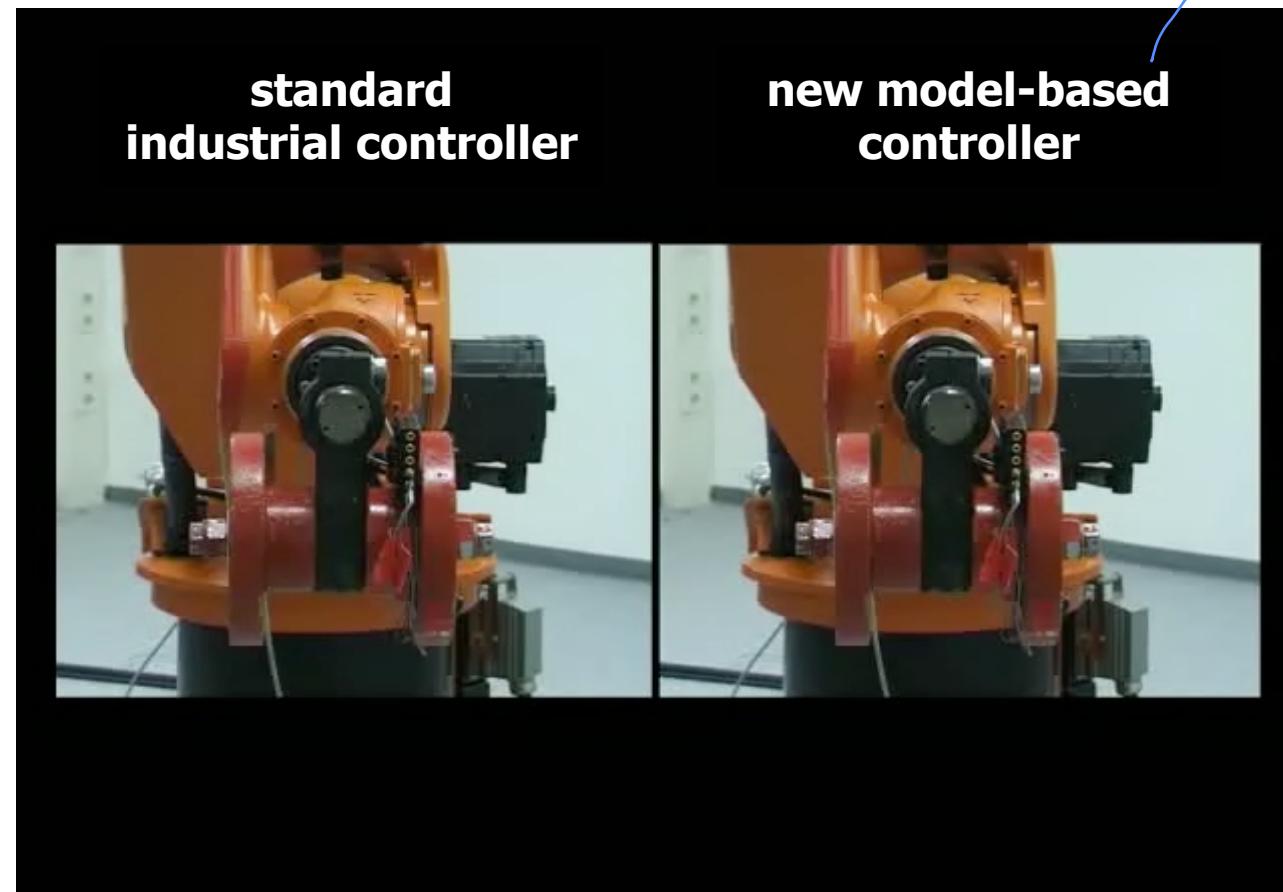
- to **improve performance** of robot controllers
 1. more complete **modeling** (kinematics and **dynamics**)
 2. introduction of **feedback** throughout all hierarchical levels
- **dynamic control** at low level allows in principle
 1. much **higher accuracy** on generic motion trajectories
 2. **larger velocity** in task execution with **same accuracy**
- interplay between **control, mechanics, electronics**
 1. able to control accurately also **lightweight/compliant** robots
 2. full utilization of task-related **redundancy**
 3. smart **mechanical design** can reduce control efforts (e.g., closed kinematic chains simplifying robot inertia matrix)
 4. **actuators** with higher dynamic performance (e.g., direct drives) and/or including controlled variable stiffness

advanced applications should justify additional costs
(e.g., laser cutting with 10g accelerations, safe human-robot interaction)

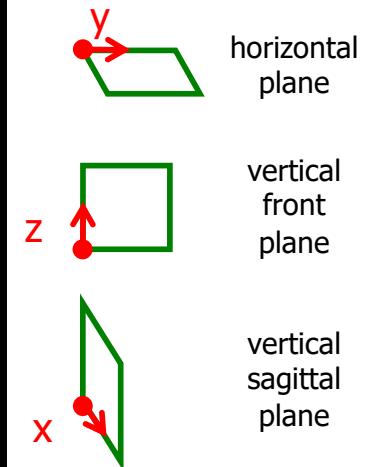


Benefits of model-based control

- trajectory tracking task: comparison between standard industrial and new model-based controller



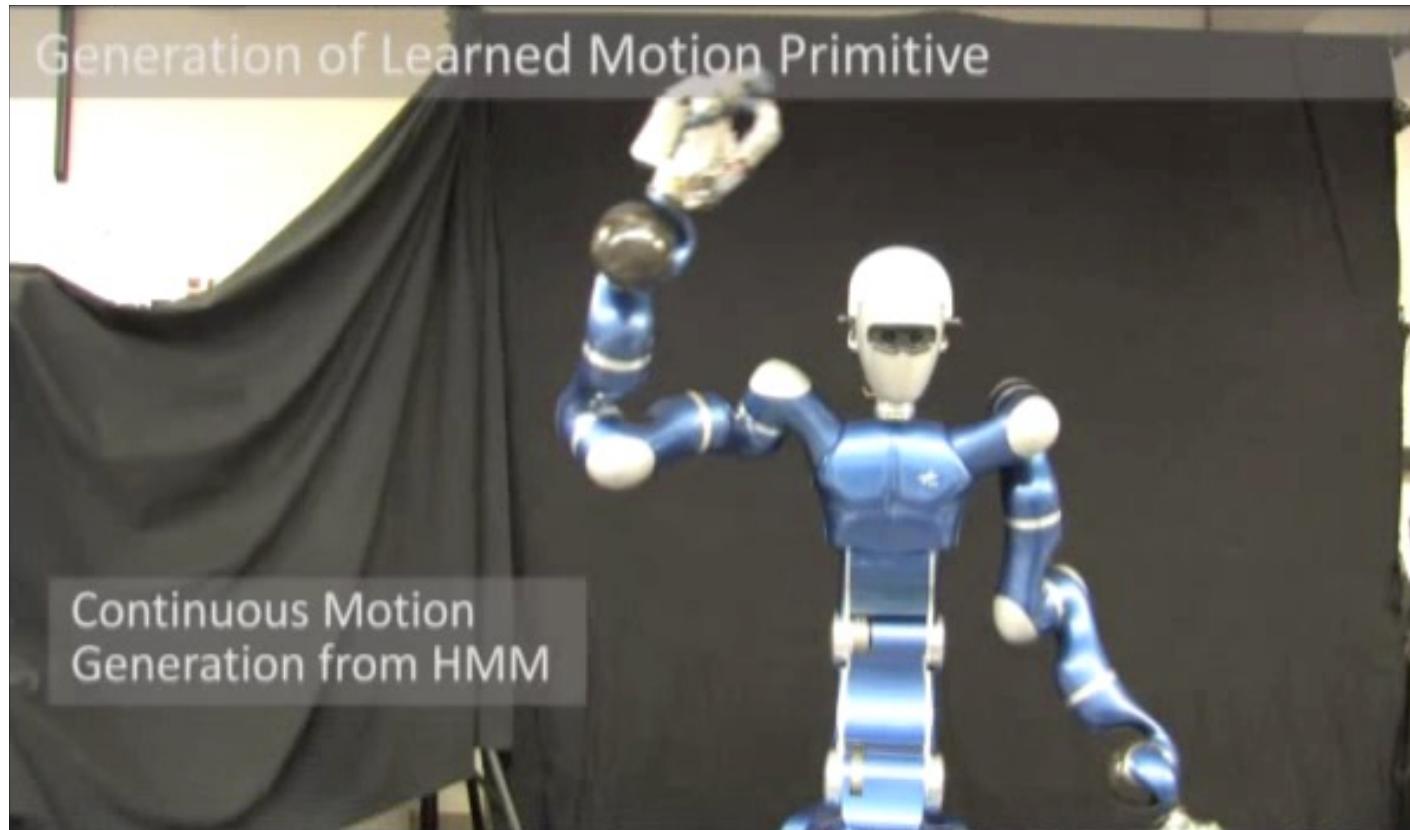
three squares in:





Robot learning by imitation

- learning from human motion primitives (imitation)
- motion refinement by kinesthetic teaching (with impedance control)



@TUM, Munich (D. Lee, C. Ott), for the EU SAPHARI project



Using visual or depth sensor feedback

Stanford University
Artificial Intelligence Laboratory

Robust Visual Servo Control Using
the Reflexxes Motion Libraries

<http://cs.stanford.edu/groups/manips>

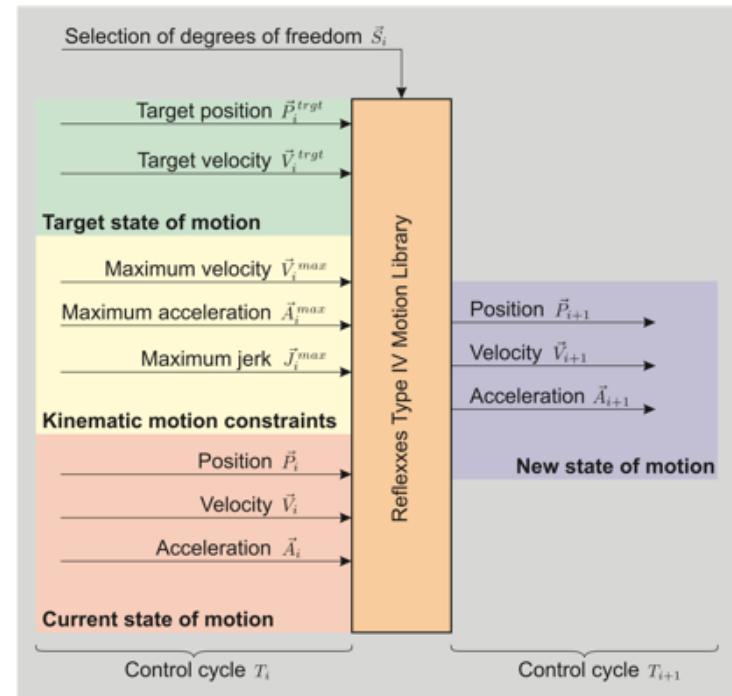
Stanford University
Artificial Intelligence Laboratory

Università di Roma "Sapienza"
Robotics Laboratory

Collision Avoidance Using
the Reflexxes Motion Libraries

video

- robust visual or depth (Kinect) feedback for motion tracking



- collision avoidance schemes
(here, redundancy w.r.t. an E-E task)

video



Panoramic view of control laws

- problems & methods for robot manipulators that will be considered
(control command is always a **joint torque**, if not **else** specified)

type of task	definition of error	joint space	Cartesian space	task space
	from one conf. to one desired	PD, PID, gravity compensation, iterative learning	PD with gravity compensation	visual servoing (kinematic scheme)
free motion	regulation	PD, PID, gravity compensation, iterative learning	PD with gravity compensation	visual servoing (kinematic scheme)
	trajectory tracking	feedback linearization, inverse dynamics + PD, passivity-based control, robust/adaptive control	feedback linearization	
contact motion (with force exchange)		-	impedance control (with variants), admittance control (kinematic scheme)	hybrid force-velocity control

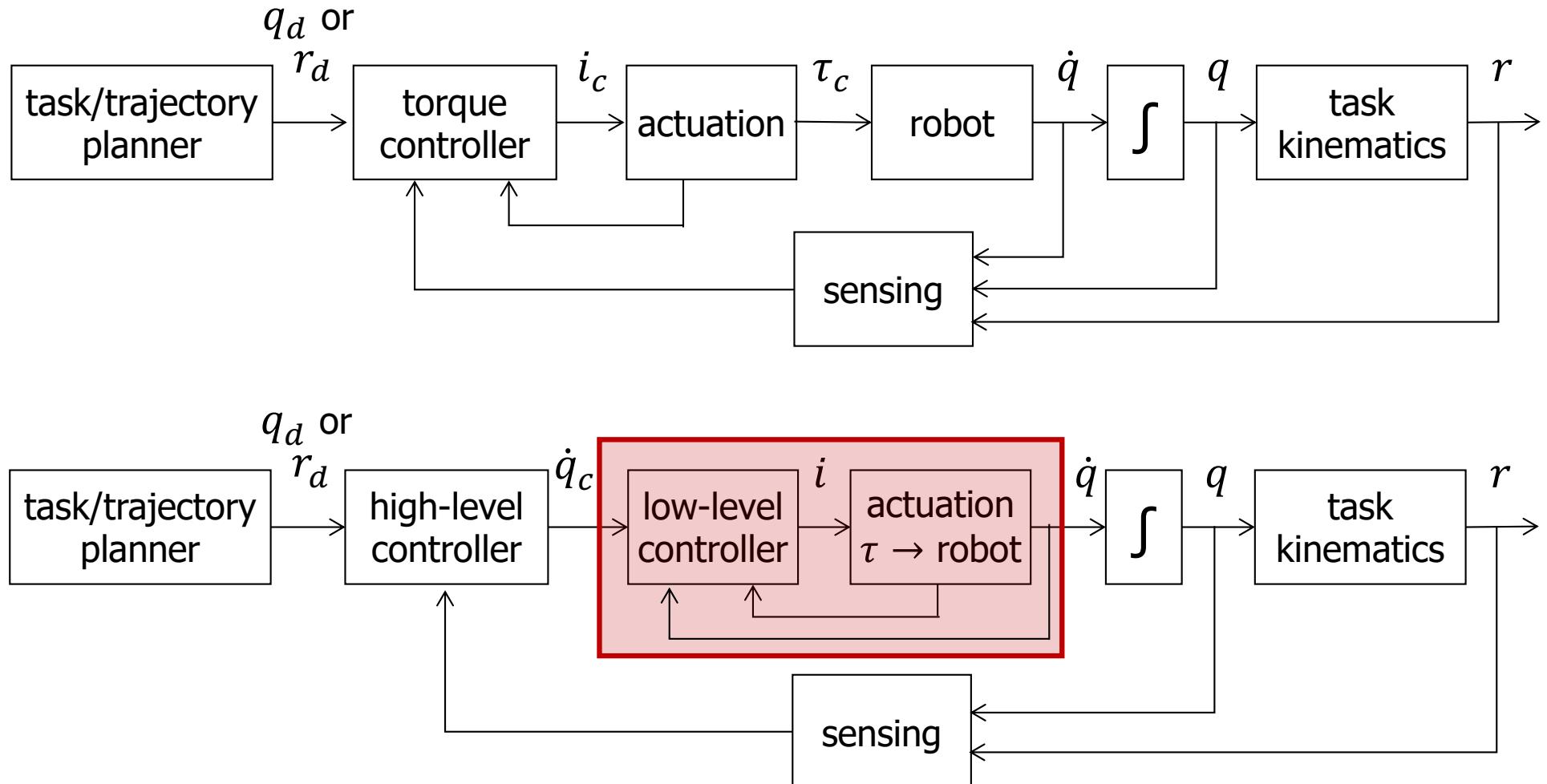


Dynamic or kinematic control laws

- specify torque to produce*
- **torque-controlled robots**
 - issue **current commands** $i = i_c$ (with $\tau_c = K_i i_c$) to drive the (electrical) motors, based on information on the **dynamic** model
 - often, a **low-level (analog) current loop** is present to enforce the execution of the desired command
 - may use a **torque measure** τ_J (by joint torque sensors) to do the same, in case of joint/transmission elasticity (with $\tau_J = K(\theta - q)$)
 - best suited for high dynamic performance and 'transparent' control of interaction forces
 - **position/motion-controlled robots**
 - issue **kinematic commands**: velocity $\dot{q} = \dot{q}_c$, acceleration $\ddot{q} = \ddot{q}_c$, or their integrated/micro-interpolated version $q = q_c$
 - references for a **low-level direct loop at high frequency** ($T_c \cong 400 \mu\text{s}!$)



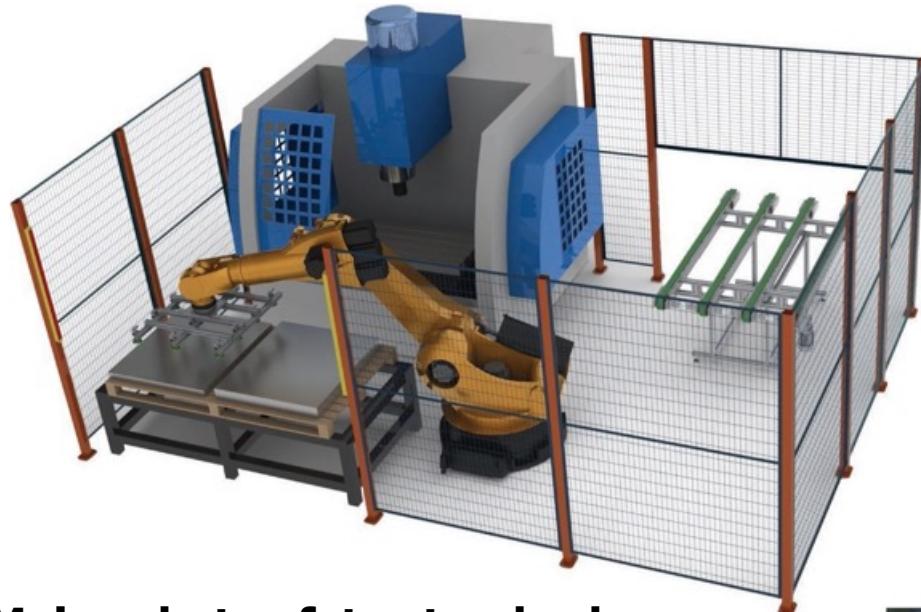
Torque- vs. position-controlled robots



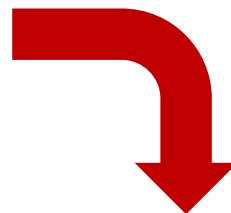
- both modes may be present even in the same robotic system



HRI in industrial settings



non-collaborative robots:
safety fences are required to
prevent harming human operators

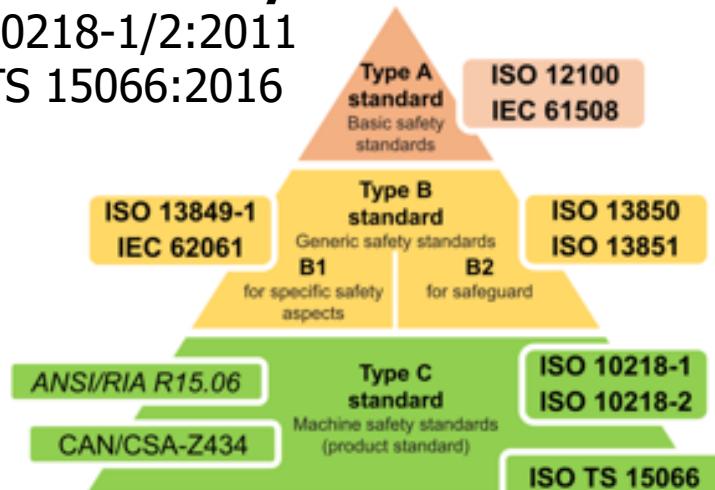


collaborative robots:
allow human workers to
stand in their proximity and
work together on the same task

Main robot safety standards

ISO 10218-1/2:2011

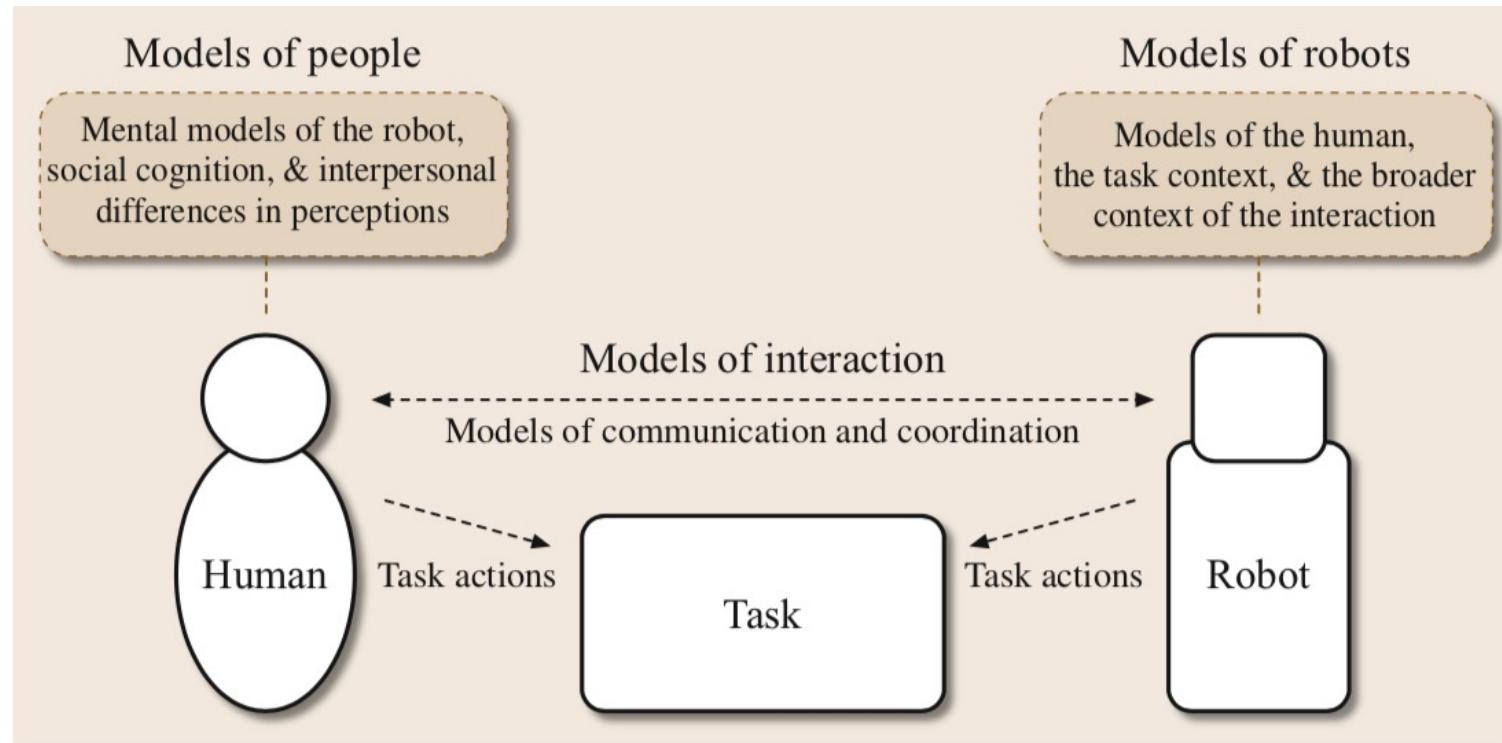
ISO/TS 15066:2016





Human-Robot Interaction taxonomy

- **cognitive** (cHRI) vs. **physical** (pHRI) Human-Robot Interaction
- cHRI models of humans, of robots, and of the interaction itself
 - dialog-based, intention- and activity-based, simulation-theoretic models



B. Mutlu, N. Roy, S. Sabanovic: *Ch. 71, Springer Handbook of Robotics, 2016*



Human-Robot Interaction taxonomy

- pHRI planned and controlled robot behaviors: 3-layer architecture

Safety

lightweight mechanical design
compliance at robot joints

**collision detection
and safe reaction**

Coexistence

robot and human sharing
the same workspace

collision avoidance
no need of physical contact

Collaboration

contactless, e.g., gestures
or voice commands

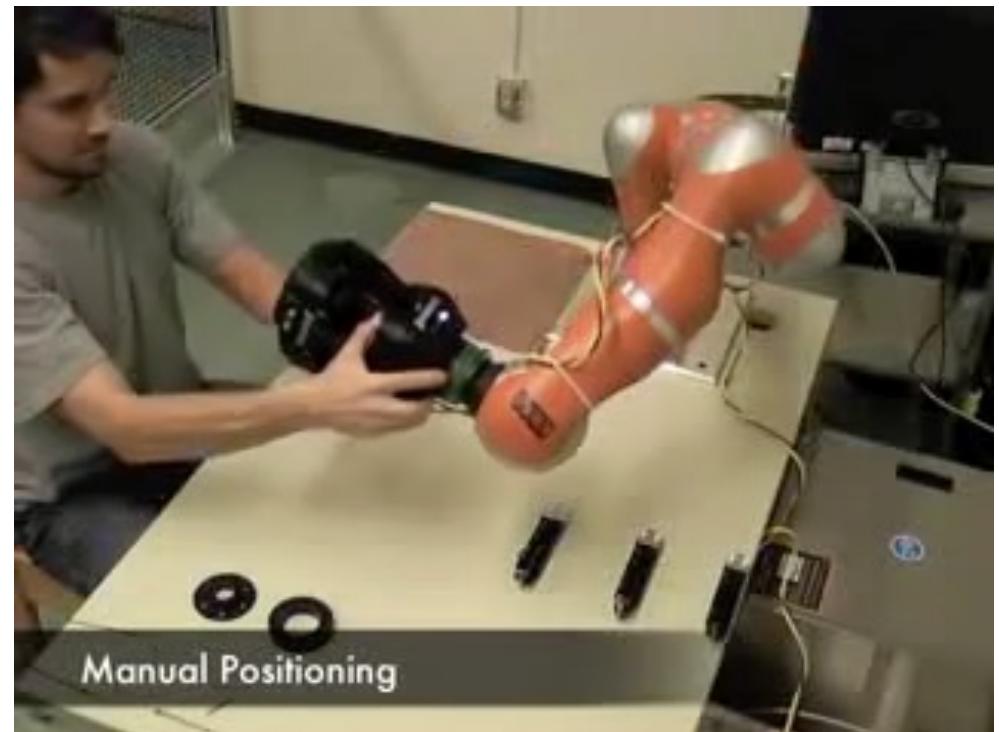
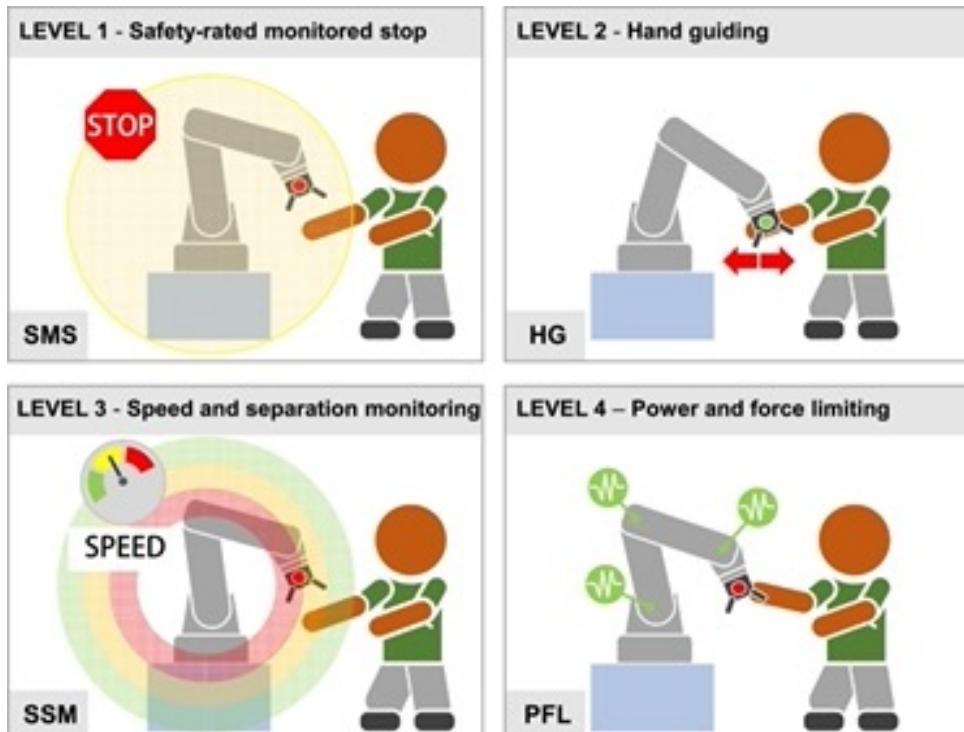
with intentional contact and
coordinated exchange of forces

A. De Luca, F. Flacco: *IEEE BioRob Conference, 2012*



Human-Robot Collaboration

- the different possible levels of pHRI are represented also within ISO safety standards (from safe coexistence to safe collaboration)



V. Villani et al.: *Mechatronics*, 2018

[video](#)



Robotics 2

Regulation in the Joint Space

(with an introduction to stability)

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI



$$\dot{v} = K_p (q_d - q) - K_d \dot{q}$$

PD NO GRAVITY
 $K_p > 0, K_d > 0$ symm

non linear eq.

$$\dot{v} = K_p (q_d - q) - K_d \dot{q} + g(q)$$

PD + GRAVITY CANCELLATION
 $K_p > 0 / K_d > 0$, symm

← how many comp.
2s sl. f.

$$\ddot{v} = K_p (q_d - q) - K_d \dot{q} + g(q_d)$$

PD + GRAVITY COMPENSATION

$K_p > 0, K_d > 0$ symm

$$\left\| \frac{\partial g}{\partial q} \right\| \leq \lambda, \forall q$$

in all cases globally asymptotically
stable in q^* state $x_e = \begin{pmatrix} q_d \\ 0 \end{pmatrix}$

⚠ we usually work with estimation of $g \Rightarrow \hat{g}(q)$

$$\hat{g}(q_d)$$



Equilibrium states of a robot

Euler-Lagrange Approach

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

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} q \\ \dot{q} \end{pmatrix}$$

generalized
coordinates

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

$\uparrow = 0$

$$= f(x) + G(x_1)u$$

x_e unforced equilibrium
 $(u = 0)$

no command applied

$$f(x_e) = 0 \rightarrow \begin{cases} x_{e2} = 0 \\ g(x_{e1}) = 0 \end{cases}$$

x_e forced equilibrium
 $(u = u(x))$

state feedback as input

$$f(x_e) + G(x_{e1})u(x_e) = 0 \rightarrow \begin{cases} x_{e2} = 0 \\ u(x_e) = g(x_{e1}) \end{cases}$$

all equilibrium states of mechanical systems have zero velocity!

joint torques must balance gravity at the equilibrium!



Stability of dynamical systems

definitions - 1

$$\dot{x} = f(x)$$

e.g., a closed-loop system
(under feedback control)

→ small perturbations \Rightarrow small effects in the state

stability of x_e

$$\forall \varepsilon > 0, \exists \delta_\varepsilon > 0: \|x(t_0) - x_e\| < \delta_\varepsilon \xrightarrow{\text{initial state}} \|x(t) - x_e\| < \varepsilon, \forall t \geq t_0 \quad \xrightarrow{\text{bound per time inst.}}$$

asymptotic stability of x_e

stability +

$$\exists \delta > 0: \|x(t_0) - x_e\| < \delta \Rightarrow \|x(t) - x_e\| \rightarrow 0, \text{ for } t \rightarrow \infty$$

asymptotic stability may become **global** ($\forall \delta > 0$, finite)

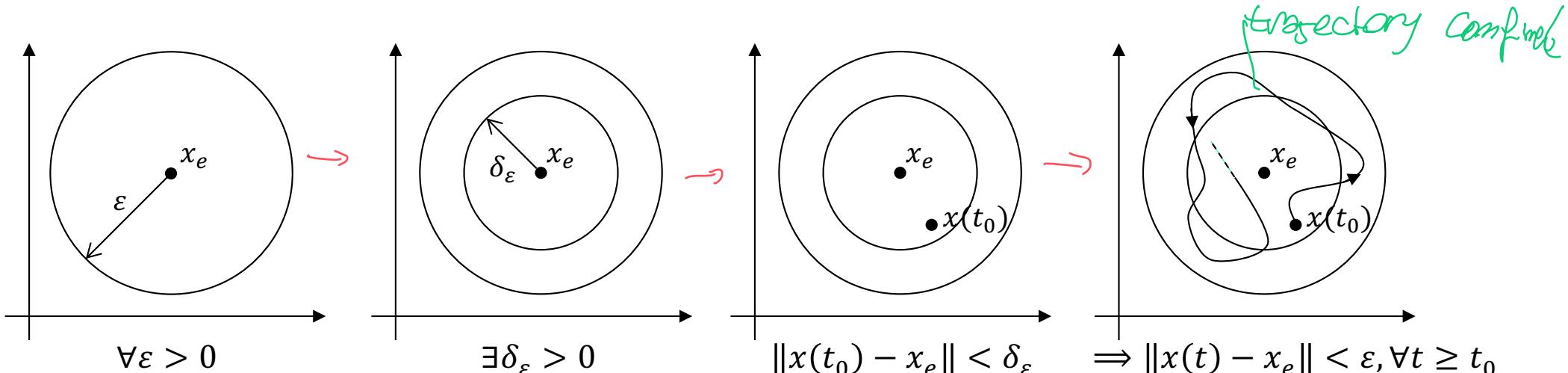
note: these are definitions of stability "in the sense of Lyapunov"

To we don't say "if exist" but "A δ exists"

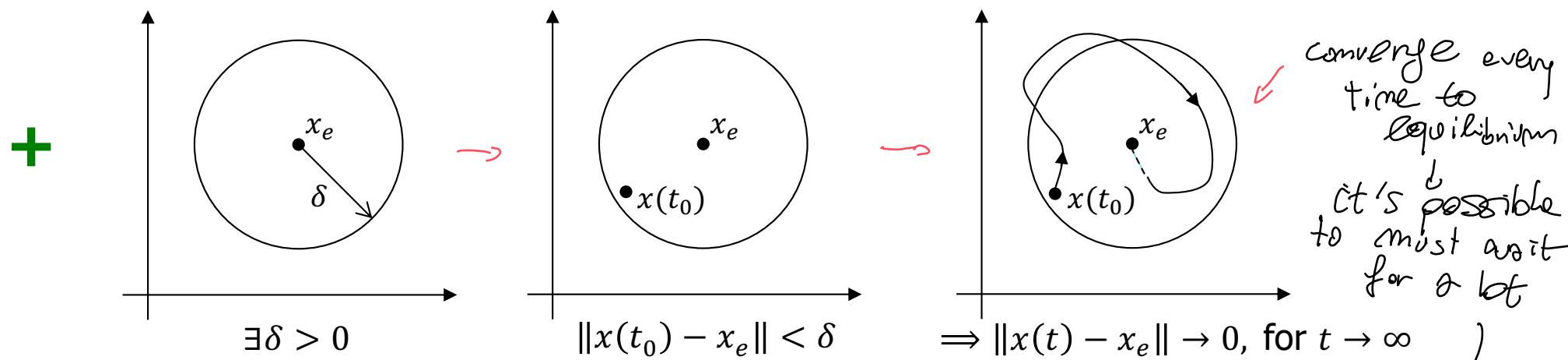


Stability vs. asymptotic stability

whiteboard...



equilibrium state x_e is **stable**



equilibrium state x_e is **asymptotically stable**

exp. stability solves



Stability of dynamical systems

definitions - 2

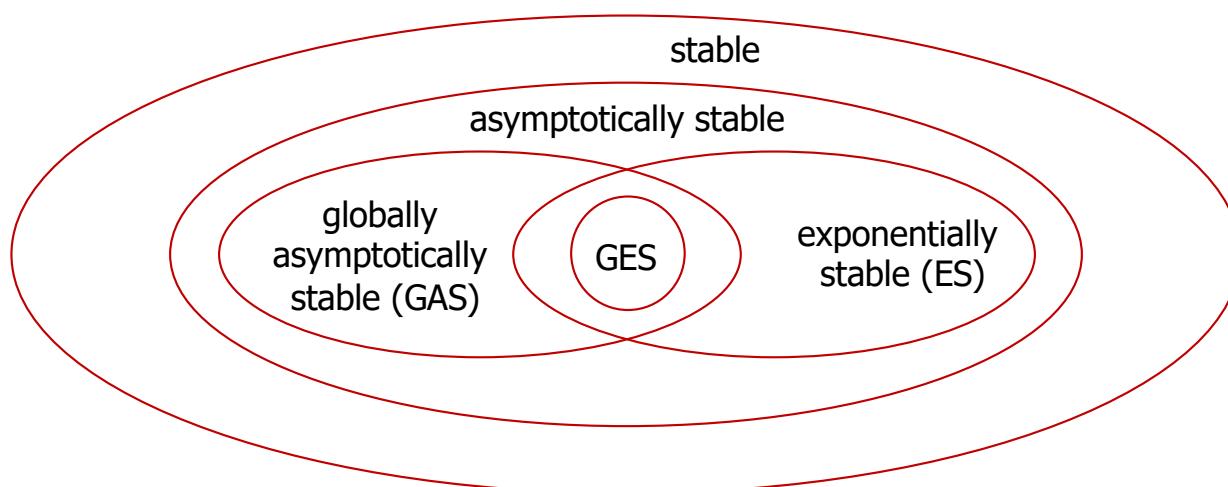
exponential stability of x_e

error future bounded by initial error and exp. function
exponential rate λ

$$\exists \delta, c, \lambda > 0: \|x(t_0) - x_e\| < \delta \Rightarrow \|x(t) - x_e\| \leq ce^{-\lambda(t-t_0)} \|x(t_0) - x_e\|$$

- allows to estimate the time needed to "approximately" converge: for $c = 1$, in $t - t_0 = 3 \times$ the time constant $\tau = 1/\lambda$, the initial error is reduced to 5%
- typically, this is a local property only (within some maximum finite radius δ)
 \Rightarrow such "domain of attraction" is hard to be estimated accurately

taxonomy
of stability
definitions



a necessary condition for x_e to be GAS is that it is the only equilibrium state of the system



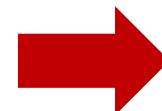
The need for analysis and criteria

whiteboard...

a nonlinear system $\dot{x} = f(x)$ in \mathbb{R}^2 two equilibria $f(x_e) = 0$

\leadsto simply unfolded
non linear
system

$$\begin{cases} \dot{x}_1 = 1 - x_1^3 \\ \dot{x}_2 = x_1 - x_2^2 \end{cases}$$



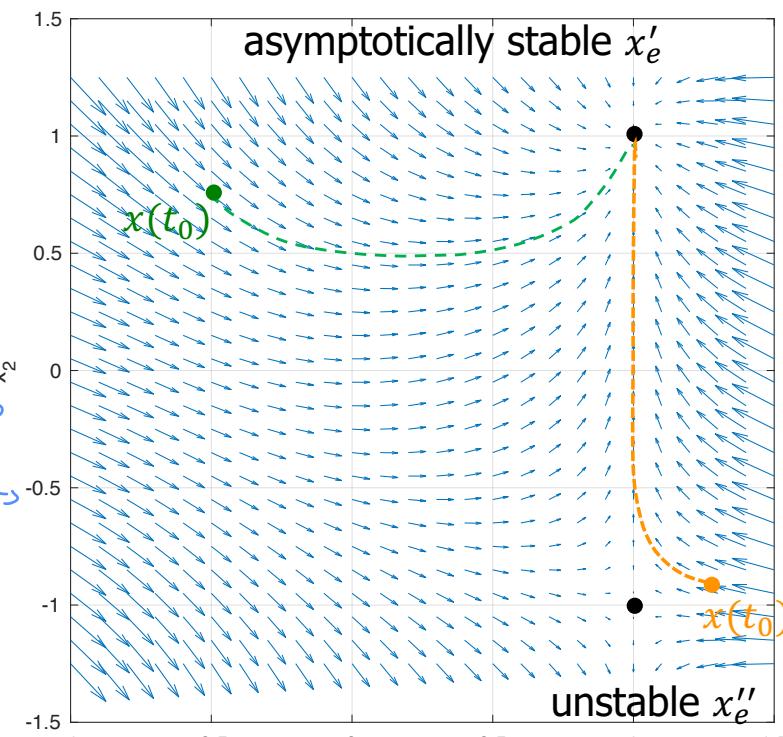
$$x'_e = (1, 1), \quad x''_e = (1, -1)$$

to assess (asymptotic) stability [or not] of equilibria, do we need to compute all system trajectories, starting from all possible initial states $x(t_0)$?



rather, we may be able to just look at the time evolution of a scalar function V , evaluated analytically along the state trajectories of the system (even in \mathbb{R}^n !)

IMPOSSIBLE IN GENERAL
 draw with matlab
 the trajectories



follow the arrows find
out the x_C 6



Stability of dynamical systems

results - 1

Lyapunov candidate

$V(x): \mathbb{R}^n \rightarrow \mathbb{R}$ such that

$$V(x_e) = 0, V(x) > 0, \forall x \neq x_e$$

positive
definite
function

typically, quadratic (e.g., $\frac{1}{2}(x - x_e)^T P(x - x_e)$ with level surfaces = ellipsoids)
may also be a local candidate only ($\forall x \neq x_e: \|x - x_e\| < \delta$)

sufficient condition of stability

$\exists V$ candidate: $\dot{V}(x) \leq 0$, along the trajectories of $\dot{x} = f(x)$

negative
semi-definite
function

sufficient condition of asymptotic stability

$\exists V$ candidate: $\dot{V}(x) < 0$, along the trajectories of $\dot{x} = f(x)$

negative
definite
function

sufficient condition of instability

$\exists V$ candidate: $\dot{V}(x) > 0$, along the trajectories of $\dot{x} = f(x)$



Stability of dynamical systems

results - 2

LaSalle Theorem

if $\exists V$ candidate: $\dot{V}(x) \leq 0$ along the trajectories of $\dot{x} = f(x)$

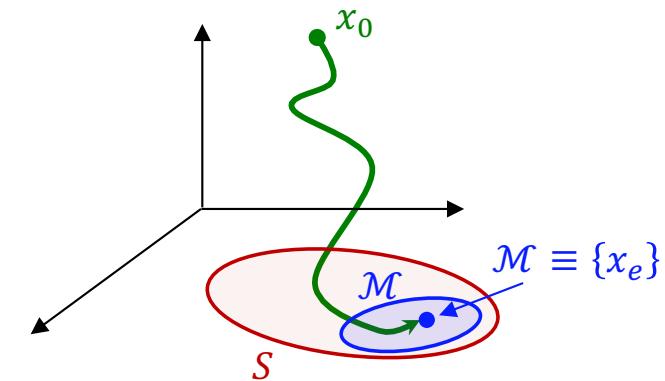
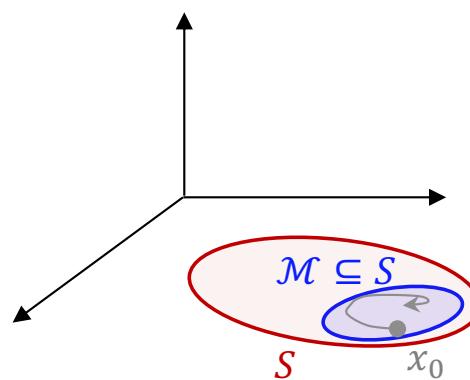
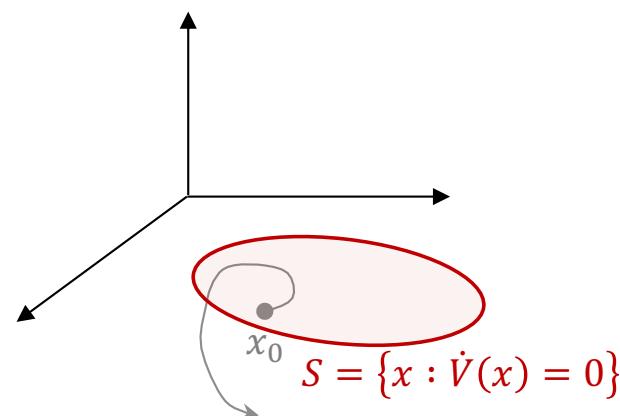


then system trajectories asymptotically converge to the largest invariant set $\mathcal{M} \subseteq S = \{x \in \mathbb{R}^n : \dot{V}(x) = 0\}$

\mathcal{M} is invariant if $x(t_0) \in \mathcal{M} \Rightarrow x(t) \in \mathcal{M}, \forall t \geq t_0$

Corollary

$\mathcal{M} \equiv \{x_e\} \rightarrow$ asymptotic stability



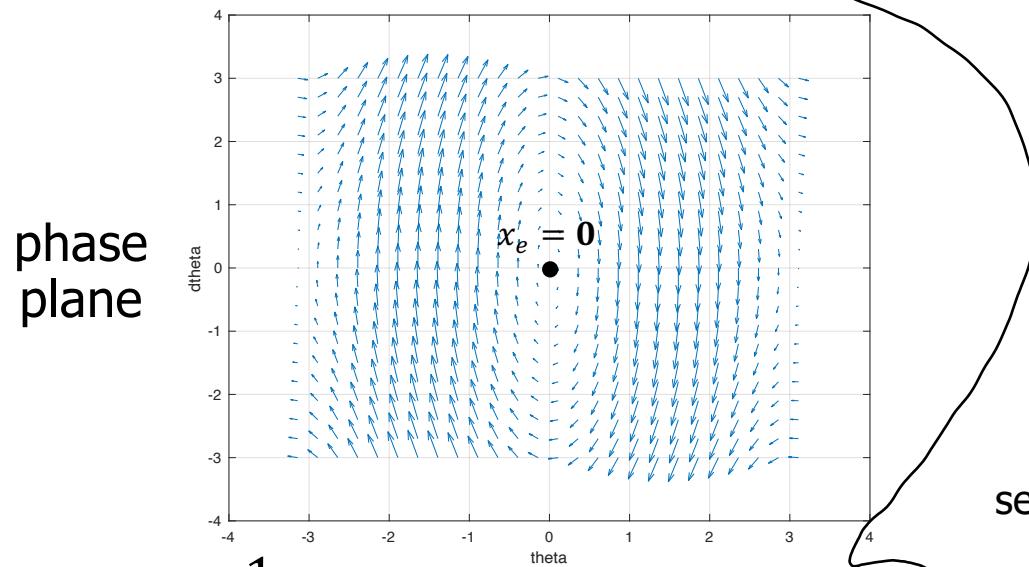


Bird-eye view on Lyapunov analysis

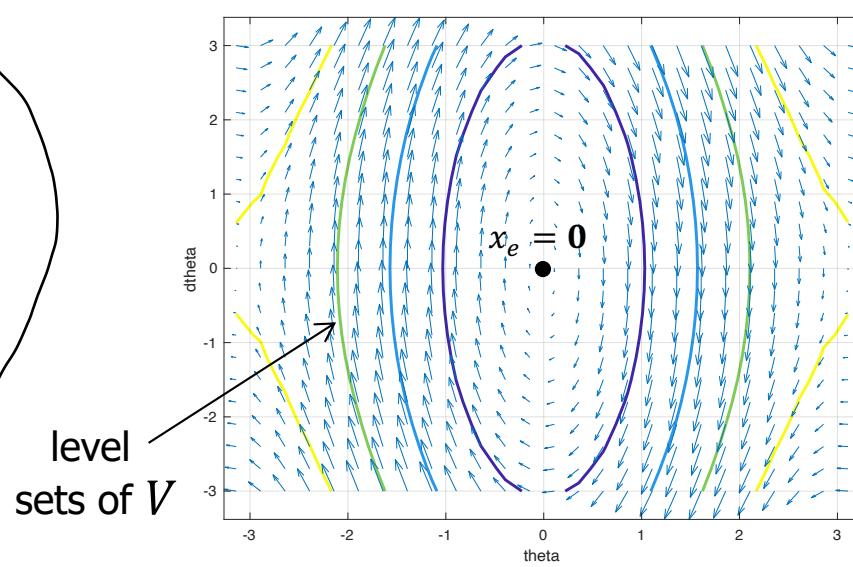
whiteboard...

a mass m at the end of an unforced (passive) pendulum of length l

$$ml^2\ddot{\theta} + b\dot{\theta} + mlg_0 \sin \theta = 0 \quad \Rightarrow \quad x = (x_1, x_2) \\ \text{lower equilibrium at } \theta_e = 0 \quad \Rightarrow \quad = (\theta, \dot{\theta}) \in \mathbb{R}^2 \quad \Rightarrow \quad \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\left(\frac{g_0}{l}\right) \sin x_1 - \left(\frac{b}{ml^2}\right) x_2 \end{cases}$$



$$V = E = \frac{1}{2} ml^2 \dot{\theta}^2 + mlg_0 (1 - \cos \theta) \geq 0$$



$$V = 0 \Leftrightarrow x_e = (\theta_e, \dot{\theta}_e) = (0,0)$$

$$\dot{V} = \dot{\theta}(ml^2\ddot{\theta} + mlg_0 \sin \theta) = -b\dot{\theta}^2 \leq 0$$

\Rightarrow stability of equilibrium $x_e = 0$ (... at least!) start from this upper eq.

\Rightarrow use LaSalle: $\dot{V} = 0 \Leftrightarrow \dot{\theta} = 0 \Rightarrow \ddot{\theta} = -\left(\frac{g_0}{l}\right) \sin \theta \neq 0$ unless $\theta = \theta_e = 0$ (or π !) converge to a set where velocity = 0

\Rightarrow local asymptotic stability



Stability of dynamical systems

results - 3

Usually dealing with time-varying systems

- previous results are also valid for **periodic** time-varying systems

$$\dot{x} = f(x, t) = f(x, t + T_p) \Rightarrow V(x, t) = V(x, t + T_p)$$

- for general **time-varying** systems (e.g., in robot trajectory tracking control)

$$\dot{x} = f(x, t)$$

Barbalat Lemma

if i) a function $V(x, t)$ is lower bounded

ii) $\dot{V}(x, t) \leq 0$

then $\Rightarrow \exists \lim_{t \rightarrow \infty} V(x, t)$ (but this does **not** imply that $\lim_{t \rightarrow \infty} \dot{V}(x, t) = 0$)

if in addition iii) $\ddot{V}(x, t)$ is bounded

then $\Rightarrow \lim_{t \rightarrow \infty} \dot{V}(x, t) = 0$

↳ if can apply then LaSalle and the other theorems

Corollary

if a Lyapunov candidate $V(x, t)$ satisfies Barbalat Lemma along the trajectories of $\dot{x} = f(x, t)$, then the conclusions of LaSalle Theorem hold



Stability of dynamical systems

additional definition and result (for robust control)

“practical” stability of a set S

$$\exists T(x(t_0), S) \in \mathbb{R}: x(t) \in S, \forall t \geq t_0 + T(x(t_0), S)$$

a finite time

also known as u.u.b. stability

⇒ trajectories $x(t)$ are “uniformly ultimately bounded” (use in robust control)

sufficient condition of u.u.b. stability of a set S

$\exists V$ candidate: i) S is a level set of V for a given c_0

$$S = S(c_0) = \{x \in \mathbb{R}^n: V(x) \leq c_0\}$$

ii) $\dot{V}(x) < 0$ along trajectories of $\dot{x} = f(x), x \notin S$



Stability of linear systems

time-invariant case

$$\dot{x} = Ax$$

$x_e = 0$ is always an equilibrium state

- I. asymptotic stability
- II. global asymptotic stability
- III. exponential stability
- IV. $\sigma(A) \subset \mathbb{C}^-$ (all eigenvalues of A have negative real part)
- V. $\forall Q > 0$ (positive definite), $\exists! P > 0$: $A^T P + PA = -Q$
Lyapunov equation $\Rightarrow \frac{1}{2}x^T Px$ is a Lyapunov candidate

ALL EQUIVALENT !!

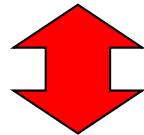
if $x_e = 0$ is an asymptotically stable equilibrium,
then it is necessarily the unique equilibrium



Stability of the linear approximation

Let $\Delta x = x - x_e$ and let $\dot{\Delta x} = \frac{df}{dx} \Big|_{x=x_e} (x - x_e) = A \Delta x$ be the linear approximation of $\dot{x} = f(x)$ around the equilibrium x_e

A asymptotically stable ($\sigma(A) \subset \mathbb{C}^-$)



the original nonlinear system is
exponentially stable at the origin

this is only a **local** result
(used also to estimate the domain of attraction)



PD control

(proportional + derivative action on the error)

robot

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

goal: asymptotic stabilization (= **regulation**)
of the closed-loop equilibrium state

$$q = q_d, \dot{q} = 0$$

possibly obtained from kinematic inversion: $q_d = f^{-1}(r_d)$

control law

$$u = K_P(q_d - q) - K_D \dot{q}$$

← simplest possible

$K_P > 0, K_D > 0$ (positive definite), symmetric



Asymptotic stability with PD control

Theorem 1

In the absence of gravity ($g(q) \equiv 0$), the robot state $(q_d, 0)$ under the given PD joint control law is **globally asymptotically stable**

Proof

always PD except eq.

Lyapunov candidate

let $e = q_d - q$ (q_d constant)

$$V = \frac{1}{2} \dot{q}^T M(q) \dot{q} + \frac{1}{2} e^T K_P e \geq 0$$

$$V = 0 \Leftrightarrow e = \dot{e} = 0$$

$$\begin{aligned} \dot{V} &= \dot{q}^T M \ddot{q} + \frac{1}{2} \dot{q}^T \dot{M} \dot{q} - e^T K_P \dot{q} = \dot{q}^T \left(u - \underbrace{S \dot{q} + \frac{1}{2} \dot{M} \dot{q}}_{= 0, \text{ due to energy conservation}} \right) - e^T K_P \dot{q} \\ &= \cancel{\dot{q}^T K_P e} - \cancel{\dot{q}^T K_D \dot{q}} - e^T \cancel{K_P \dot{q}} = -\dot{q}^T K_D \dot{q} \leq 0 \quad (K_D > 0, \text{ symmetric}) \end{aligned}$$

up to here, we proved stability only

but $\dot{V} = 0 \Leftrightarrow \dot{q} = 0$ continues ...



Asymptotic stability with PD control

$$\dot{V} = 0 \Leftrightarrow \dot{q} = 0$$

LaSalle



system trajectories converge to the largest invariant set of states \mathcal{M} where $\dot{q} \equiv 0$
(that is $\dot{q} = \ddot{q} = 0$)

$$\dot{q} = 0$$



$$M(q)\ddot{q} = K_P e$$

closed-loop dynamics

$$\ddot{q} = \underbrace{M^{-1}(q)K_P e}_{\text{invertible}}$$

invertible

$$\dot{q} = 0, \ddot{q} = 0 \Leftrightarrow e = 0$$



the only invariant state in $\dot{V} = 0$ is given by $q = q_d, \dot{q} = 0$



note: typically, $K_P = \text{diag}\{k_{Pi}\}$, $K_D = \text{diag}\{k_{Di}\}$,

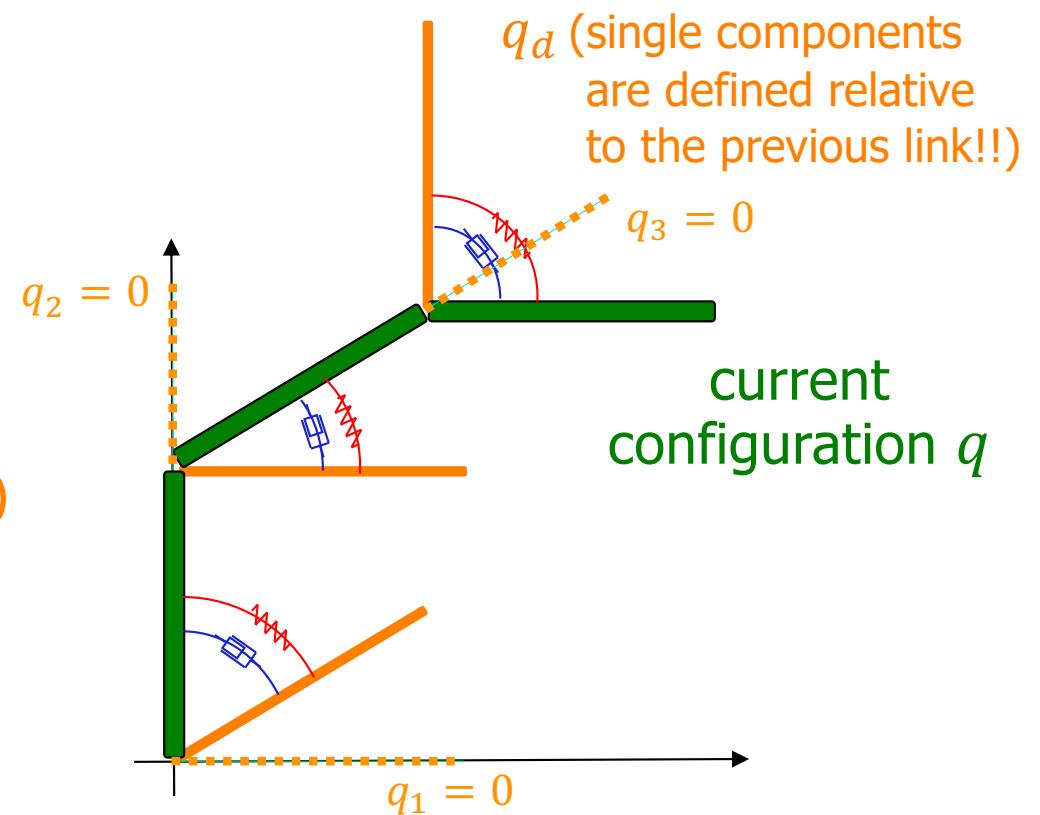
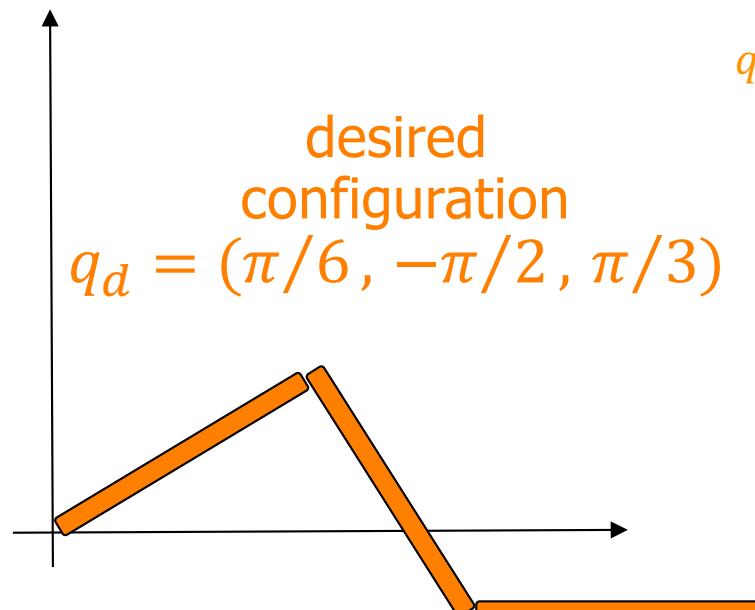
→ decentralized linear control (local to each joint)



Mechanical interpretation

- for diagonal positive definite gain matrices K_P and K_D (thus, with positive diagonal elements), such values correspond to stiffness of “virtual” **springs** and viscosity of “virtual” **dampers** placed at the joints

stiffness $k_{Pi} > 0$
 viscosity $k_{Di} > 0$





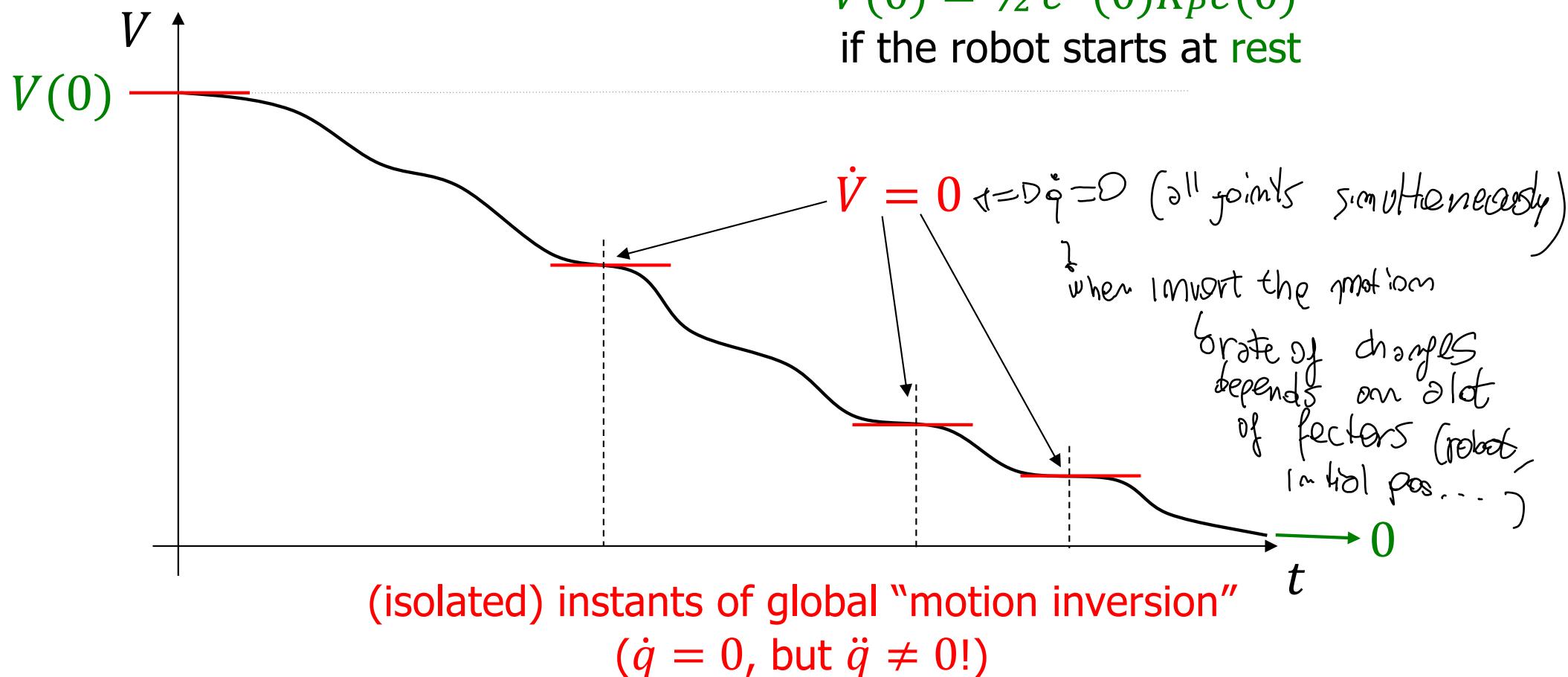
Plot of the Lyapunov function V

- time evolution of the Lyapunov candidate

\downarrow quadratic term of position error
in

$$V(0) = \frac{1}{2} e^T(0) K_P e(0)$$

if the robot starts at rest





Comments on PD control - 1

- choice of control gains affects robot evolution during transients and practical settling times *tuning is hard*
 - hard to define values that are “optimal” in the whole workspace
 - “full” K_P and K_D gain matrices allow to assign desired eigenvalues to the linear approximation of the robot dynamics around the final desired state $(q_d, 0)$
- when (joint) viscous friction is present, the derivative term in the control law is not strictly necessary
 - $-F_V \dot{q}$ in the robot model acts similarly to $\underline{-K_D \dot{q}}$ in the control law, but the latter can be modulated at will *we can avoid this(?)*
- in the absence of tachometers, the actual realization of the derivative term in the feedback law requires some processing of joint position data measured by digital encoders (or analog resolvers/potentiometers)



Comments on PD control - 2

- analog or digital implementation of derivative action in the control law when only position is measured at the joints (e.g., through encoders)

continuous-time
control law (design)

$$u(t) = K_P e(t) + K_D \dot{e}(t)$$

$$e = q_d - q, \dot{e} = -\dot{q}$$



representation in
the Laplace domain

$$u(s) = (K_P + K_D s) e(s)$$

not realizable as such
(non-proper transfer function)

$$u(s) = \left(K_P + \frac{K_D s}{1 + \tau s} \right) e(s)$$

derivative action limited
in bandwidth (up to $\omega \leq 1/\tau$)

transformation in
the Zeta-domain
(e.g., via backward
differentiation rule on
samples, every T_c sec)

$$u(z) = \left(K_P + K_D \frac{1 - z^{-1}}{T_c} \right) e(z)$$



$$u(z) = \left(K_P + K_D \frac{\frac{1 - z^{-1}}{T_c}}{1 + \tau \frac{1 - z^{-1}}{T_c}} \right) e(z)$$



discrete-time
implementations

$$u_k = K_P e_k + K_D \frac{e_k - e_{k-1}}{T_c}$$

both realizable

$$u_k = K_P e_k + \frac{K_D}{\tau + T_c} (e_k - e_{k-1}) + \frac{\tau}{\tau + T_c} (u_{k-1} - K_P e_{k-1})$$



Inclusion of gravity

- in the presence of gravity, the same previous arguments (and proof) show that the control law

$$u = K_P(q_d - q) - K_D \dot{q} + g(q)$$

$$K_P > 0, K_D > 0$$

will make the equilibrium state $(q_d, 0)$ globally asymptotically stable (nonlinear cancellation of gravity)

- if gravity is not cancelled or only approximately cancelled

$$u = K_P(q_d - q) - K_D \dot{q} + \hat{g}(q) \quad \hat{g}(q) \neq g(q)$$

it is $q \rightarrow q^* \neq q_d, \dot{q} \rightarrow 0$, with a steady-state position error

- q^* is not unique in general, except when K_P is chosen large enough
- explanation in terms of linear systems: there is no integral action before the point of access of the constant "disturbance" acting on the system



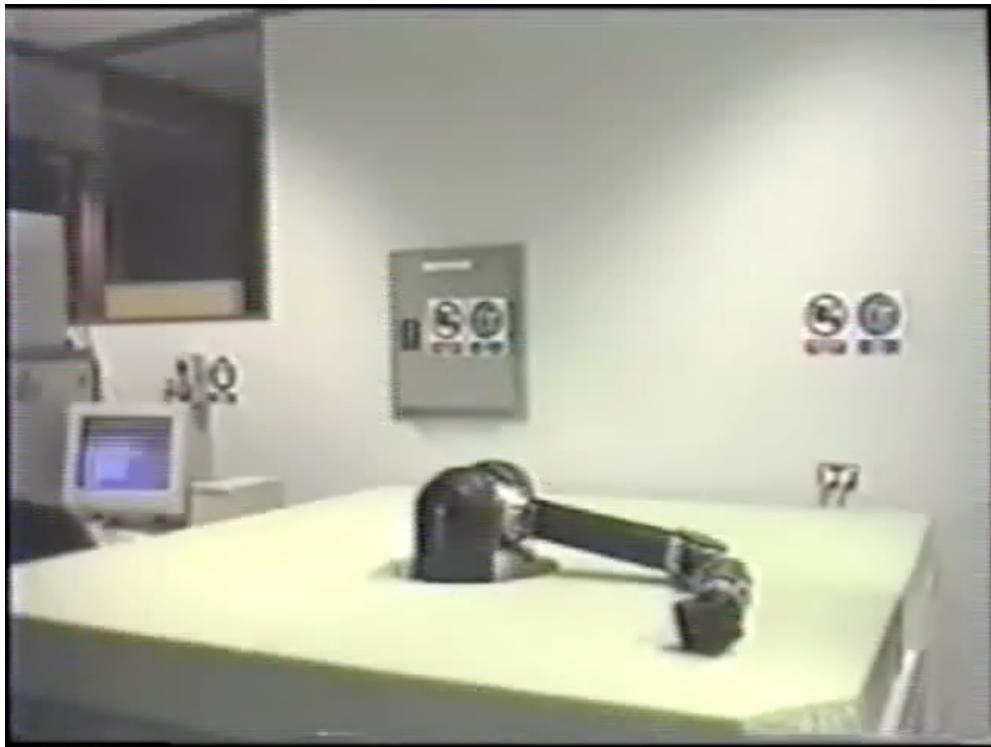
Approximate cancellation of gravity

WAM Barrett
(with some viscous friction)

$$\hat{g}(q) = g(q)$$

$$u = \hat{g}(q)$$

$$\hat{g}(q) \neq g(q)$$



two-part video

<http://handbookofrobotics.org/view-chapter/69/videodetails/611>

PD control + constant gravity compensation



if the robot potential energy $U(q)$ is bounded for all q , then its partial derivative $g(q)$ is also **bounded** everywhere and the following **structural property** holds

finite

$$\exists \alpha > 0: \left\| \frac{\partial^2 U}{\partial q^2} \right\| = \left\| \frac{\partial g}{\partial q} \right\| \leq \alpha, \forall q$$

contains in general trigonometric function

Upper bound to the norm

consequence

$$\|g(q) - g(q_d)\| \leq \alpha \|q - q_d\|$$

all terms here are vectors

note: induced norm of a matrix

$$\|A\| = \sqrt{\lambda_{\max}(A^T A)} \triangleq A_M \geq A_m \triangleq \sqrt{\lambda_{\min}(A^T A)}$$

consider real eigenvectors

LINEAR CONTROL law

$$u = K_P(q_d - q) - K_D \dot{q} + g(q_d)$$

*$K_P, K_D > 0$
symmetric*

linear feedback + **constant** feedforward



More on the basic assumption ...

(in PD control + gravity compensation)

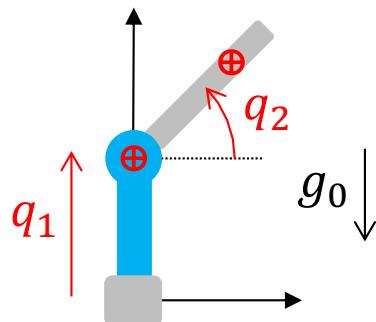
when is the (non-zero) gravity term $g(q)$ bounded for all q ?

- the robot has **all revolute** joints
 - all terms in $U(q)$ and thus in $g(q)$ are trigonometric (bounded)
- the robot has both types of joints, but **no prismatic variables** in $g(q)$
 - potential energy $U(q)$ may still be unbounded!
- all prismatic** joints of the robot have a **limited range**
 - ... ok, but one should take these limits into account in the control analysis

$$U = g_0(m_1 q_1 + m_2(q_1 + d_{c2} \sin q_2)) + U_0$$

$$g(q) = g_0 \left(\frac{(m_1 + m_2)}{m_2 d_{c2} \cos q_2} \right)$$

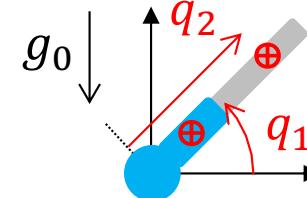
$$\left\| \frac{\partial g}{\partial q} \right\| \leq \alpha = m_2 d_{c2} g_0$$



PR robot

$$U = g_0(m_1 d_{c1} + m_2 q_2) \sin q_1 + U_0$$

$$g(q) = g_0 \left(\frac{(m_1 d_{c1} + m_2 q_2) \cos q_1}{m_2 \sin q_1} \right)$$



RP robot

$$\left\| \frac{\partial g}{\partial q} \right\| \leq ??$$



PD control + constant gravity compensation

stability analysis

Theorem 2

If $K_{P,m} > \alpha$, the state $(q_d, 0)$ of the robot under joint-space PD control + constant gravity compensation at q_d is **globally asymptotically stable**

Proof

1.

$(q_d, 0)$ is the unique closed-loop equilibrium state

in fact, for $\dot{q} = 0$ and $\ddot{q} = 0$, it is $K_P e = g(q) - g(q_d)$

which can hold only for $q = q_d$, because when $q \neq q_d$

$$\|K_P e\| \geq K_{P,m} \|e\| > \alpha \|e\| \geq \|g(q) - g(q_d)\|$$

$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = K_P e - K_D \dot{q} + g(q_d)$$



PD control + constant gravity compensation

stability analysis

with $e = q_d - q$, $g(q) = \left(\frac{\partial U}{\partial q}\right)^T$, consider as Lyapunov candidate

$$V = \frac{1}{2} \dot{q}^T M(q) \dot{q} + \frac{1}{2} e^T K_P e + U(q) - U(q_d) + e^T g(q_d)$$

2.

V is convex in \dot{q} and e , and zero only for $e = \dot{q} = 0$

$$\left(\frac{\partial V}{\partial \dot{q}}\right)^T = M(q) \dot{q} = 0 \text{ only for } \dot{q} = 0$$

$$\frac{\partial^2 V}{\partial \dot{q}^2} = M(q) > 0$$



$(q_d, 0)$ is a
global minimum
of $V \geq 0$

$$\left(\frac{\partial V|_{\dot{q}=0}}{\partial e}\right)^T = K_P e - \left(\frac{\partial U}{\partial q}\right)^T + g(q_d) = K_P e + g(q_d) - g(q) = 0$$

$$\partial e / \partial q = -I$$

only for $q = q_d$

$$\frac{\partial^2 V|_{\dot{q}=0}}{\partial e^2} = K_P + \frac{\partial^2 U}{\partial q^2} > 0, \text{ since } \|K_P\| = K_{P,M} \geq K_{P,m} > \alpha$$



PD control + constant gravity compensation

stability analysis

differentiating

$$V = \frac{1}{2} \dot{q}^T M(q) \dot{q} + \frac{1}{2} e^T K_P e + U(q) - U(q_d) + e^T g(q_d)$$

$$\begin{aligned}\dot{V} &= \dot{q}^T \left(M(q) \ddot{q} + \frac{1}{2} \dot{M}(q) \dot{q} \right) - e^T K_P \dot{q} + \frac{\partial U(q)}{\partial q} \dot{q} - \dot{q}^T g(q_d) \\ &= \dot{q}^T \left(u - S(q, \dot{q}) \dot{q} + \frac{1}{2} \dot{M}(q) \dot{q} - g(q) \right) - e^T K_P \dot{q} + \dot{q}^T (g(q) - g(q_d)) \\ &= \cancel{\dot{q}^T K_P e} - \cancel{\dot{q}^T K_D \dot{q}} + \dot{q}^T (g(q_d) - g(q)) - \cancel{e^T K_P \dot{q}} + \cancel{\dot{q}^T (g(q) - g(q_d))} \\ &= -\dot{q}^T K_D \dot{q} \leq 0\end{aligned}$$

for $\dot{V} = 0$ ($\Leftrightarrow \dot{q} = 0$), we have in the **closed-loop** system

$$M(q) \ddot{q} + g(q) = K_P e + g(q_d)$$

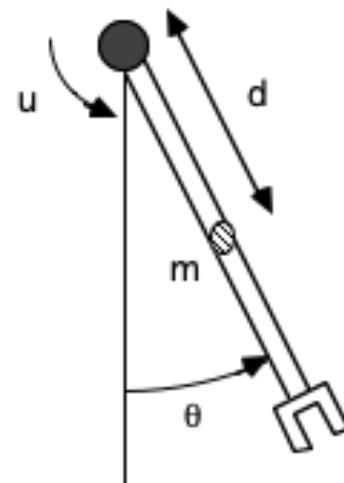
$$\rightarrow \ddot{q} = M^{-1}(q)(K_P e + g(q_d) - g(q)) = 0 \Leftrightarrow e = 0$$

by LaSalle Theorem, the thesis follows





Example of a single-link robot stability analysis



task: regulate the link position to the upward equilibrium

$$\theta_d = \pi \rightarrow g(\theta_d) = 0 \quad (\text{at } \theta_d = \pi, \text{ dist gravity} = 0, \text{ but during traj. not})$$

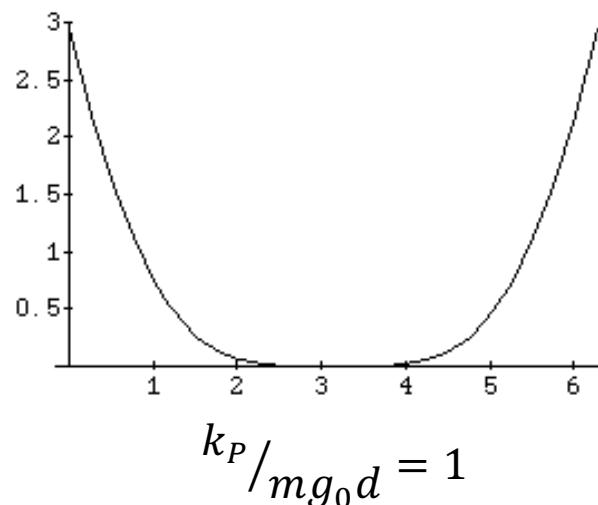
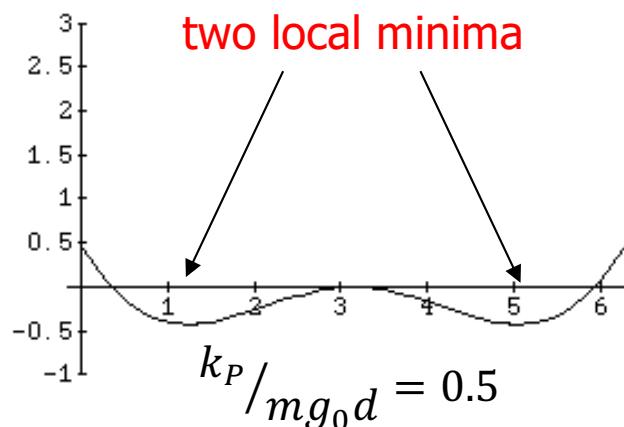
PD control + constant gravity compensation (here, zero!)

$$u = k_P(\pi - \theta) - k_D \dot{\theta}$$

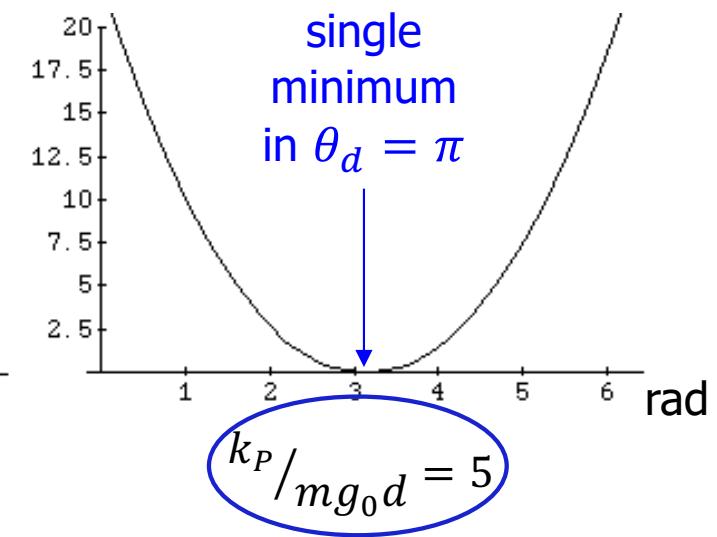
by Theorem 2, it is sufficient (here, also necessary*) to choose

$$k_P > \alpha = mg_0d, \quad k_D > 0$$

$$I\ddot{\theta} + mg_0d \sin \theta = u$$



plots of $V(\theta)$ (for $\dot{\theta} = 0$)

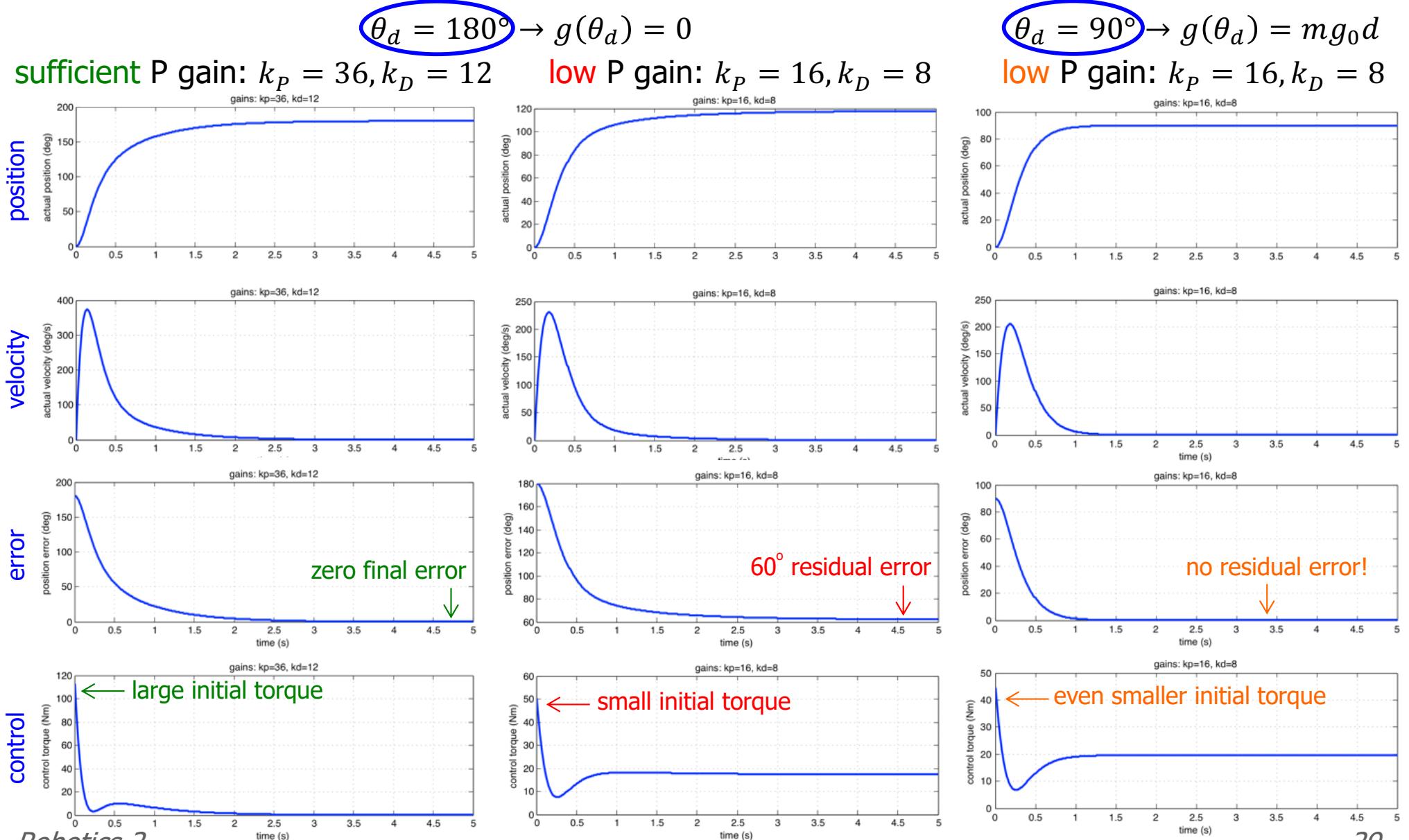


* by a local analysis of the linear approximation at π



Example of a single-link robot

simulations with data: $I = 0.9333$, $mg_0d = 19.62$ ($= \alpha$)





Approximate gravity compensation

the **approximate** control law

$$u = K_P(q_d - q) - K_D \dot{q} + \hat{g}(q_d)$$

leads, under similar hypotheses, to a **closed-loop equilibrium** q^*

- its **uniqueness is not guaranteed** (unless K_P is large enough)
- for $K_P \rightarrow \infty$, one has $q^* \rightarrow q_d$

conclusion: in the **presence of gravity**, the previous regulation control laws require an **accurate knowledge** of the **gravity term** in the dynamic model to guarantee the **zeroing** of the **position error** (since we can only use "finite" control gains \Rightarrow in practice, not too large)



PID control

- in linear systems, the addition of an integral control action is used to eliminate a constant error in the step response at steady state
- in robots, a PID may be used to recover such a position error due to an incomplete (or absent) gravity compensation/cancellation

→ the control law $u(t) = K_P(q_d - q(t)) + K_I \int_0^t (q_d - q(\tau)) d\tau - K_D \dot{q}(t)$

- is independent from any robot dynamic model term
- if the desired closed-loop equilibrium is asymptotically stable under PID control, the integral term is “loaded” at steady state to the value

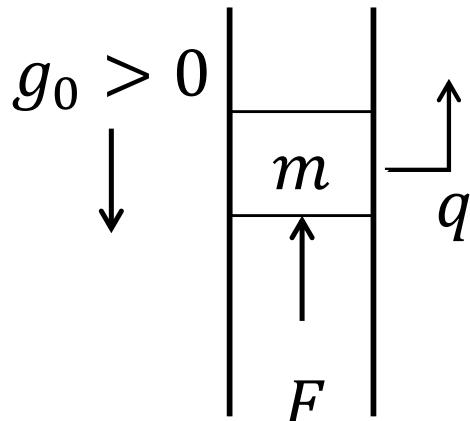
$$K_I \int_0^\infty (q_d - q(\tau)) d\tau = g(q_d)$$

- however, one can show only local asymptotic stability of this law, i.e., for $q(0) \in \Delta(q_d)$, under complex conditions on K_P, K_I, K_D and $e(0)$



Linear example with PID control

whiteboard...



$$m\ddot{q} + mg_0 = F \quad (\text{no friction})$$

$$F = k_P(q_d - q) - k_D\dot{q}$$

(PD \Rightarrow steady-state error $e = q_d - \bar{q}$, with $\bar{q} = q_d - \frac{mg_0}{k_P}$)

$$F = k_P(q_d - q) - k_D\dot{q} + mg_0$$

COMPARISON IN THIS CASE BECAUSE IS A CONST TERM

(PD + gravity cancellation \Rightarrow regulation $\forall k_P > 0, k_D > 0$)

$$F = k_P(q_d - q) - k_D\dot{q} + k_I \int_0^t (q_d - q(\tau)) d\tau$$

(PID \Rightarrow regulation $\forall k_I > 0, k_D > 0, k_P > \frac{mk_I}{k_D} > 0$)

with global exponential stability!

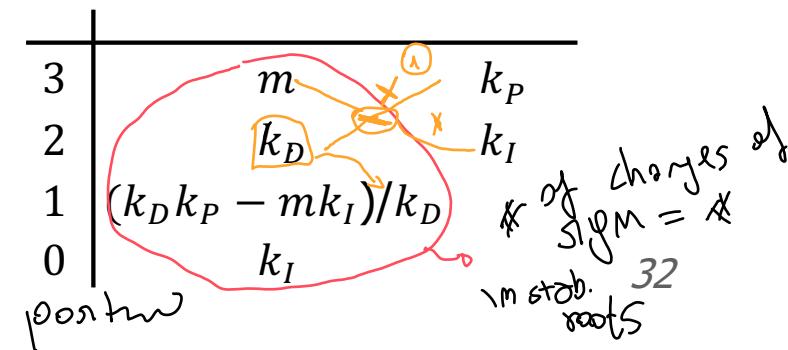
Laplace domain analysis: $e(s) = \mathcal{L}[e(t)]$, $d(s) = \mathcal{L}[mg_0]$ + Routh criterion

$$\frac{e(s)}{d(s)} = W_d(s) = \frac{s}{ms^3 + k_D s^2 + k_P s + k_I}$$

Robotics 2

if all 3 coeff. are positive in 2nd order (poly^2) \Rightarrow 2 roots

to studiare le soluzioni e vedere se gli:



2 states here q_1, \dot{q}_1

Let's plug $M = F$ into Newton eq.

$$m\ddot{q} + mg = K_p(q_0 - q) - K_0\dot{q} + K_I \int_0^t (q_d - q) dt$$

$$\dot{x}_1 = x_2$$

$$\ddot{x}_2 = \frac{1}{m} [K_p(q_0 - x_1) - K_0 \cancel{x}_2 - mg + K_I z]$$

$$\dot{z} = Az + B \begin{pmatrix} q_1 \\ mg \end{pmatrix}$$

$$S = \begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ z \end{pmatrix} \in \mathbb{R}^3$$

linear system closed form

that in this way can be solved in 2

$$\dot{z} = q_d - q$$

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} q_1 \\ \dot{q}_1 \end{pmatrix}$$



Saturated PID control

- more in general, one can prove global asymptotic stability of $(q_d, 0)$, under lower bound limitations for K_P, K_I, K_D (depending on suitable “bounds” on the terms in the dynamic model), for a nonlinear PID law

$$u(t) = K_P(q_d - q(t)) + K_I \int_0^t \Phi(q_d - q(\tau)) d\tau - K_D \dot{q}$$

where $\Phi(q_d - q)$ is a saturation-type function, such as

$$\Phi(x) = \begin{cases} \sin x, & |x| \leq \pi/2 \\ 1, & x > \pi/2 \\ -1, & x < -\pi/2 \end{cases} \quad \text{or} \quad \Phi(x) = \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

(see paper by R. Kelly, IEEE TAC, 1998; available as extra material on the course web)



Limits of robot regulation controllers

- response times needed for reaching the desired steady state are **not easily predictable in advance**
 - depend heavily on robot dynamics, on PD/PID gains, on the required total displacement, and on the interested area of the robot workspace
 - integral term (when present) needs some time to “unload” itself from the error history accumulated during transients
 - large initial errors are stored in the integral term
 - anti-windup schemes stop the integration when commands saturate
 - ... an intuitive explanation for the success of “saturated” PID law
- control efforts in the few first instants of motion typically exceed by far those required at steady state
 - especially for high positional gains
 - may lead to saturation (hard nonlinearity) of robot actuators



Regulation in industrial robots

- in industrial robots, the planner generates a **reference trajectory** $q_r(t)$ even when the task requires **only** positioning/regulation of the robot
 - “smooth” enough, with a user-defined **transfer time** T
 - reference trajectory **interpolates** initial and final desired position

$$q_r(0) = q(0) \quad q_r(t \geq T) = q_d$$

- $q_r(t)$ is used within a control law of the form

$$u = K_P(q_r(t) - q) + K_D(\dot{q}_r(t) - \dot{q}) + g(q)$$

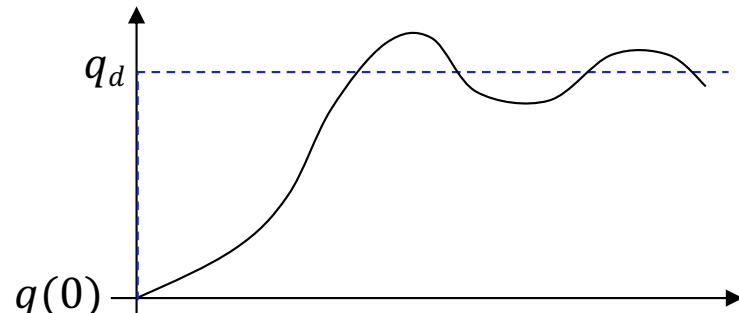
↑
often neglected → non interessante di seguire
verso la linea di regolazione

e.g., PD with gravity cancellation

- in this way, the position error is **initially zero**
- robot motion stays only “in the vicinity” of the reference trajectory until $t = T$, typically with **small** position errors (gains can be **larger!**)
- final regulation is only a “local” problem ($e(T) = q_d - q(T)$ is small)

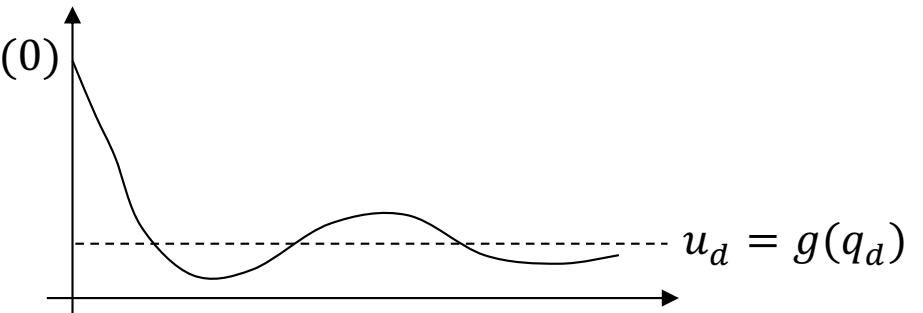


Qualitative comparison

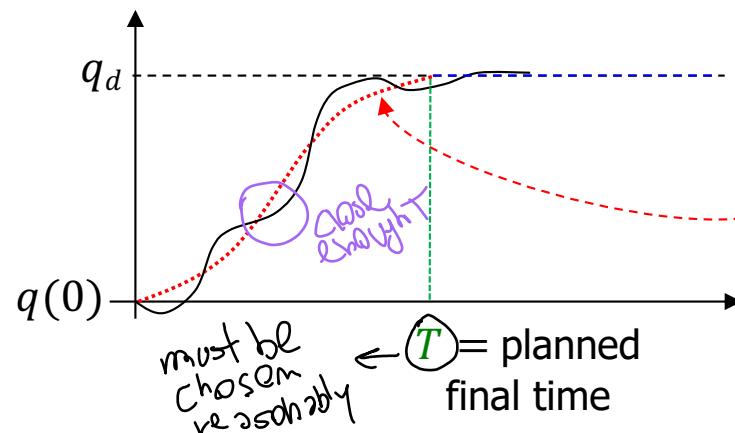


joint variables

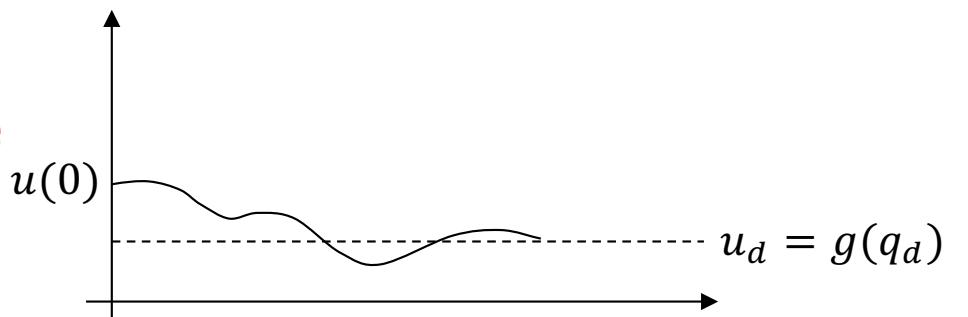
step variation
of desired position



control commands



time-varying
position
reference
 $q_r(t)$



- no saturation of commands: in principle, much larger gains can be used
- better prediction of settling times: local exponential convergence (designed on the linear approximation of the robot dynamics around $(q_d, 0)$)
- “fine tuning” of control gains made easier, but still a tedious and delicate task



Quantitative comparison

planar 2R arm

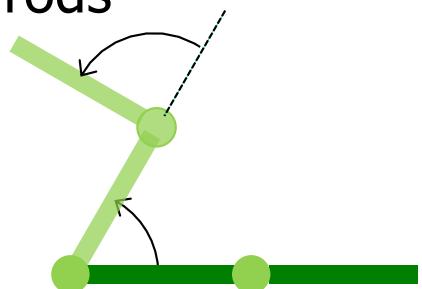
m_1	10 [kg]
m_2	5 [kg]
l_1	0.5 [m]
l_2	0.5 [m]
d_1	0.25 [m]
d_2	0.25 [m]
I_1	5/24 [kg m ²]
I_2	5/48 [kg m ²]

robot data: links are uniform thin rods

no gravity (horizontal plane)

rest-to-rest motion task:

$$q(0) = (0, 0) \text{ (straight)} \rightarrow q_d = (\pi/3, \pi/2)$$



interpolating trajectory: cubic polynomials

three case studies every positive gain works

a) low gains (overdamped) $K_P = \text{diag}\{80, 40\}, K_D = \text{diag}\{60, 30\}$

vs interpolating trajectory in $T = 2$ s

b) medium gains (very overdamped) $K_P = \text{diag}\{200, 100\}, K_D = \text{diag}\{200, 100\}$

vs interpolating trajectory in $T = 2$ s

usually most be different.

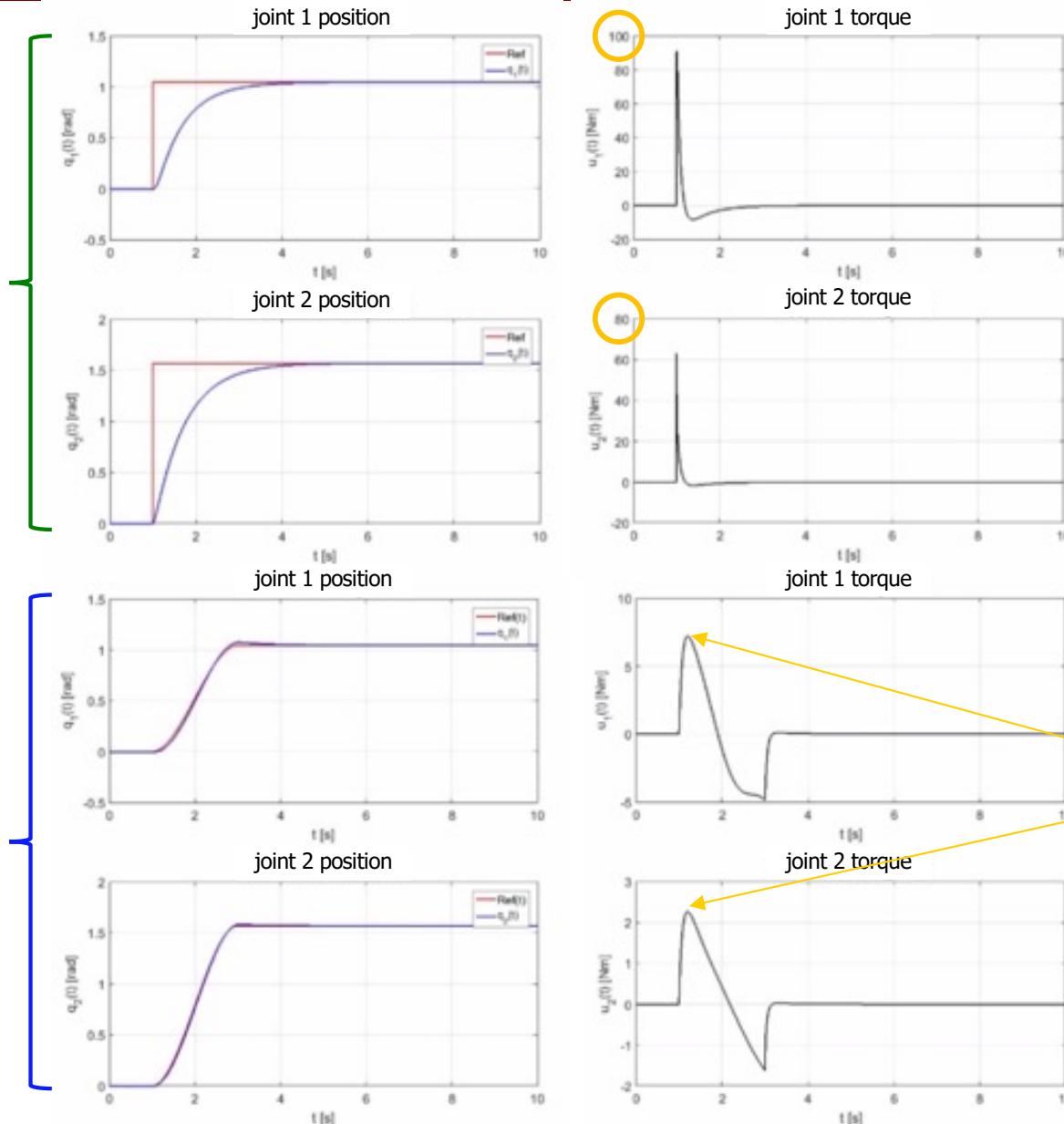
c) high gains $K_P = \text{diag}\{1250, 180\}, K_D = \text{diag}\{200, 70\}$

vs interpolating trajectory in $T = 1$ s, with torque saturation $u_{1,\max} = 30$ Nm,
 $u_{2,\max} = 10$ Nm



Comparison on a planar 2R arm – case a

PD with low gains
 $K_P = \text{diag}\{80, 40\}$
 $K_D = \text{diag}\{60, 30\}$
 (overdamped)



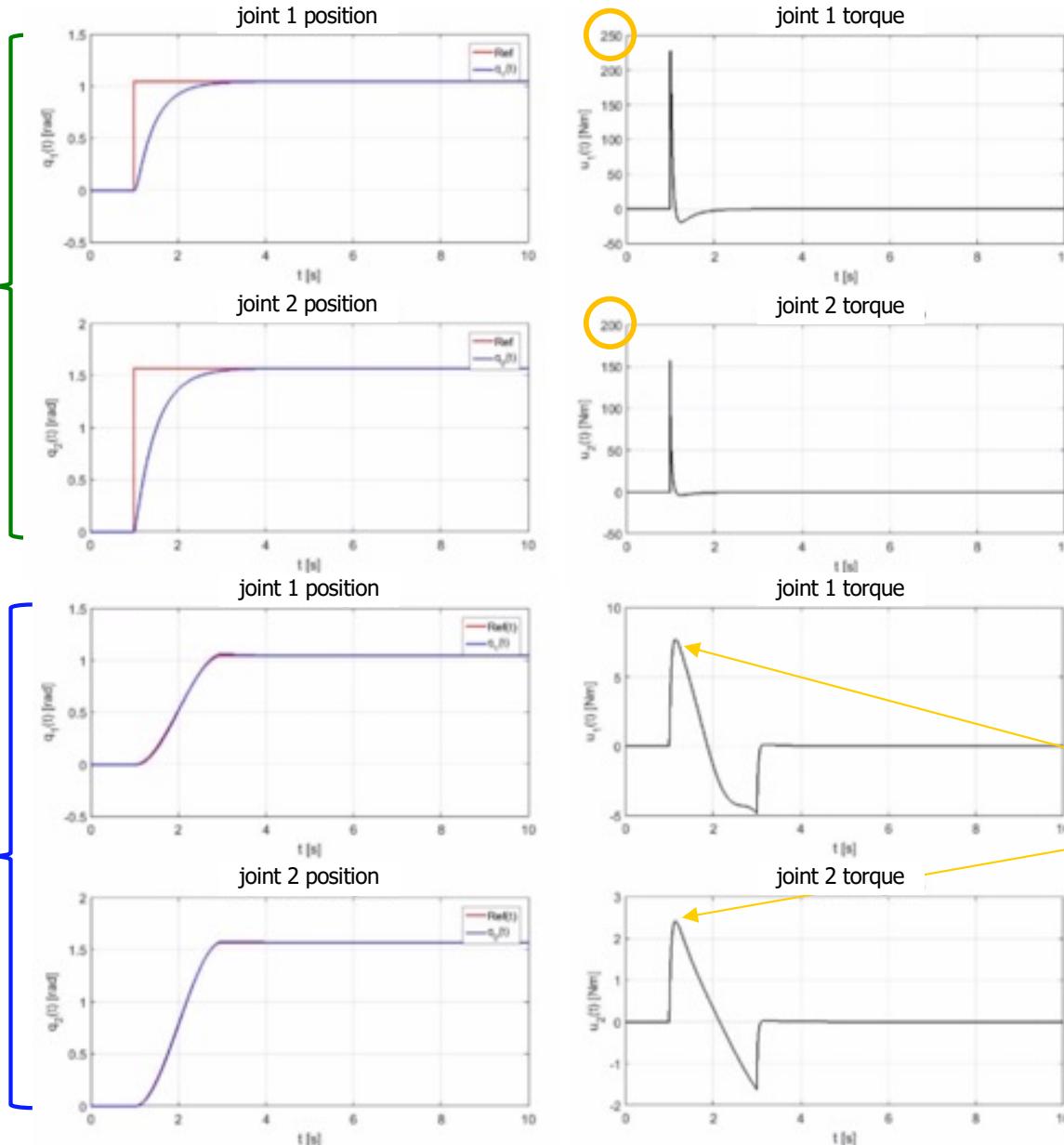
PD with same gains
 on interpolating
 trajectory of $T = 2$ s

a reduction of the
 peak control effort
 by a factor of 100
 on joint 1 &
 by a factor of 30
 on joint 2

max torques
 of 7 and 2.3 Nm

Comparison on a planar 2R arm – case b

PD with medium gains
 $K_P = \text{diag}\{200, 100\}$
 $K_D = \text{diag}\{200, 100\}$
 (very overdamped)



PD with same gains
 on interpolating
 trajectory of $T = 2$ s

even stronger
 peak reduction,
 with similar total
 control effort,
 plus improved
 tracking of
 reference trajectory
 on both joints

max torques
 of 7.5 and 2.4 Nm

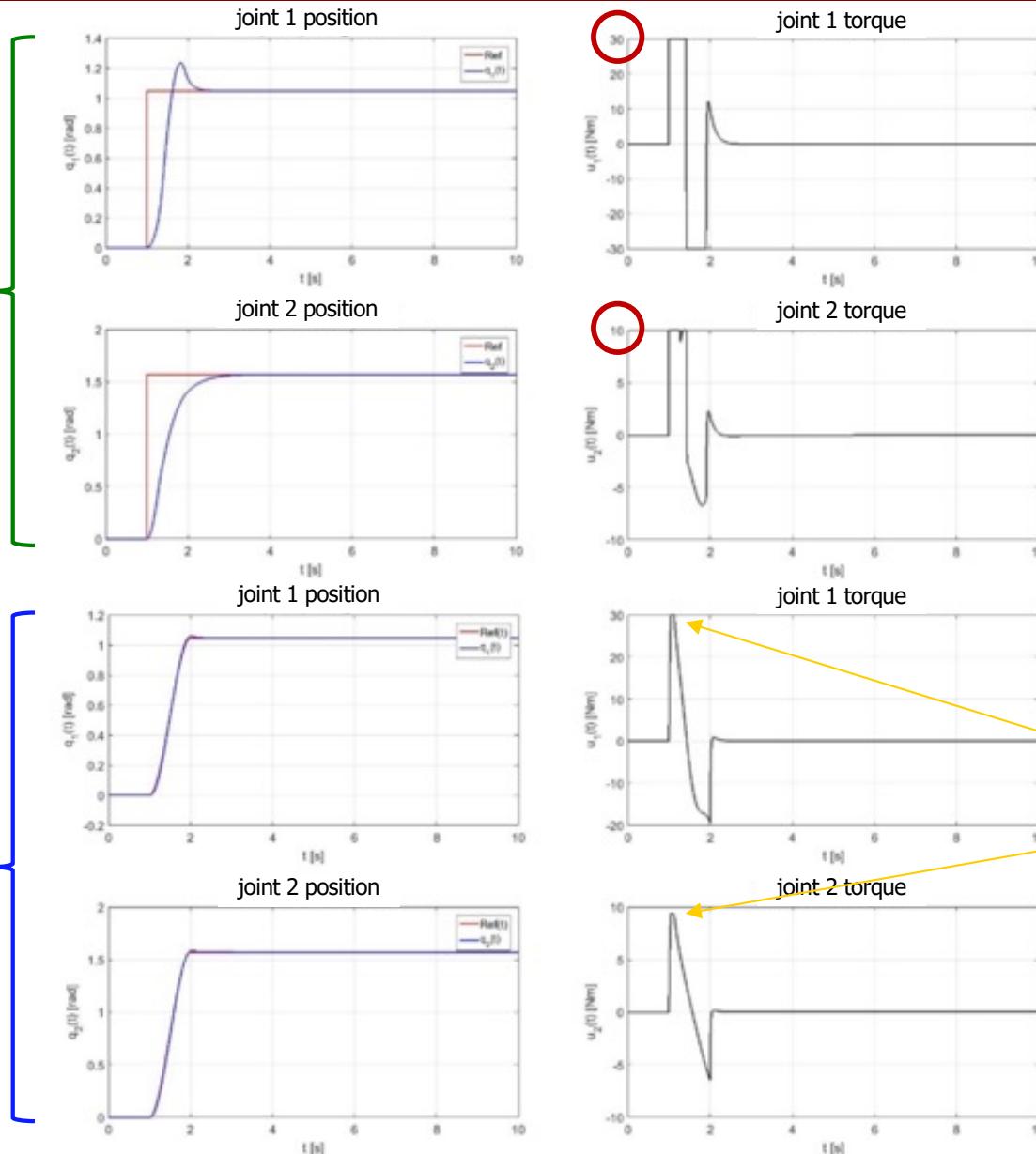


Comparison on a planar 2R arm – case c

PD with high gains
 $K_P = \text{diag}\{1250, 180\}$
 $K_D = \text{diag}\{200, 70\}$

torque saturation
 $u_{1,\max} = 30 \text{ Nm}$
 $u_{2,\max} = 10 \text{ Nm}$

PD with same gains
 on interpolating
 trajectory of $T = 1 \text{ s}$



position overshoot
 and long saturations
 are avoided,
 with **very good**
tracking of the
 faster reference
 trajectory

max torques
 of 30 and 9.5 Nm



Robotics 2

Iterative Learning for Gravity Compensation

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Control goal

- regulation of arbitrary equilibrium configurations in the presence of gravity
 - without explicit knowledge of robot dynamic coefficients (nor of the structure of the gravity term)
 - without the need of “high” position gain
 - without complex conditions on the control gains
- based on an iterative control scheme that uses
 1. PD control on joint position error + constant feedforward term
 2. iterative update of the feedforward term at successive steady-state conditions
- derive sufficient conditions for the global convergence of the iterative scheme with zero final error



Preliminaries

- robot dynamic model

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

- available bound on the gradient of the gravity term

$$\left\| \frac{\partial g(q)}{\partial q} \right\| \leq \alpha$$

- regulation attempted with a joint-based PD law (without gravity cancellation nor compensation)

$$u = K_P(q_d - q) - K_D \dot{q} \quad K_P > 0, K_D > 0$$

- at steady state, there is a **non-zero error** left (if K_P large enough)

$$q = \bar{q}, \dot{q} = 0 \quad g(\bar{q}) = K_P(q_d - \bar{q}) \quad \bar{e} = q_d - \bar{q} \neq 0$$



Iterative control scheme

- **control law** at the i -th iteration (for $i = 1, 2, \dots$)

$$u = \gamma K_P(q_d - q) - K_D \dot{q} + u_{i-1} \quad \gamma > 0$$

with a **constant compensation term** u_{i-1} (**feedforward**)

- $K_P > 0, K_D > 0$ are chosen **diagonal** for simplicity
- q_0 is the initial robot configuration
- $u_0 = 0$ is the 'easiest' initialization of the feedforward term
- at the **steady state** of the i -th iteration ($q = q_i, \dot{q} = 0$), one has

$$g(q_i) = \gamma K_P(q_d - q_i) + u_{i-1}$$

- **update law** of the compensation term (for next iteration)

$$u_i = \gamma K_P(q_d - q_i) + u_{i-1} \quad [= g(q_i)]$$

← for implementation → [for analysis]



Convergence analysis

Theorem

- IDFT

(a) $\lambda_{\min}(K_P) > \alpha$ } if K_P works it's enough to
 (b) $\gamma \geq 2$ } double it

guarantee that the sequence $\{q_0, q_1, q_2, \dots\}$ converges to q_d (and $\dot{q} = 0$) from **any** initial value q_0 (and \dot{q}_0), i.e., **globally**

- condition (a) is sufficient for the global asymptotic stability of the desired equilibrium state when using

$$u = K_P(q_d - q) - K_D \dot{q} + g(q_d)$$

with a known gravity term and diagonal gain matrices

- the additional sufficient condition (b) guarantees the convergence of the iterative scheme, yielding

$$\lim_{i \rightarrow \infty} u_i = g(q_d)$$



Proof

- let $e_i = q_d - q_i$ be the error at the end of the i -th iteration; based on the update law, it is $u_i = g(q_i)$ and thus

$$\begin{aligned}\|u_i - u_{i-1}\| &= \|g(q_i) - g(q_{i-1})\| \leq \alpha \|q_i - q_{i-1}\| \\ &\leq \alpha (\|e_i\| + \|e_{i-1}\|)\end{aligned}$$

adding and subtracting q_d

- on the other hand, from the update law it is

$$\|u_i - u_{i-1}\| = \gamma \|K_P e_i\|$$

- combining the two above relations under (a), we have

$$\gamma \alpha \|e_i\| < \gamma \lambda_{\min}(K_P) \|e_i\| \leq \gamma \|K_P e_i\| \leq \alpha (\|e_i\| + \|e_{i-1}\|)$$

$$\text{or } \|e_i\| < \frac{1}{\gamma} (\|e_i\| + \|e_{i-1}\|)$$



Proof (cont)

- condition (b) guarantees that the error sequence $\{e_0, e_1, e_2, \dots\}$

$$\|e_i\| < \frac{\frac{1}{\gamma}}{1 - \frac{1}{\gamma}} \|e_{i-1}\| = \frac{1}{\gamma - 1} \|e_{i-1}\|$$

is a **contraction mapping**, so that

$$\lim_{i \rightarrow \infty} \|e_i\| = 0$$

with asymptotic convergence from any initial state



⇒ the robot progressively approaches the desired configuration through successive steady-state conditions

- K_P and K_D affect each transient phase
- coefficient γ drives the convergence rate of intermediate steady states to the final one



Remarks

- combining (a) and (b), the sufficient condition only requires the **doubling** of the proportional gain w.r.t. the known gravity case

$$\widehat{K}_P = \gamma K_P \quad \rightarrow \quad \lambda_{\min}(\widehat{K}_P) > 2\alpha$$

- for a diagonal \widehat{K}_P , this condition implies a (positive) lower bound on the single diagonal elements of the matrix
- again, it is only a **sufficient** condition
 - the scheme may converge even if this condition is violated ...
- the scheme can be interpreted as using an **integral term**
 - updated only in correspondence of a **discrete sequence of time instants**
 - with a guaranteed **global** convergence (and implicit stability)



Numerical results

- 3R robot with uniform links, moving in the vertical plane

$$l_1 = l_2 = l_3 = 0.5 \text{ [m]}$$

$$m_1 = 30, m_2 = 20, m_3 = 10 \text{ [kg]} \quad \rightarrow \quad \boxed{\alpha \simeq 400}$$

- with saturations of the actuating torques

$$U_{1,\max} = 800, U_{2,\max} = 400, U_{3,\max} = 200 \text{ [Nm]}$$

- three cases, from the downward position $q_0 = (0, 0, 0)$

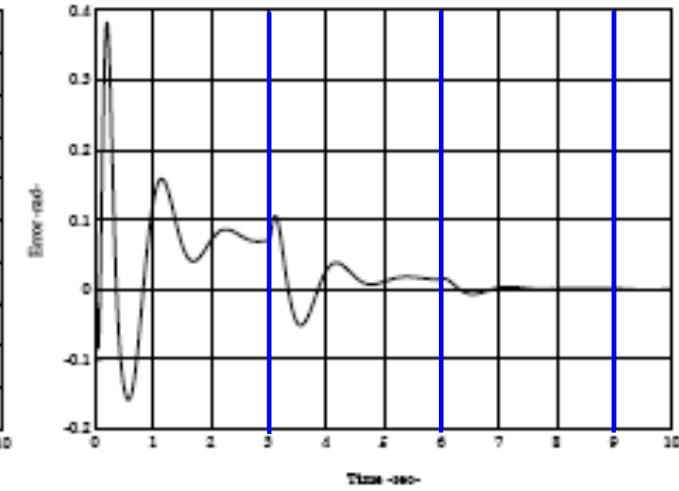
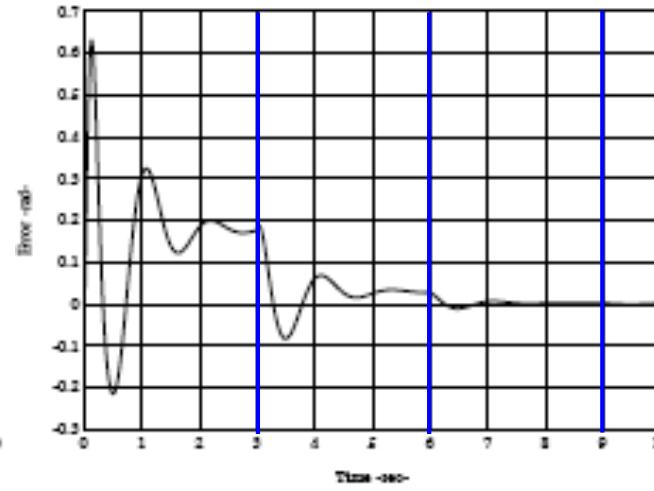
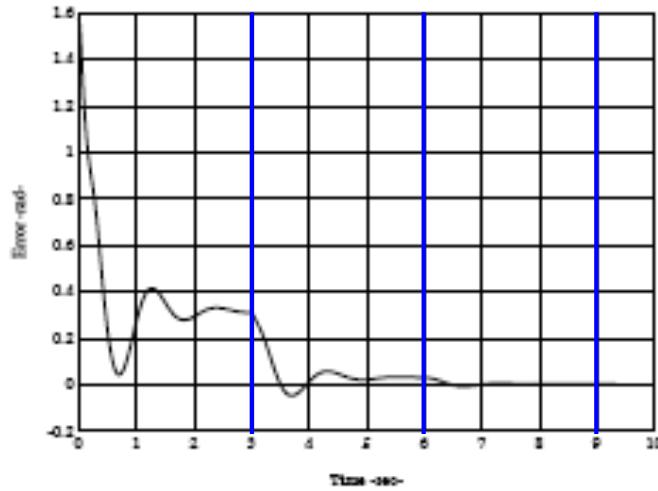
$$\text{I: } q_d = (\pi/2, 0, 0) \quad \left. \right\} \quad \left\{ \begin{array}{l} \hat{K}_P = \text{diag}\{1000, 600, 280\} \\ K_D = \text{diag}\{200, 100, 20\} \end{array} \right.$$

$$\text{II: } q_d = (3\pi/4, 0, 0) \quad \left. \right\} \quad \left\{ \begin{array}{l} \hat{K}_P = \text{diag}\{1000, 600, 280\} \\ K_D = \text{diag}\{200, 100, 20\} \end{array} \right.$$

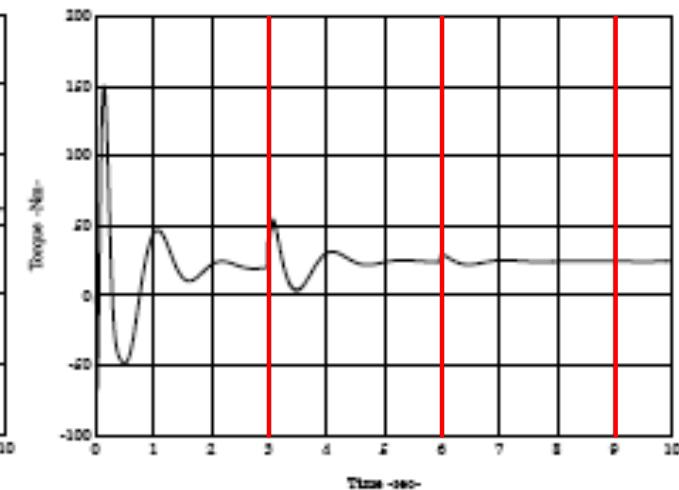
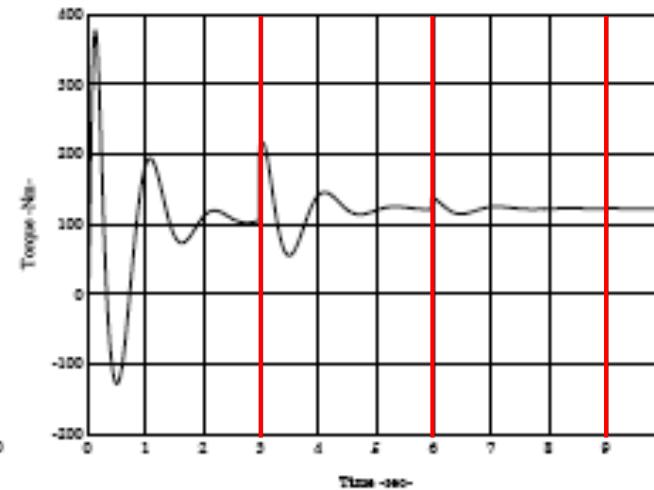
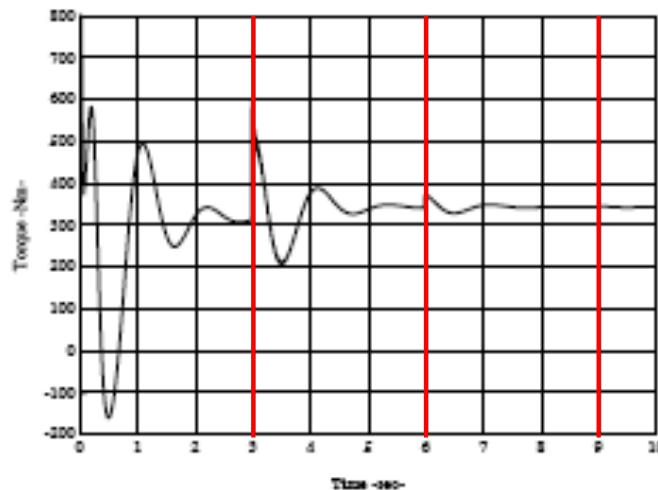
$$\text{III: } q_d = (3\pi/4, 0, 0) \quad \left. \right\} \quad \left\{ \begin{array}{l} \hat{K}_P = \text{diag}\{500, 500, 500\} \\ K_D = \text{as before} \end{array} \right.$$



Case I: $q_d = (\pi/2, 0, 0)$



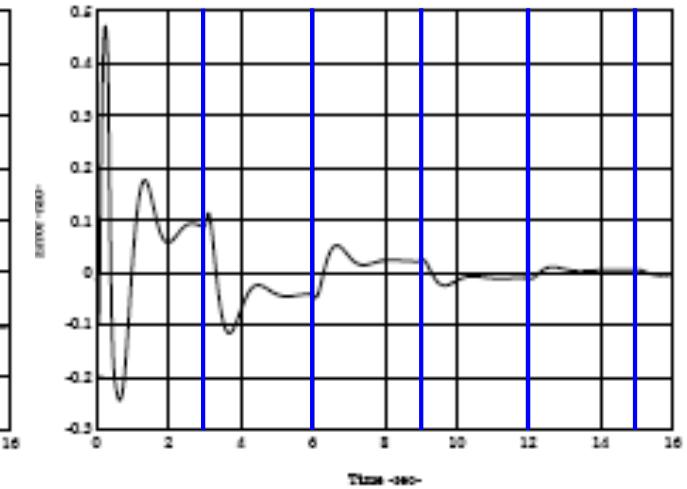
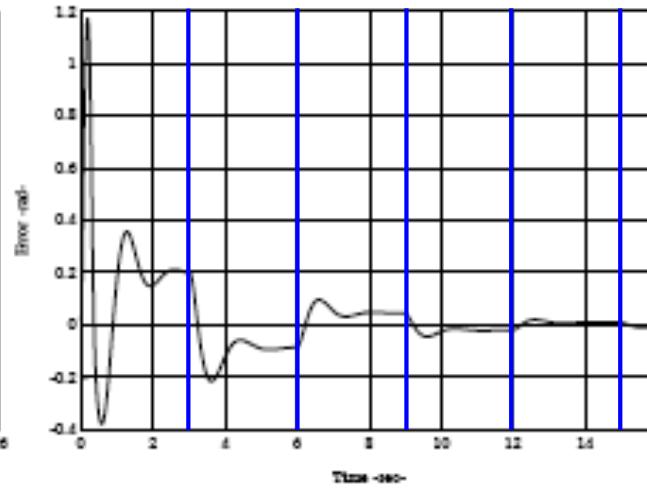
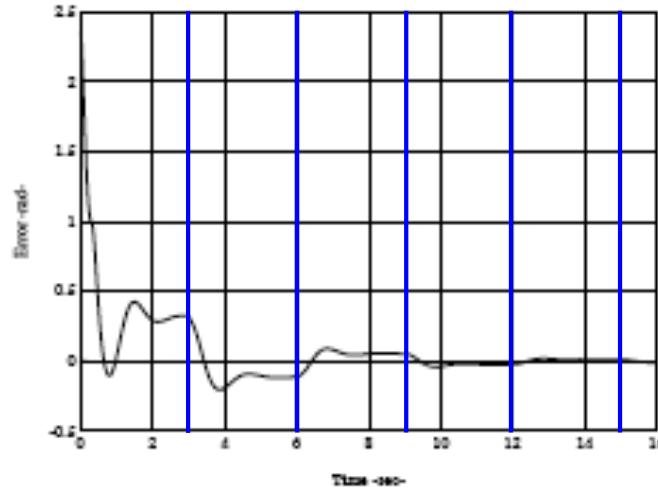
joint position errors (zero after 3 updates)



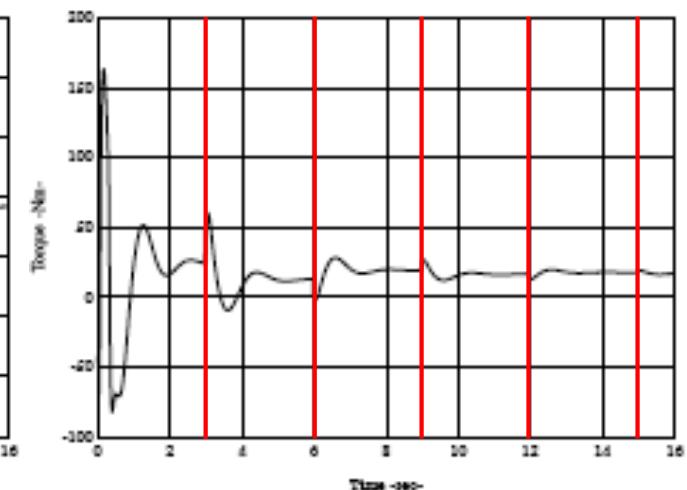
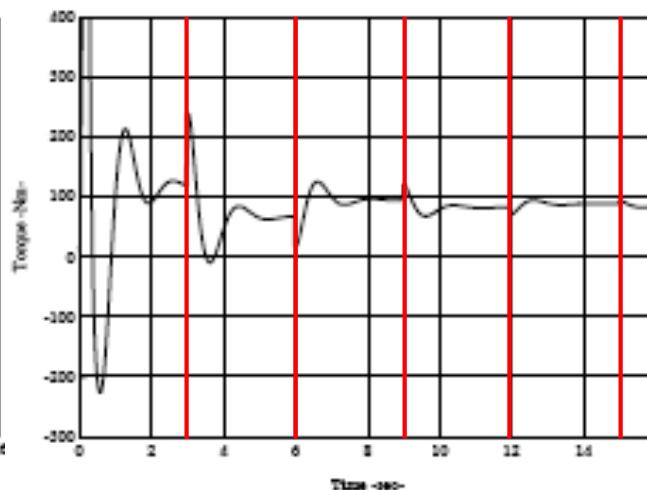
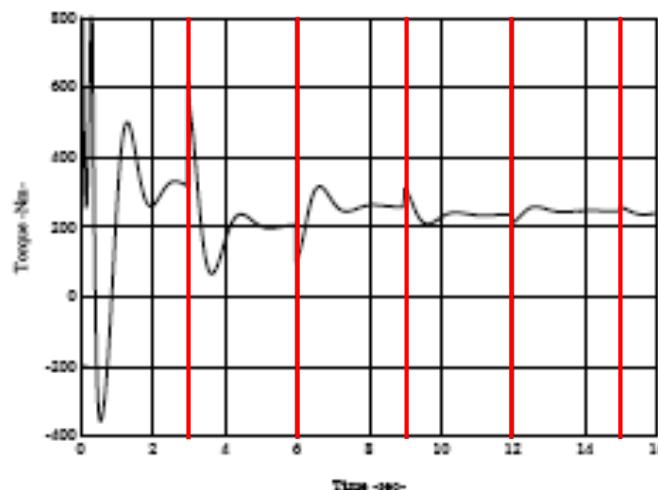
control torques



Case II: $q_d = (3\pi/4, 0, 0)$



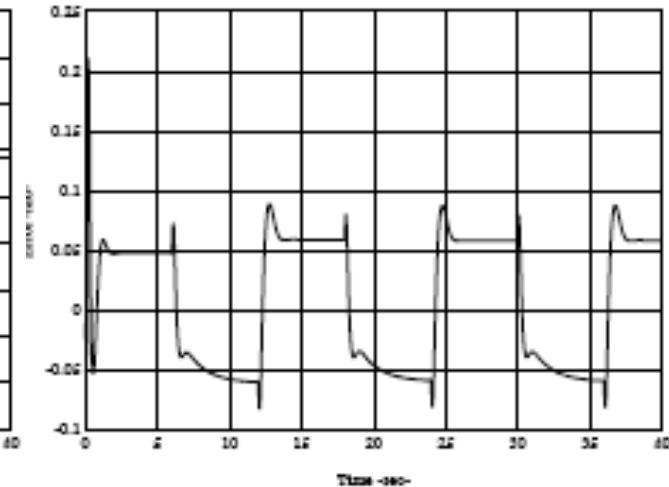
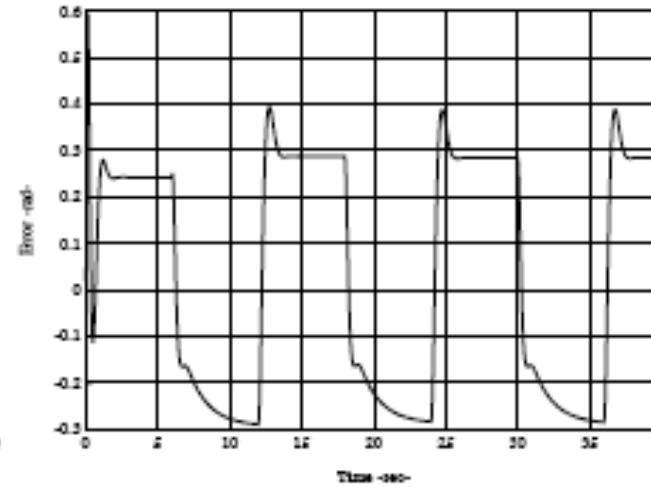
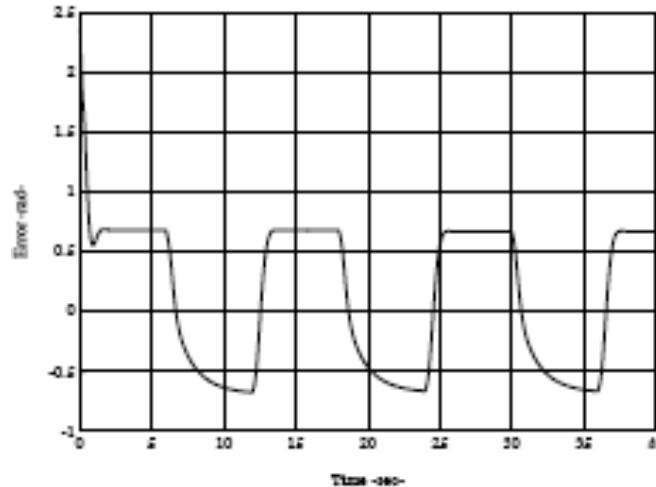
joint position errors (zero after 5 updates)



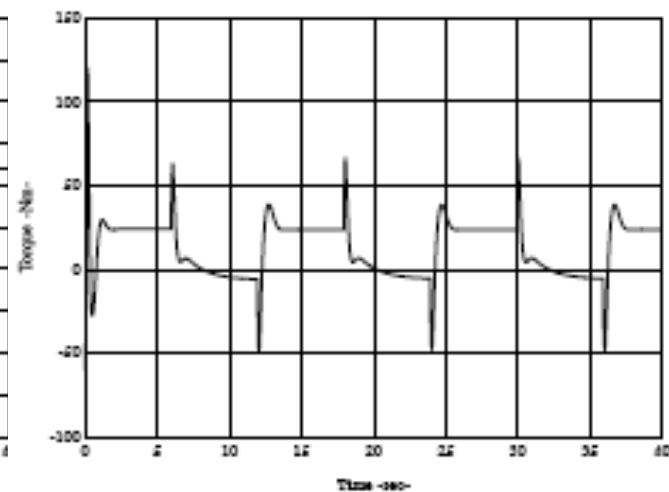
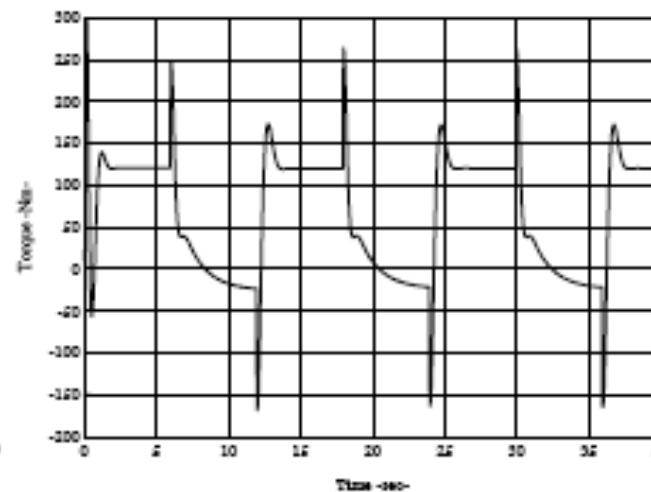
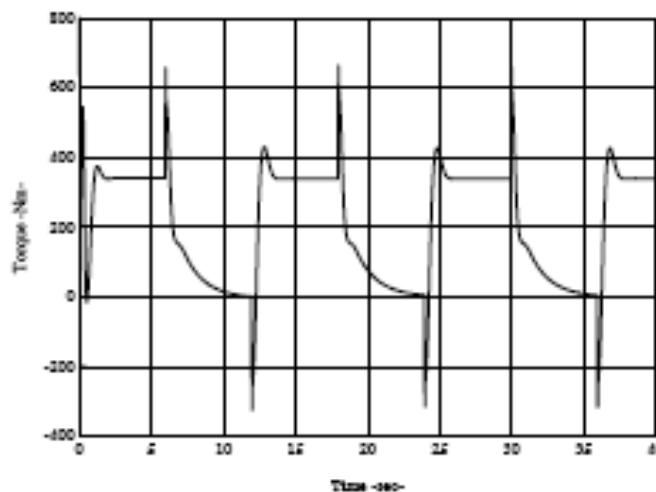
control torques



Case III: $q_d = (3\pi/4, 0, 0)$, reduced gains



joint position errors (limit cycles, no convergence!)



control torques



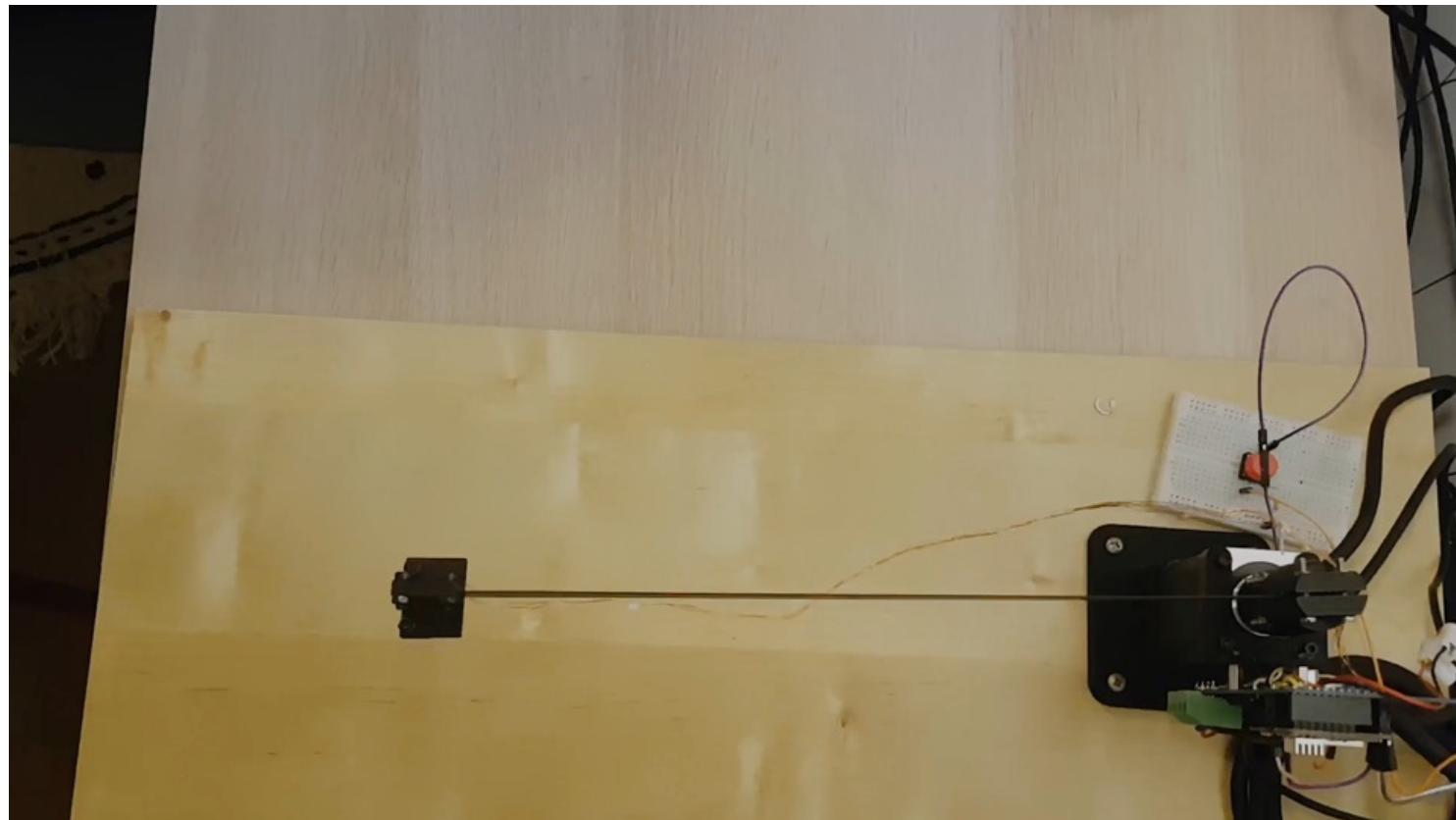
Final comments

- only **few iterations** are needed for obtaining convergence, learning the correct gravity compensation at the desired q_d
- Sufficiency of the condition on the P gain
 - even if violated, convergence can still be obtained (first two cases); otherwise, a limit motion cycle takes place between two equilibrium configurations that are both incorrect (as in the third case)
 - this shows 'how far' is sufficiency from necessity
- analysis can be refined to get lower bounds on the K_{P_i} (diagonal case) that are smaller, but still sufficient for convergence
 - intuitively, lower values for K_{P_i} should still work for distal joints
- in practice, update of the feedforward term occurs when the robot is close enough to a steady state (joint velocities and position variations are below suitable thresholds)



Control experiments with flexible robots without gravity

even for just a **single** but **flexible link** in the absence of gravity, a **rest-to-rest** maneuver **without residual oscillations** is difficult to be performed by a **pure PD joint control** action



[video](#)

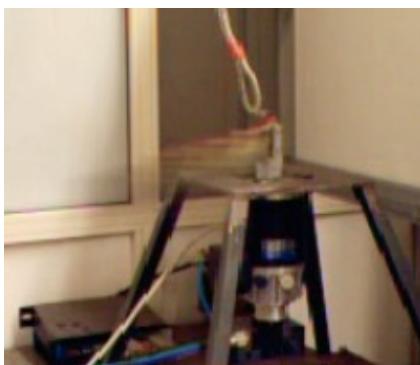
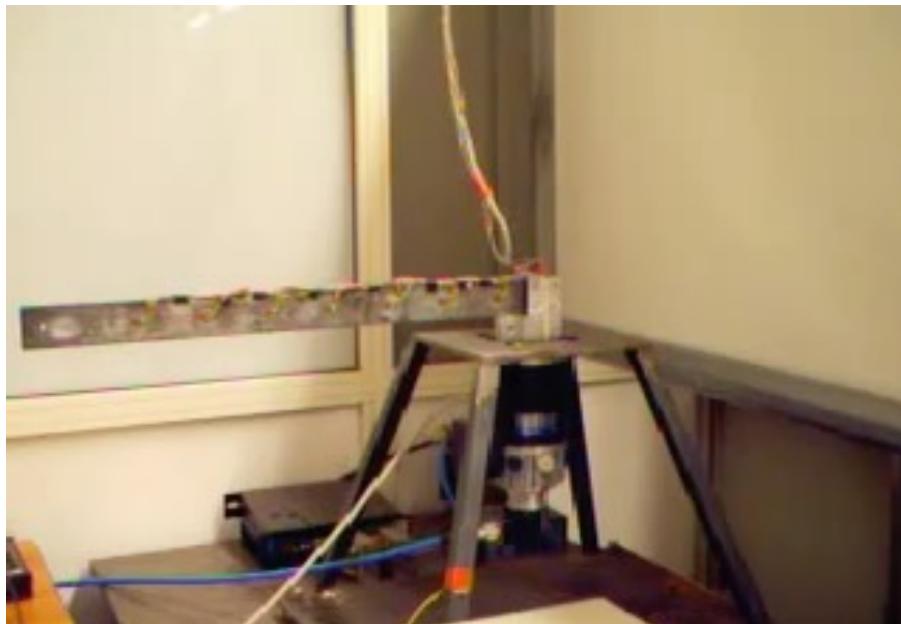
ICRA 2023

S. Drost. P. Pustina. F. Angelini, A. De Luca, G. Smit, C. Della Santina,
"Experimental validation of functional iterative learning control on a one-link flexible arm"



Control experiments with flexible robots without gravity

video



rest-to-rest maneuver in given motion time
for a single flexible link (PD + feedforward)

video

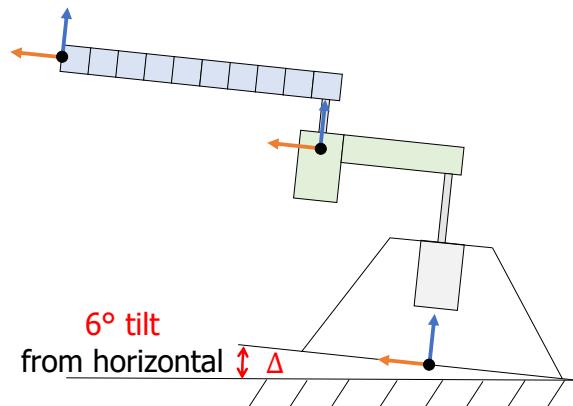


end-effector trajectory tracking for FlexArm
—a planar 2R robot with flexible forearm

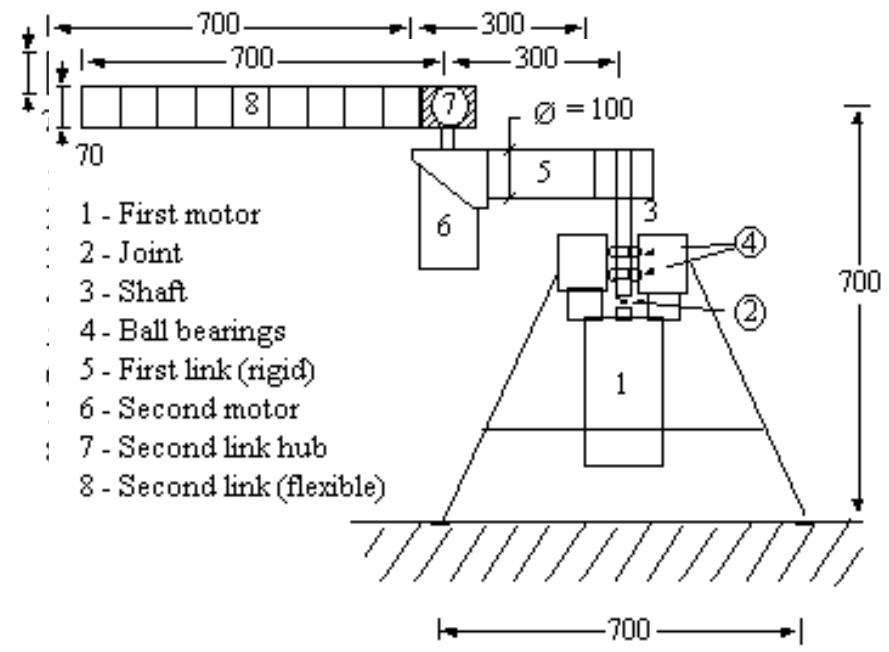
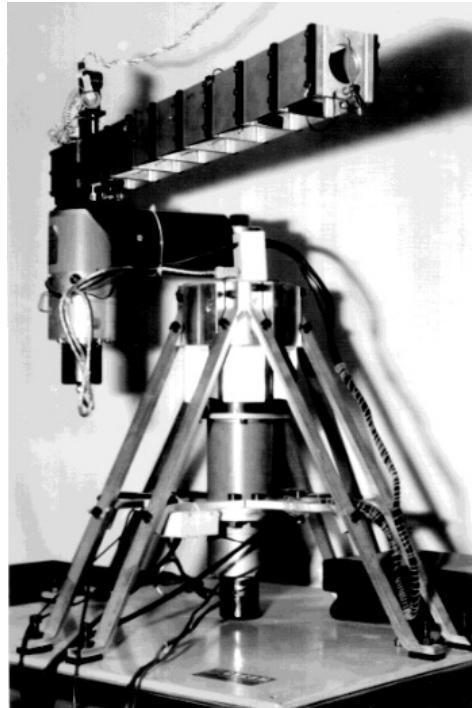


Extension to flexible robots

- the same iterative learning control approach has been extended to position regulation in robots with **flexible joints and/or links** under gravity
 - at the motor/joint level
 - at the Cartesian level (end-effector tip position, **beyond flexibility**), using a **double iterative scheme**
- experimentally validated on the **two-link FlexArm @ DIS** (now DIAG!)

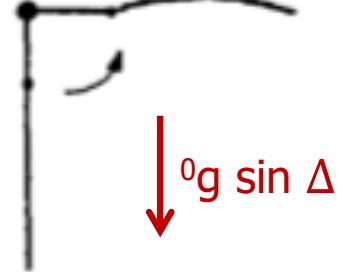


with supporting base
tilted by $\Delta \approx 6^\circ$
(inclusion of gravity)

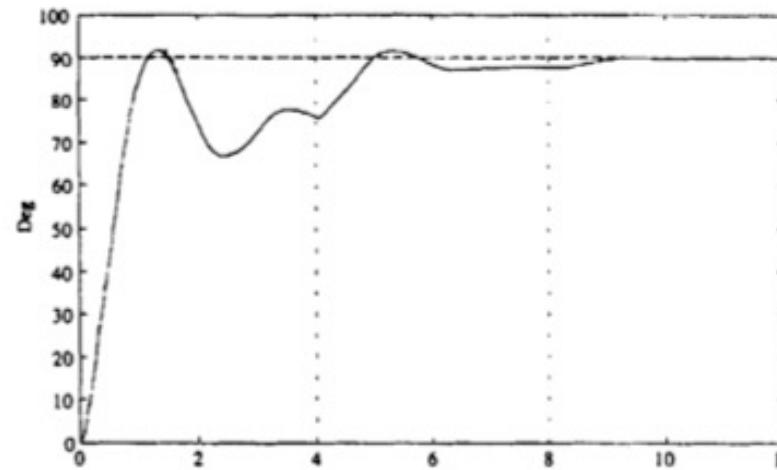




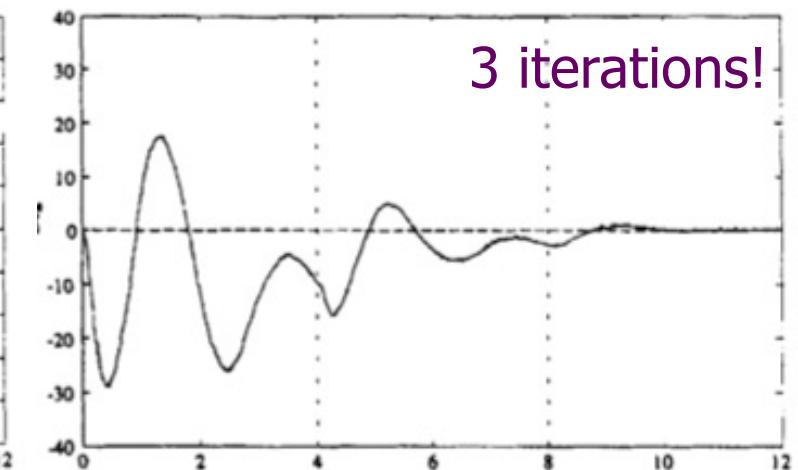
Experimental results for tip regulation



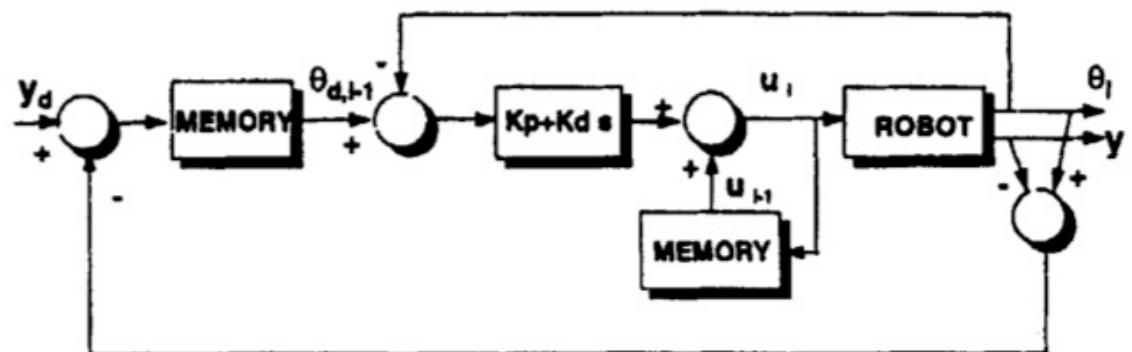
motion task:
 $(0^\circ, 0^\circ) \Rightarrow (90^\circ, 0^\circ)$



first link position

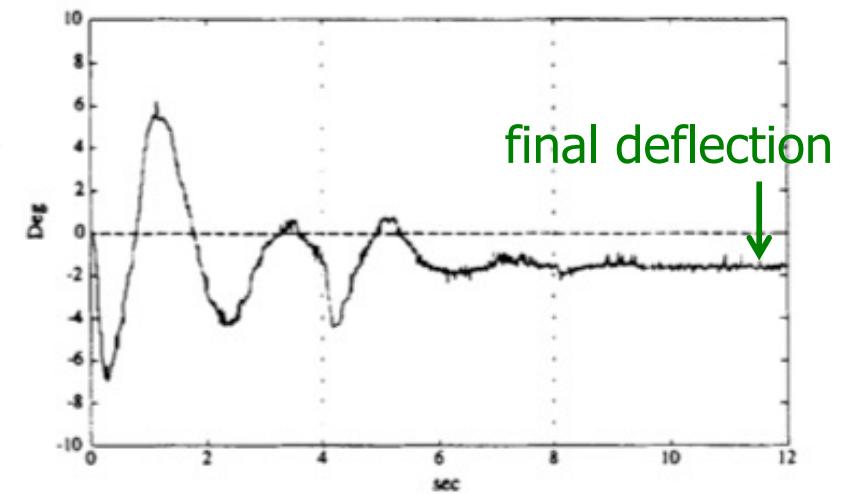


tip angle w.r.t. first link



double iterative scheme

De Luca, Panzieri: Int J Adapt Cont & Sign Proc, 1996
 (factor $\gamma \rightarrow 1/\beta$ in the original paper)



second link deflection



Robotics 2

Trajectory Tracking Control

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Inverse dynamics control

given the robot **dynamic model** with N joints

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

Symbolic
Lagrangian
form

$\overbrace{c(q, \dot{q}) + g(q)} + \text{friction model}$

and a twice-differentiable **desired trajectory** for $t \in [0, T]$

$$q_d(t) \rightarrow \dot{q}_d(t), \ddot{q}_d(t)$$

applying the **feedforward** torque in **nominal conditions**

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

simply computations
with given q_d, \dots

yields exact reproduction of the desired motion, provided that $q(0) = q_d(0), \dot{q}(0) = \dot{q}_d(0)$ (initial **matched state**)



In practice ...

never reaches
desired trajectory

a number of differences from the nominal condition

- initial state is “**not matched**” to the desired trajectory $q_d(t)$
- **disturbances** on the actuators, from unexpected collisions, truncation errors on data, ...
- **inaccurate knowledge** of robot dynamic parameters $\pi \rightarrow \hat{\pi}$ (link masses, inertias, center of mass positions)
- **unknown** value of the carried payload
- presence of **unmodeled** dynamics (complex friction phenomena, transmission elasticity, ...)



require the use of **feedback information**



most accurate estimation of
values obtained with
Offline procedure

Introducing feedback

$$\hat{u}_d = \hat{M}(q_d)\ddot{q}_d + \hat{n}(q_d, \dot{q}_d)$$

with \hat{M} , \hat{n} estimates of terms
(or coefficients) in the dynamic model

note: \hat{u}_d can be computed off line [e.g., by $\hat{NE}_\alpha(q_d, \dot{q}_d, \ddot{q}_d)$]

feedback is introduced to make the control scheme more robust

different possible implementations depending on
amount of computational load share

- OFF LINE (↔ open loop)
- ON LINE (↔ closed loop)

two-step control design:

1. compensation (feedforward) or cancellation (feedback) of nonlinearities
2. synthesis of a linear control law stabilizing the trajectory error to zero



A series of trajectory controllers

(assuming the nominal case: $\hat{M} = M, \hat{n} = n$)

positive def.
matrices
(no special
need of be
diagonal)

1. inverse dynamics compensation (FFW) + PD

$$u = \hat{u}_d + K_P(q_d - q) + K_D(\dot{q}_d - \dot{q})$$

Proportional Derivative local stabilization
of trajectory error

$$e(t) = q_d(t) - q(t)$$

global if additional
conditions on
 K_P and K_D

2. inverse dynamics compensation (FFW) + variable PD

$$u = \hat{u}_d + \hat{M}(q_d)[K_P(q_d - q) + K_D(\dot{q}_d - \dot{q})]$$

3. feedback linearization (FBL) + [PD+FFW] = "COMPUTED TORQUE"

$$u = \hat{M}(q)[\ddot{q}_d + K_P(q_d - q) + K_D(\dot{q}_d - \dot{q})] + \hat{n}(q, \dot{q})$$

4. feedback linearization (FBL) + [PID+FFW]

$$u = \hat{M}(q) \left[\ddot{q}_d + K_P(q_d - q) + K_D(\dot{q}_d - \dot{q}) + K_I \int (q_d - q) dt \right] + \hat{n}(q, \dot{q})$$

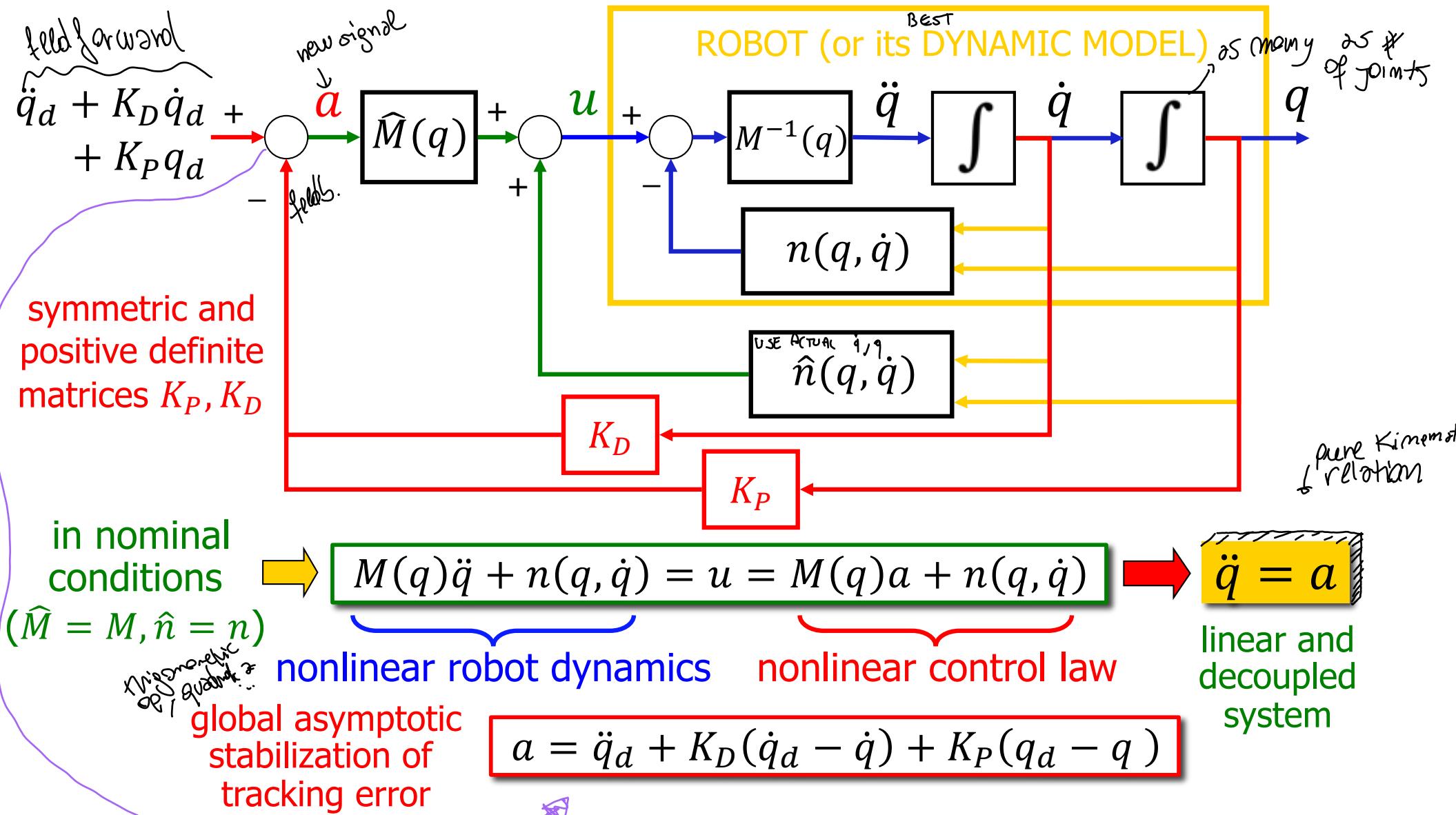
no more non
linear terms
→ (not nonlinear)

global stabilization for any $K_P > 0, K_D > 0$ (and not too large $K_I > 0$)

more robust to small uncertainties/disturbances, even if more complex to implement in real time



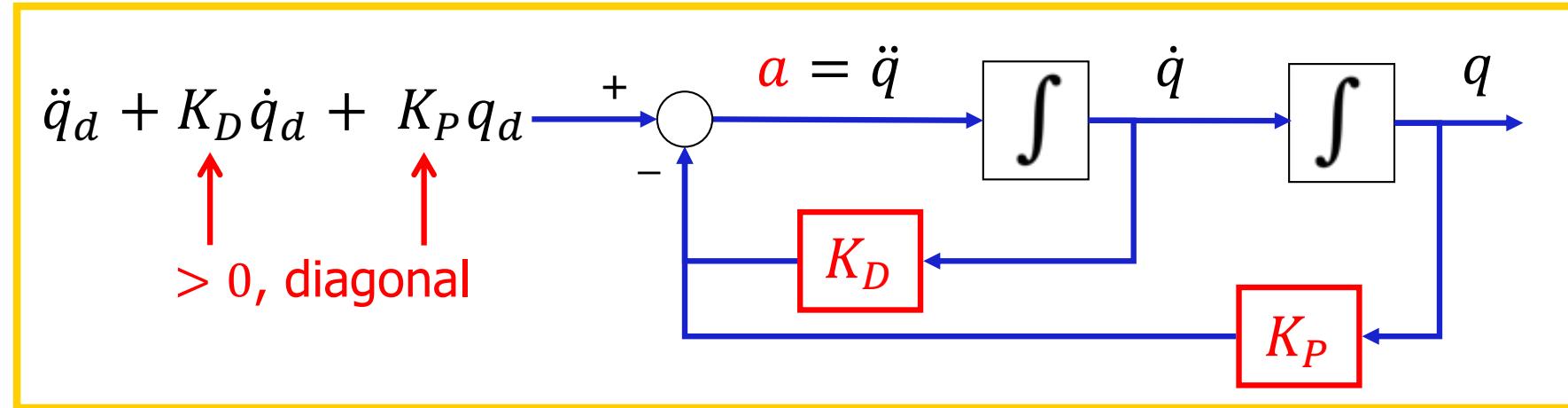
Feedback linearization control





Lyssam

Interpretation in the linear domain



under feedback linearization control, the robot has a dynamic behavior that is **invariant**, **linear** and **decoupled** in its whole state space ($\forall(q, \dot{q})$)

linearity

a unitary mass ($m = 1$) in the joint space !!

error transients $e_i = q_{di} - q_i \rightarrow 0$ exponentially, prescribed by K_{Pi}, K_{Di} choice

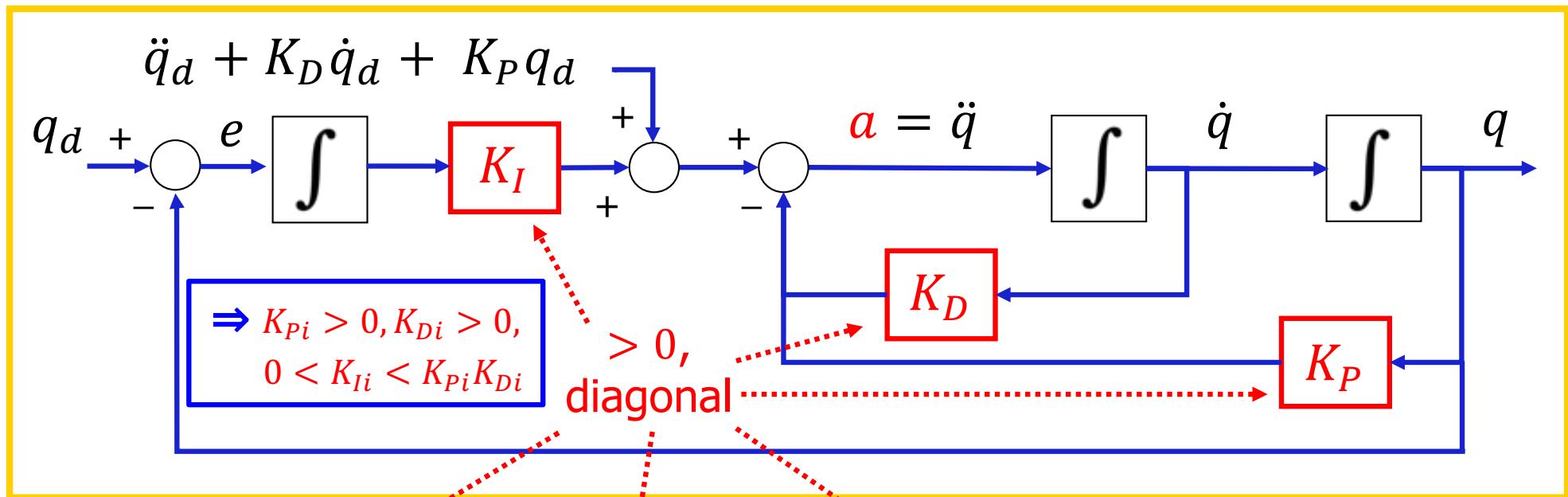
decoupling

each joint coordinate q_i evolves **independently** from the others, forced by a_i

$$\ddot{e} + K_D \dot{e} + K_P e = 0 \Leftrightarrow \ddot{e}_i + K_{Di} \dot{e}_i + K_{Pi} e_i = 0$$



Addition of an integral term: PID whiteboard...



$$\ddot{q} = \mathbf{a} = \ddot{q}_d + K_D(\dot{q}_d - \dot{q}) + K_P(q_d - q) + K_I \int (q_d - q) d\tau \quad e = q_d - q$$

$$\Rightarrow (1) \quad e_i = q_{di} - q_i \quad (i = 1, \dots, N) \quad \xrightarrow{\text{expand}} \quad (2) \quad \ddot{e}_i + K_{Di} \dot{e}_i + K_{Pi} e_i + K_{Ii} \int e_i d\tau = 0$$

LINEAR
DIFF EQ
CAN USE
ZERFACE

$$\mathcal{L}[e_i(t)] \Rightarrow (3) \quad \left(s^2 + K_{Di}s + K_{Pi} + K_{Ii} \frac{1}{s} \right) e_i(s) = 0$$

$$s \times \Rightarrow (4) \quad (s^3 + K_{Di}s^2 + K_{Pi}s + K_{Ii})e_i(s) = 0 \quad \xrightarrow{\text{in that way poly in } s} \quad (5)$$

$$\xrightarrow{\text{can use easy ratioth}}$$

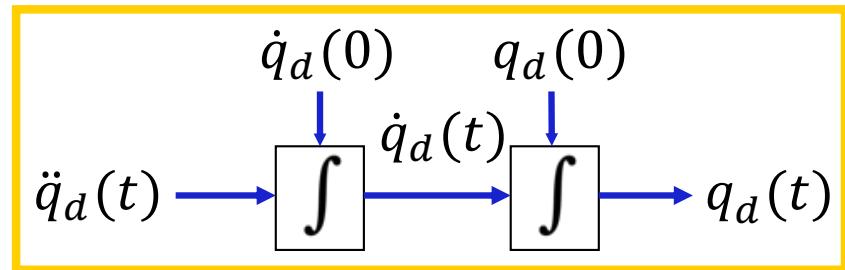
	3	1	K_{Pi}
	2	K_{Di}	K_{Ii}
	1	$(K_{Di}K_{Pi} - K_{Ii})/K_{Di}$	
	0	K_{Ii}	

⇒ (6)
exponential
stability
conditions by
Routh criterion



Remarks

- desired joint trajectory can be generated from **Cartesian** data



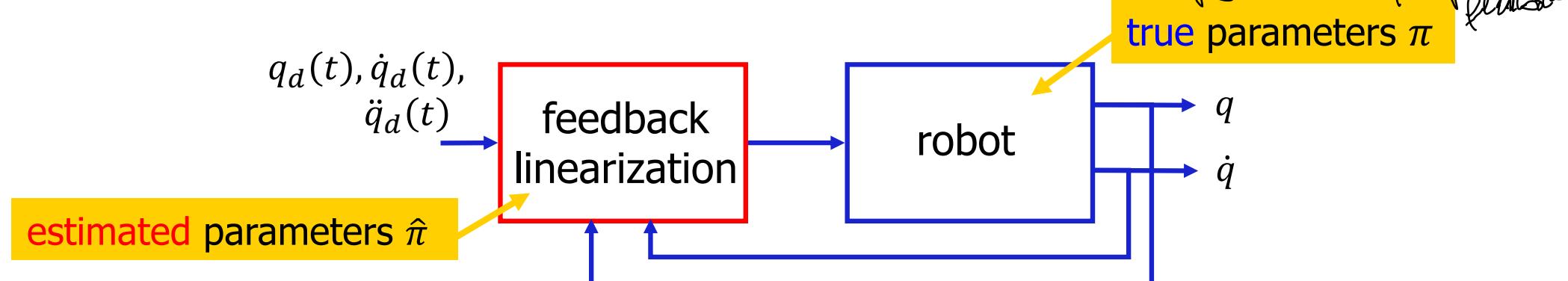
$$\ddot{p}_d(t), \dot{p}_d(0), p_d(0)$$

$$q_d(0) = f^{-1}(p_d(0))$$

$$\dot{q}_d(0) = J^{-1}(q_d(0))\dot{p}_d(0)$$

$$\ddot{q}_d(t) = J^{-1}(q_d)[\ddot{p}_d(t) - J(q_d)\dot{q}_d]$$

- real-time computation by **Newton-Euler** algo: $u_{FBL} = \widehat{NE}_\alpha(\underline{q}, \dot{\underline{q}}, \underline{a})$
- simulation** of feedback linearization control

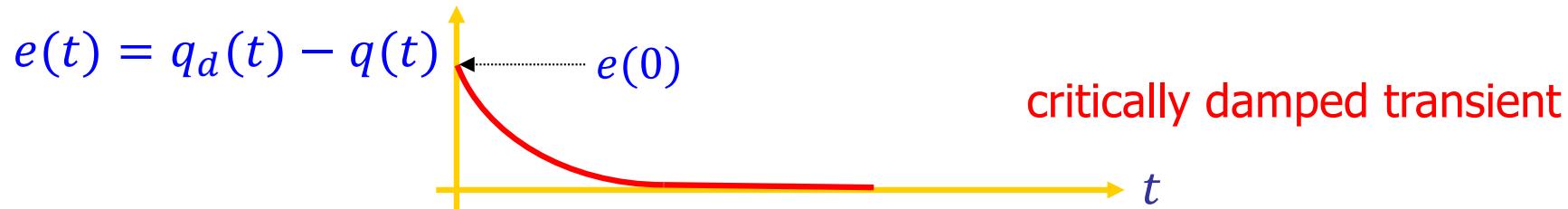


Hint: there is no use in simulating this control law in the ideal case ($\hat{\pi} = \pi$); robot behavior will be **identical** to the linear and decoupled case of **stabilized double integrators!!**



Further comments

- choice of the diagonal elements of K_P, K_D (and K_I)
 - shaping the error transients, with an eye also to motor saturations...



- parametric identification
 - to be done in advance, using the property of linearity in the dynamic coefficients of the robot dynamic model
- choice of the sampling time of a digital implementation
 - compromise between computational time and tracking accuracy, typically $T_c = 0.5 \div 10$ ms
- exact linearization by (state) feedback is a general technique of nonlinear control theory
 - can be used for robots with elastic joints, wheeled mobile robots, ...
 - non-robotics applications: satellites, induction motors, helicopters, ...



Another example of feedback linearization design

- dynamic model of robots with elastic joints

- q = link position
- θ = motor position (after reduction gears)
- B_m = diagonal matrix (> 0) of inertia of the (balanced) motors
- K = diagonal matrix (> 0) of (finite) stiffness of the joints

$4N$ state variables
 $x = (q, \theta, \dot{q}, \dot{\theta})$

$$\left\{ \begin{array}{l} M(q)\ddot{q} + c(q, \dot{q}) + g(q) + K(q - \theta) = 0 \\ B_m\ddot{\theta} + K(\theta - q) = u \end{array} \right. \quad \begin{array}{l} (1) \\ (2) \end{array}$$

Link dynamics equation

- is there a control law that achieves exact linearization via feedback?

$$u = \alpha(q, \theta, \dot{q}, \dot{\theta}) + \beta(q, \theta, \dot{q}, \dot{\theta}) a$$

YES and it yields $\frac{d^4 q_i}{dt^4} = a_i, \quad i = 1, \dots, N$

linear and decoupled system:
 N chains of 4 integrators (not clear)
(to be stabilized by linear control design)

Hint: differentiate (1) w.r.t. time until motor acceleration $\ddot{\theta}$ appears;
substitute this from (2); choose u so as to cancel all nonlinearities ...



Surveillance
Control
Controllability

Alternative global trajectory controller

$$u = M(q)\ddot{q}_d + S(q, \dot{q})\dot{q}_d + g(q) + F_V\dot{q}_d + K_P e + K_D \dot{e}$$



SPECIAL factorization such that
 $\dot{M} - 2S$ is skew-symmetric



symmetric and
positive definite matrices

- **global asymptotic stability** of $(e, \dot{e}) = (0,0)$ (trajectory tracking)
 - we can use LaSalle if time varying closed loop
- proven by Lyapunov +Barbalat (time-varying system) +LaSalle
- does **not** produce a complete **cancellation** of nonlinearities
 - the variables \dot{q} and \ddot{q} that appear **linearly** in the model are evaluated on the **desired** trajectory → replace everywhere \dot{q}, \ddot{q}
- does **not** induce a **linear** and **decoupled** behavior of the trajectory error $e(t) = q_d(t) - q(t)$ in the closed-loop system
- however, it lends itself more easily to an **adaptive version**
- **computation:** by **4×** standard or **1×** modified NE algorithm



Analysis of asymptotic stability of the trajectory error - 1

$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) + F_V\dot{q} = u$ robot dynamics (including friction)

control law $u = M(q)\ddot{q}_d + S(q, \dot{q})\dot{q}_d + g(q) + F_V\dot{q}_d + K_P e + K_D \dot{e}$

- Lyapunov candidate and its time derivative (with $e = q_d - q$)

$$V = \frac{1}{2}\dot{e}^T M(q)\dot{e} + \frac{1}{2}e^T K_P e \geq 0 \Rightarrow \dot{V} = \frac{1}{2}\dot{e}^T \dot{M}(q)\dot{e} + \dot{e}^T M(q)\ddot{e} + e^T K_P \dot{e}$$

- the closed-loop system equations yield

$$M(q)\ddot{e} = -S(q, \dot{q})\dot{e} - (K_D + F_V)\dot{e} - K_P e \quad \text{explicit by eq.}$$

- substituting and using the skew-symmetric property of $\dot{M} - 2S$

$$\dot{V} = -\dot{e}^T (K_D + F_V)\dot{e} \leq 0 \quad \dot{V} = 0 \Leftrightarrow \dot{e} = 0$$

- since the system is time-varying (due to $q_d(t)$), direct application of LaSalle theorem is NOT allowed \Rightarrow use Barbalat lemma...

$$q = q_d(t) - e, \dot{q} = \dot{q}_d(t) - \dot{e} \Rightarrow V = V(e, \dot{e}, t) = V(x, t)$$

error state x

⇒ go to
slide 10 in
block 8



Analysis of asymptotic stability of the trajectory error - 2

- since i) V is lower bounded and ii) $\dot{V} \leq 0$, we have to check only condition iii) in order to apply Barbalat lemma

$$\ddot{V} = -2\dot{e}^T(K_D + F_V)\ddot{e} \quad \dots \text{is this bounded?}$$

- from i) + ii), V is bounded $\Rightarrow e$ and \dot{e} are bounded
- assume that the desired trajectory has bounded velocity \dot{q}_d
- using the following two properties of dynamic model terms

$$0 < \alpha_m \leq \|M^{-1}(q)\| \leq \alpha_M < \infty \quad \|S(q, \dot{q})\| \leq \alpha_S \|\dot{q}\|$$

then also \ddot{e} will be bounded (in norm) since

$$\ddot{e} = -M^{-1}(q)[S(q, \dot{q})\dot{e} + K_P e + (K_D + F_V)\dot{e}]$$

$$\begin{array}{c} \uparrow \\ \text{bounded} \\ \text{in norm by } \alpha_M \end{array} \quad \begin{array}{c} \uparrow \\ \text{bounded} \\ \text{in norm by } \alpha_S \|\dot{q}\| \end{array} \quad \begin{array}{c} \uparrow \\ \text{bounded} \end{array} \quad \begin{array}{c} \uparrow \\ \text{bounded} \end{array}$$

$\left. \begin{array}{l} \text{from i) + ii), } V \text{ is bounded} \\ \text{assume that the desired trajectory has bounded velocity } \dot{q}_d \end{array} \right\} \Rightarrow \begin{array}{l} \dot{q} \text{ is} \\ \text{bounded} \end{array}$

$$\left. \begin{array}{l} \text{using the following two properties of dynamic model terms} \\ \text{then also } \ddot{e} \text{ will be bounded (in norm) since} \\ \ddot{e} = -M^{-1}(q)[S(q, \dot{q})\dot{e} + K_P e + (K_D + F_V)\dot{e}] \\ \uparrow \\ \text{bounded} \\ \text{in norm by } \alpha_M \\ \uparrow \\ \text{bounded} \\ \text{in norm by } \alpha_S \|\dot{q}\| \\ \uparrow \\ \text{bounded} \end{array} \right\} \Rightarrow \lim_{t \rightarrow \infty} \dot{V}(t) = 0$$



Analysis of asymptotic stability of the trajectory error – end of proof

- we can conclude by proceeding as in LaSalle theorem

$$\dot{V} = 0 \Leftrightarrow \dot{e} = 0$$

- the closed-loop dynamics in this situation is

$$M(q)\ddot{e} = -K_P e$$

$$\Rightarrow \ddot{e} = 0 \Leftrightarrow e = 0 \quad \rightarrow \quad (e, \dot{e}) = (0, 0)$$

is the largest
invariant set in $\dot{V} = 0$

\rightarrow (global) asymptotic tracking
will be achieved





Regulation as a special case

- what happens to the control laws designed for trajectory tracking when q_d is **constant**? are there simplifications?
- **feedback linearization**

$$u = M(q)[K_P(q_d - q) - K_D \dot{q}] + c(q, \dot{q}) + g(q)$$

- no special simplifications
- however, this is a solution to the regulation problem with **exponential stability** (and decoupled transients at each joint!)
- **alternative global controller**

$$u = K_P(q_d - q) - K_D \dot{q} + g(q)$$

- we recover the simpler PD + gravity cancellation control law!!



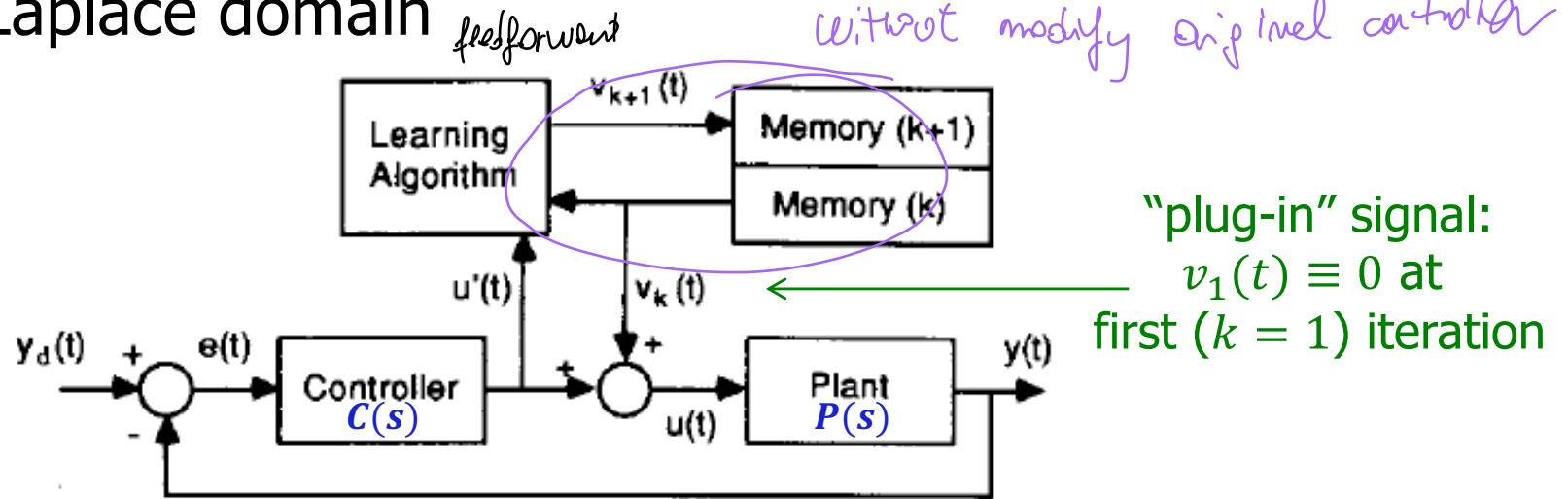
Trajectory execution without a model

- is it possible to accurately reproduce a desired smooth joint-space reference trajectory with **reduced or no information** on the robot dynamic model?
- this is feasible (and possibly simple) in case of **repetitive motion tasks** over a finite interval of time
 - trials are performed iteratively, storing the **trajectory error** information of the current execution [k -th iteration] and processing it off line before the next trial [$(k + 1)$ -iteration] starts
 - the robot should be **reinitialized** in the same initial state at the beginning of each trial (typically, with $\dot{q} = 0$)
 - the control law is made of a **non-model based part** (often, a decentralized PD law) + a **time-varying feedforward** which is updated before every trial
- this scheme is called **iterative trajectory learning**



Scheme of iterative trajectory learning

- control design can be illustrated on a **SISO linear system** in the Laplace domain



$$W(s) = \frac{y(s)}{y_d(s)} = \frac{P(s)C(s)}{1 + P(s)C(s)}$$

closed-loop system **without** learning (feedf.)
($C(s)$ is, e.g., a PD control law)

$$u_k(s) = u'_k(s) + v_k(s) = C(s)e_k(s) + v_k(s) \quad \text{control law at iteration } k$$

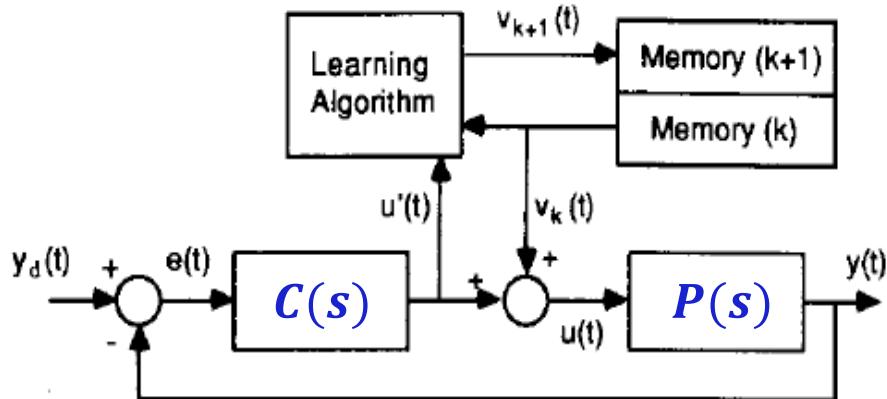
$$y_k(s) = W(s)y_d(s) + \frac{P(s)}{1 + P(s)C(s)} v_k(s) \quad \text{system output at iteration } k$$



Background math on feedback loops

whiteboard...

- algebraic manipulations on block diagram signals in the Laplace domain:
 $x(s) = \mathcal{L}[x(t)]$, $x = \{y_d, y, u', v, e\} \Rightarrow \{y_d, y_k, u'_k, v_k, e_k\}$, with transfer functions



$$\begin{aligned} y(s) &= P(s)u(s) = P(s)(v(s) + u'(s)) \\ &= P(s)v(s) + P(s)C(s)e(s) \\ &= P(s)v(s) + P(s)C(s)(y_d(s) - y(s)) \end{aligned}$$

$$\Rightarrow (1 + P(s)C(s)) y(s) = \\ = P(s)v(s) + P(s)C(s)y_d(s)$$

$$\Rightarrow y(s) = \frac{P(s)C(s)}{1 + P(s)C(s)} y_d(s) + \frac{P(s)}{1 + P(s)C(s)} v(s) = W(s)y_d(s) + W_v(s)v(s)$$

- feedback control law at iteration k

$$u'_k(s) = C(s)(y_d(s) - y_k(s)) = C(s)y_d(s) - P(s)C(s)(v_k(s) + u'_k(s))$$

$$\Rightarrow u'_k(s) = \frac{C(s)}{1 + P(s)C(s)} y_d(s) - \frac{P(s)C(s)}{1 + P(s)C(s)} v_k(s) = W_c(s)y_d(s) - W(s)v_k(s)$$

- error at iteration k

$$e_k(s) = y_d(s) - y_k(s) = y_d(s) - (W(s)y_d(s) + W_v(s)v_k(s)) = (1 - W(s))y_d(s) - W_v(s)v_k(s)$$

$$W_e(s) = 1/(1 + P(s)C(s))$$



Learning update law

- the **update** of the feedforward term is designed as

$$v_{k+1}(s) = \alpha(s)u'_k(s) + \beta(s)v_k(s)$$

with α and β **suitable filters**
(also non-causal, of the FIR type)

recursive expression
of feedforward term

$$v_{k+1}(s) = \frac{\alpha(s)C(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s))v_k(s)$$

recursive expression
of error $e = y_d - y$

$$e_{k+1}(s) = \frac{1 - \beta(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s))e_k(s)$$

- if a **contraction condition** can be enforced

$$|\beta(s) - \alpha(s)W(s)| < 1 \quad (\text{for all } s = j\omega \text{ frequencies such that ...})$$

then **convergence** is obtained for $k \rightarrow \infty$

$$v_\infty(s) = \frac{y_d(s)}{P(s)} \frac{\alpha(s)W(s)}{1 - \beta(s) + \alpha(s)W(s)} \quad e_\infty(s) = \frac{y_d(s)}{1 + P(s)C(s)} \frac{1 - \beta(s)}{1 - \beta(s) + \alpha(s)W(s)}$$



Proof of recursive updates

whiteboard...

- recursive expression for the **feedforward** v_k

$$\begin{aligned}
 v_{k+1}(s) &= \alpha(s)u'_k(s) + \beta(s)v_k(s) = \alpha(s)C(s)e_k(s) + \beta(s)v_k(s) \\
 &= \alpha(s)C(s)[W_e(s)y_d(s) - W_v(s)v_k(s)] + \beta(s)v_k(s) \\
 &= \frac{\alpha(s)C(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s)) v_k(s)
 \end{aligned}$$

- recursive expression for the **error** e_k

$$e_k(s) = y_d(s) - y_k(s) = y_d(s) - P(s)(v_k(s) + u'_k(s))$$

$$\Rightarrow v_k(s) = \frac{1}{P(s)}(y_d(s) - e_k(s)) - u'_k(s)$$

$$\begin{aligned}
 y_{k+1}(s) &= P(s)(v_{k+1}(s) + u'_{k+1}(s)) = P(s)(\alpha(s)u'_k(s) + \beta(s)v_k(s) + u'_{k+1}(s)) \\
 &= P(s)\left(\alpha(s)C(s)e_k(s) + \beta(s)\frac{1}{P(s)}(y_d(s) - e_k(s)) - \beta(s)C(s)e_k(s) + C(s)e_{k+1}(s)\right)
 \end{aligned}$$

$$\begin{aligned}
 e_{k+1}(s) &= y_d(s) - y_{k+1}(s) \\
 &= (1 - \beta(s))y_d(s) - [(\alpha(s) - \beta(s))P(s)C(s) - \beta(s)]e_k(s) - P(s)C(s)e_{k+1}(s) \\
 \Rightarrow e_{k+1}(s) &= \frac{1 - \beta(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s)) e_k(s)
 \end{aligned}$$



Proof of convergence

whiteboard...

from recursive expressions

$$v_{k+1}(s) = \frac{\alpha(s)C(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s)) v_k(s)$$

$$e_{k+1}(s) = \frac{1 - \beta(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s)) e_k(s)$$

compute **variations** from k to $k + 1$ (repetitive term in trajectory y_d vanishes!)

$$\Delta v_{k+1}(s) = v_{k+1}(s) - v_k(s) = (\beta(s) - \alpha(s)W(s)) \Delta v_k(s)$$

$$\Delta e_{k+1}(s) = e_{k+1}(s) - e_k(s) = (\beta(s) - \alpha(s)W(s)) \Delta e_k(s)$$

by **contraction mapping** condition $|\beta(s) - \alpha(s)W(s)| < 1 \Rightarrow \{v_k\} \rightarrow v_\infty, \{e_k\} \rightarrow e_\infty$

$$v_\infty(s) = \frac{\alpha(s)C(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s)) v_\infty(s)$$

$$e_\infty(s) = \frac{1 - \beta(s)}{1 + P(s)C(s)} y_d(s) + (\beta(s) - \alpha(s)W(s)) e_\infty(s)$$

$$\Rightarrow v_\infty(s) = \frac{y_d(s)}{P(s)} \frac{\alpha(s)W(s)}{1 - \beta(s) + \alpha(s)W(s)} \quad e_\infty(s) = \frac{y_d(s)}{1 + P(s)C(s)} \frac{1 - \beta(s)}{1 - \beta(s) + \alpha(s)W(s)}$$



$|\beta - \alpha W| < 1$ most still works

Comments on convergence

x

- if the choice $\beta = 1$ allows to satisfy the contraction condition, then convergence to zero tracking error is obtained

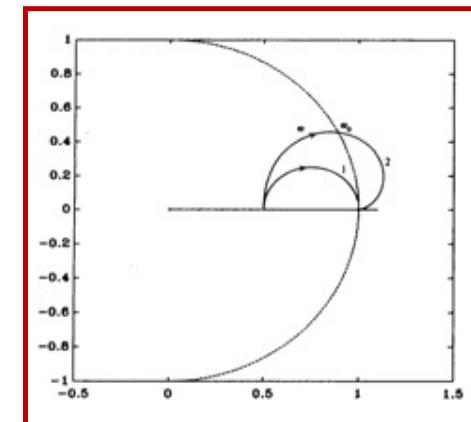
$$e_\infty(s) = 0$$

and the inverse dynamics command has been learned

$$v_\infty(s) = \frac{y_d(s)}{P(s)}$$

$$v_{k+1}(s) = \frac{\alpha(s)C(s)}{1 + P(s)C(s)} y_d(s) + \underbrace{(\beta(s) - \alpha(s)W(s))v_k(s)}_{\approx 0}$$

- in particular, for $\alpha(s) = 1/W(s)$ convergence would be in 1 iteration only!
- the use of filter $\beta(s) \neq 1$ allows to obtain convergence (but with residual tracking error) even in presence of unmodeled high-frequency dynamics
- the two filters can be designed from very poor information on system dynamics, using classic tools (e.g., Nyquist plots)





Application to robots

- for N -dof robots modeled as

$$[B_m + M(q)]\ddot{q} + [F_V + S(q, \dot{q})]\dot{q} + g(q) = u$$

handwritten notes: inertia of motor
 viscous friction

we choose as (initial = pre-learning) **control law**

$$u = u' = K_P(q_d - q) + K_D(\dot{q}_d - \dot{q}) + \hat{g}(q)$$

and design the **learning filters** (at each joint) using the **linear approximation** (not exact know.)

$$W_i(s) = \frac{q_i(s)}{q_{di}(s)} = \frac{K_{Di}s + K_{Pi}}{\hat{B}_{mi}s^2 + (\hat{F}_{Vi} + K_{Di})s + K_{Pi}} \quad i = 1, \dots, N$$

- initialization** of feedforward uses the best estimates

$$\nu_1 = [\hat{B}_m + \hat{M}(q_d)]\ddot{q}_d + [\hat{F}_V + \hat{S}(q_d, \dot{q}_d)]\dot{q}_d + \hat{g}(q_d)$$

or **simply** $\nu_1 = 0$ (in the worst case) at first trial $k = 1$

we will consider
of gravity
 $g(q) = \hat{g}(q)$
 forget about W and
for having a
linear system
 I can
use linear
methods



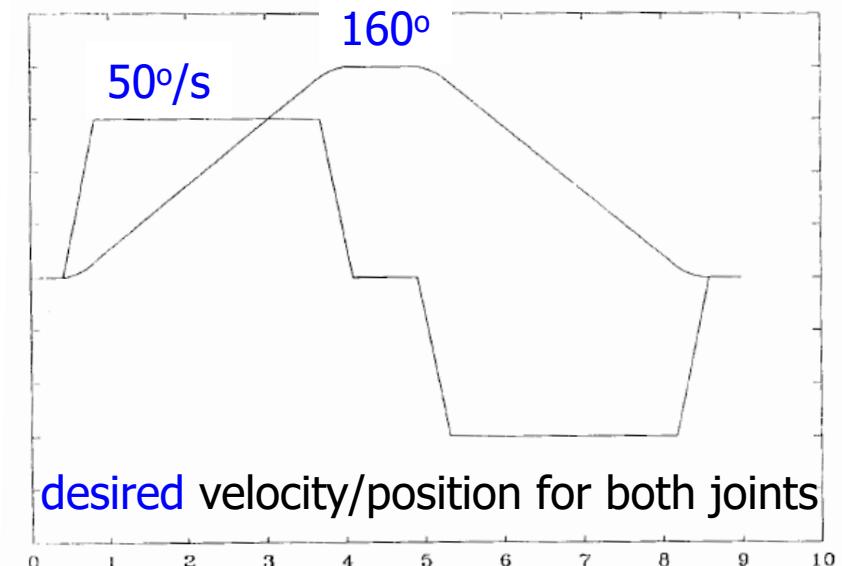
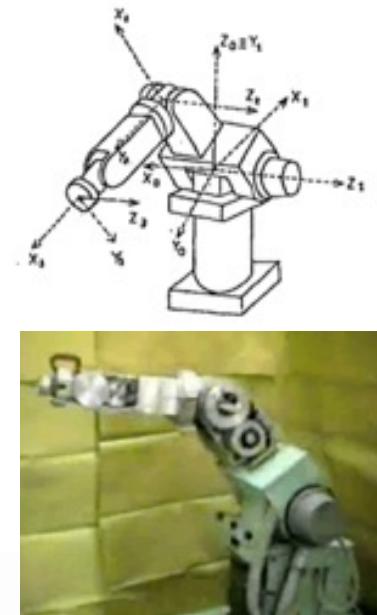
Experimental set-up

- joints 2 and 3 of 6R MIMO CRF robot prototype @DIS

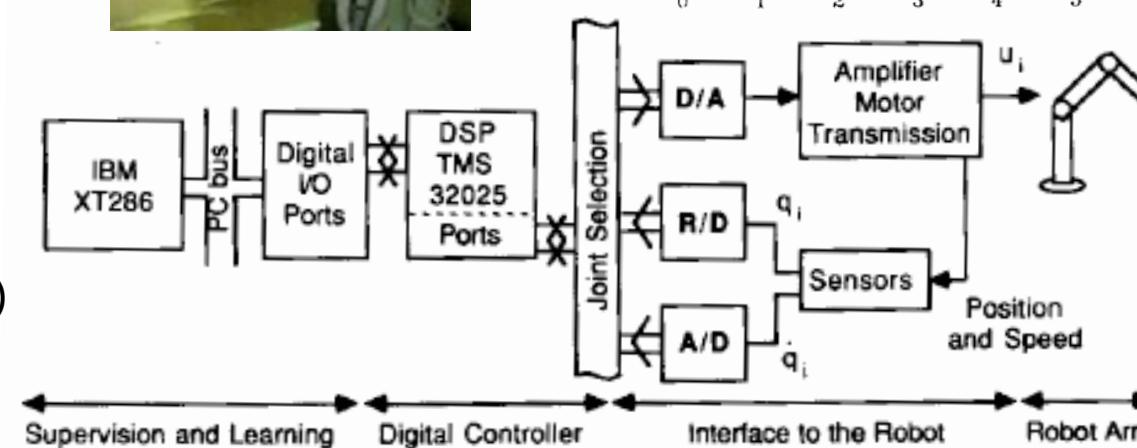
$\approx 90\%$ gravity balanced through springs

high level of dry friction

Harmonic Drives transmissions with ratio 160:1



DSP $T_c = 400\mu s$
D/A = 12 bit
R/D = 16 bit/ 2π
A/D = 11 bit/(rad/s)

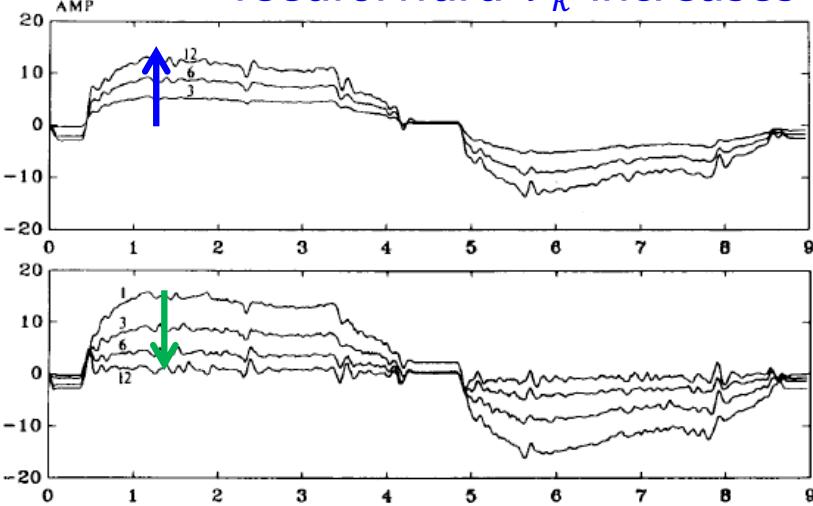
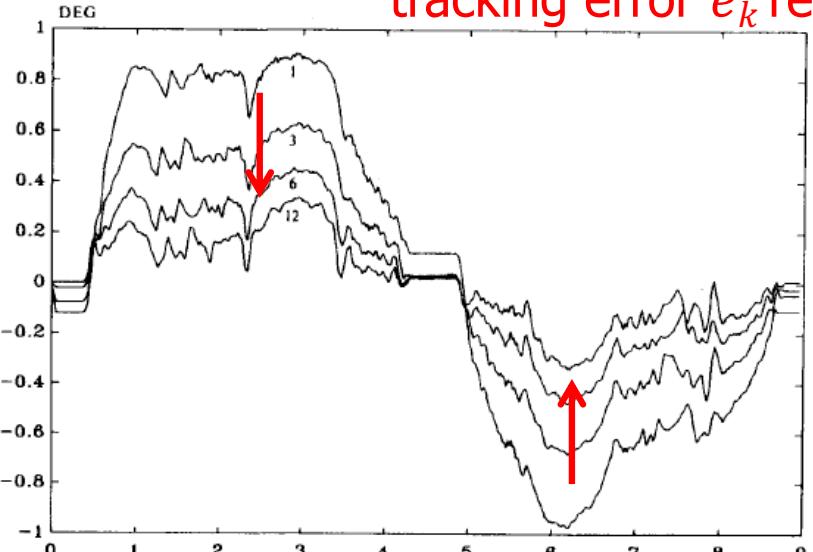


DC motors with current amplifiers
resolvers and tachometers

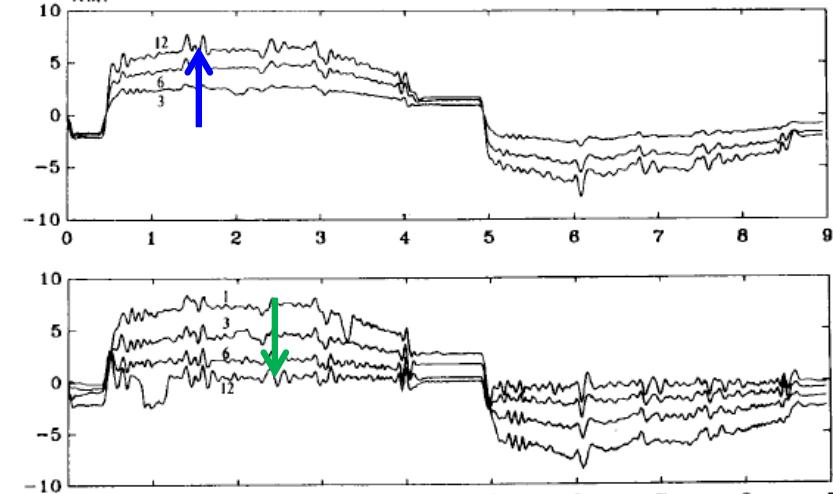
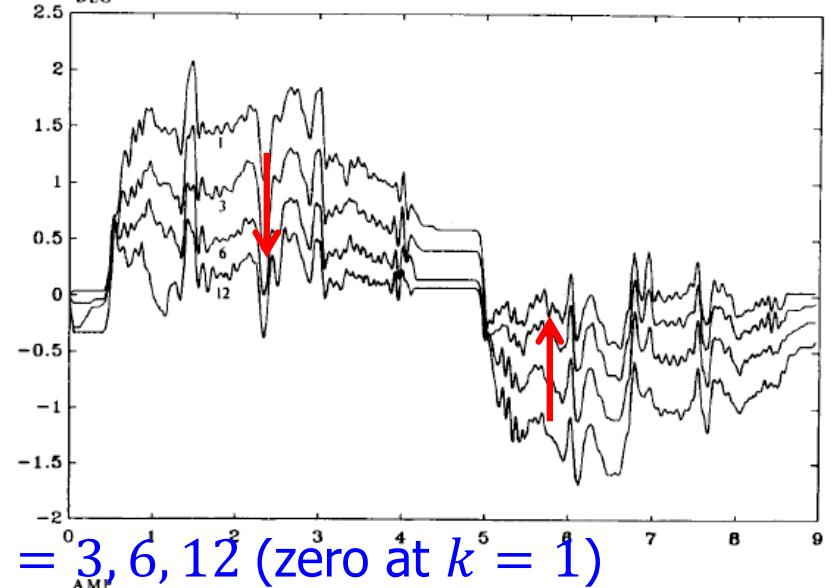


Experimental results

joint 2



joint 3





On-line learning control

- re-visitation of the learning idea so as to acquire the missing dynamic information in model-based trajectory control
- **on-line learning** approach
 - the robot improves tracking performance already while executing the task in feedback mode
- uses only position measurements from encoders
 - no need of joint torque sensors
- machine learning techniques used for
 - data collection and organization
 - regressor construction for estimating model perturbations
- **fast convergence**
 - starting with a reasonably good robot model
- extensions to underactuated robots or with flexible components

$n(q, \dot{q}) \rightarrow$ term for general active forces of the model \approx gravity, F_r, \dots



Control with approximate FBL

- dynamic model, its nominal part and (unstructured) uncertainty

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

ΔM known
 Δn known

- model-based (approximate) feedback linearization

$$\tau_{FBL} = \hat{M}(q)a + \hat{n}(q, \dot{q})$$

- resulting closed-loop dynamics with perturbation

$$\ddot{q} = a + \delta(q, \dot{q}, a) \quad \leftarrow \quad \delta = (M^{-1}\hat{M} - I)a + M^{-1}(\hat{n} - n)$$

- control law for tracking $q_d(t)$ is completed by using (at $t = t_k$) a linear design (PD with feedforward) and a learning regressor ε_k

$$a = a_k = u_k + \varepsilon_k$$

ε_k must be learned

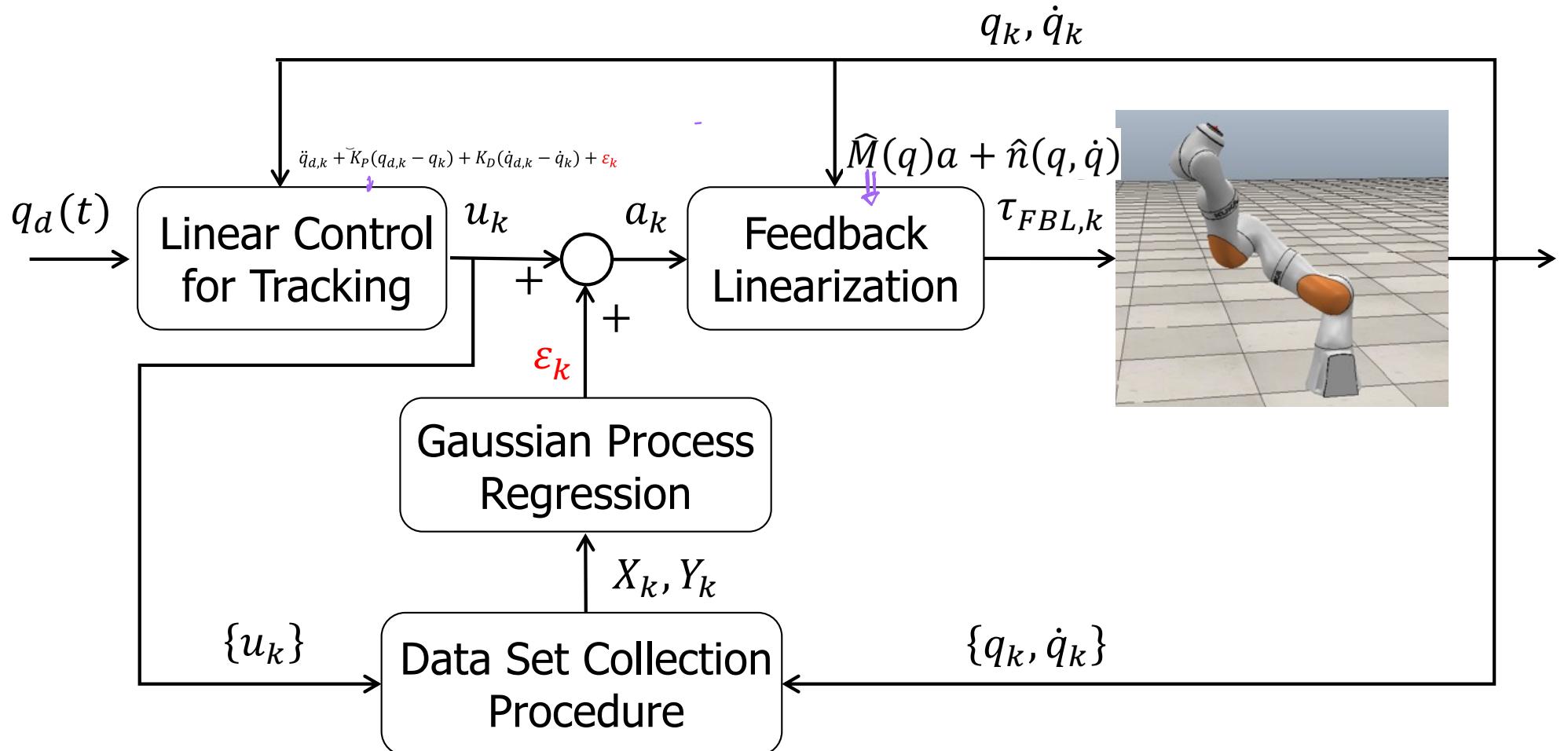
$$= \ddot{q}_{d,k} + K_P(q_{d,k} - q_k) + K_D(\dot{q}_{d,k} - \dot{q}_k) + \varepsilon_k$$

ε_k is it is not
of it but
of sampling time

Robotics 2 choose $\varepsilon_k = \delta \rightarrow \ddot{q} = a$ \leftarrow so we want to learn an $\varepsilon \approx \delta$ in order to obtain $a = \ddot{q}$ (?)



On-line learning scheme





On-line regressor

- Gaussian Process (GP) regression to estimate the perturbation δ

- from input-output observations that are noisy, with $\omega \sim \mathcal{N}(0, \Sigma_\omega)$, the generated data points at the k -th control step are

$$X_k = (q_k, \dot{q}_k, u_k) \quad Y_k = \ddot{q}_k - u_k$$

- assuming the ensemble of n_d observations with a joint Gaussian distribution

$$\begin{pmatrix} Y_{1:n_d-1} \\ Y_{n_d} \end{pmatrix} \sim \mathcal{N} \left(0, \begin{pmatrix} K & k \\ k^T & \kappa(X_{n_d}, X_{n_d}) \end{pmatrix} \right)$$

a Kernel
to be chosen

- the predictive distribution that approximates $\delta(\hat{X})$ for a generic query \hat{X} is

$$\varepsilon(\hat{X}) \sim \mathcal{N}(\mu(\hat{X}), \sigma^2(\hat{X}))$$

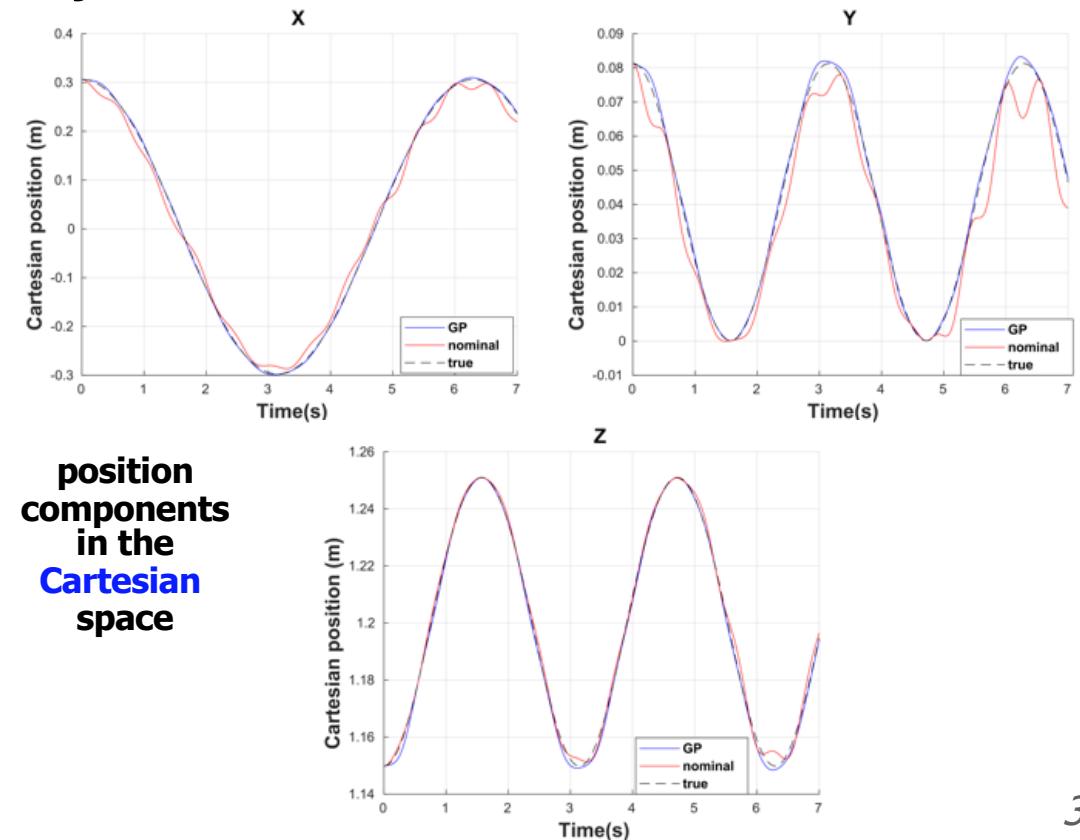
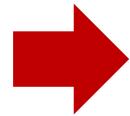
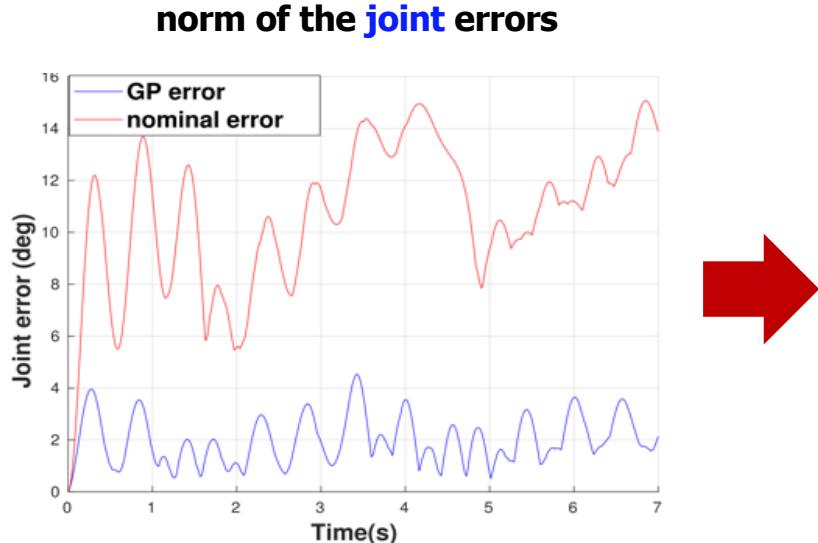
with

$$\left. \begin{aligned} \mu(\hat{X}) &= k^T(\hat{X})(K + \Sigma_\omega)^{-1}Y \\ \sigma^2(\hat{X}) &= \kappa(\hat{X}, \hat{X}) - k^T(\hat{X})(K + \Sigma_\omega)^{-1}k(\hat{X}) \end{aligned} \right\} \Rightarrow \varepsilon_k = \varepsilon(X_k)$$



Simulation results

- Kuka LWR iiwa, 7-dof robot
- model perturbations: dynamic parameters with $\pm 20\%$ variation, uncompensated joint friction
- 7 separate GPs (one for each joint), each with 21 inputs at every $t = t_k$
- sinusoidal trajectories for each joint



... at the first and only iteration!



Simulation results

video
(slowed
down)



An Online Learning Procedure for Feedback Linearization Control without Torque Measurements

M. Capotondi, G. Turrisi, C. Gaz, V. Modugno, G. Oriolo, A. De Luca

Robotics Lab, DIAG
Sapienza Università di Roma

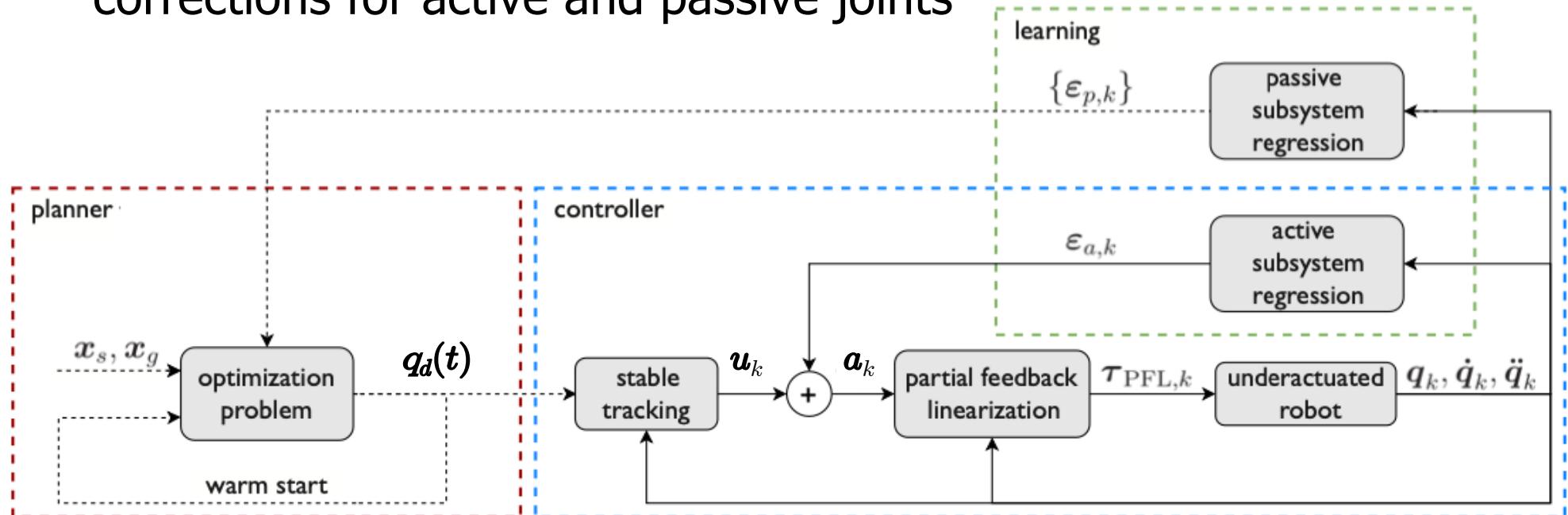
October 2019



Extension to underactuated robots

$$\begin{pmatrix} M_{aa}(q) & M_{ap}(q) \\ M_{ap}^T(q) & M_{pp}(q) \end{pmatrix} \begin{pmatrix} \ddot{q}_a \\ \ddot{q}_p \end{pmatrix} + \begin{pmatrix} n_a(q, \dot{q}) \\ n_p(q, \dot{q}) \end{pmatrix} = \begin{pmatrix} \tau \\ 0 \end{pmatrix}$$

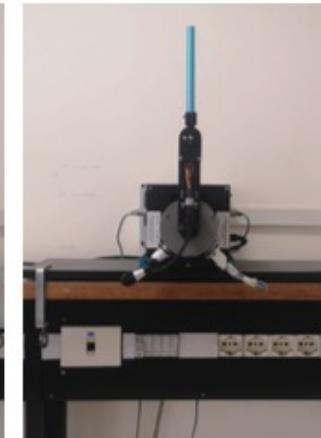
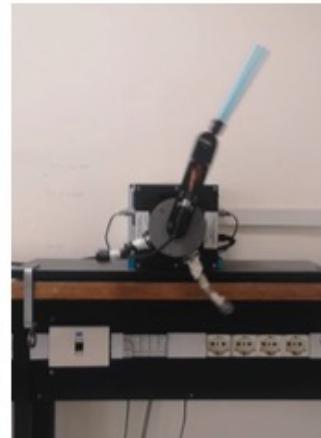
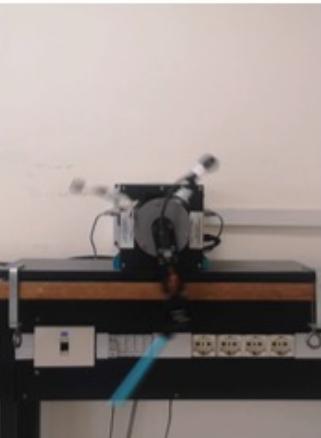
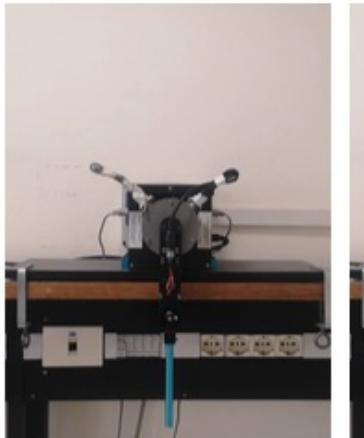
- planner optimizes motion of **passive** joints (at every iteration)
- controller for **active** joints with **partial** feedback linearization
- two **regressors** (on/off-line) for learning the required acceleration corrections for active and passive joints



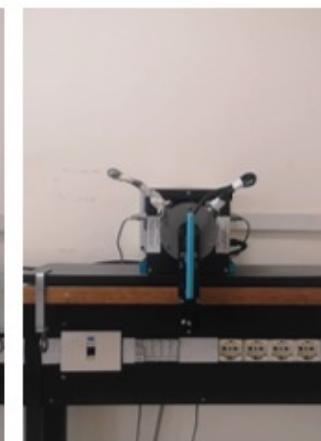


Experiments on the Pendubot

- Pendubot, 2-dof robot with passive second joint
- **swing-up** maneuvers from down-down to a new equilibrium state



⇒ up-up



⇒ down-up



Experimental results

IEEE Robotics and Automation Letters, vol. 7(1), 2022

video



Iterative Learning Control for Underactuated Robots

Giulio Turrisi, Marco Capotondi, Claudio Gaz,
Valerio Modugno, Giuseppe Oriolo, Alessandro De Luca

Robotics Lab, DIAG,
Sapienza Università di Roma

March 2021

convergence in 2 iterations!

latest video with more simulations & experiments
on YouTube https://youtu.be/1aKG_8gfvk



Robotics 2

Adaptive Trajectory Control

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Motivation and approach

- need of adaptation in robot motion control laws
 - large uncertainty on the robot dynamic parameters
 - poor knowledge of the inertial payload
- characteristics of **direct** adaptive control
 - direct aim is to bring to zero the state trajectory error, i.e., position and velocity errors
 - no need to estimate on-line the true values of the dynamic coefficients of the robot (as opposed to **indirect** adaptive control)
- main tool and methodology
 - **linear parametrization** of robot dynamics
 - **nonlinear** control law of the **dynamic** type (the controller has its own 'states')



Summary of robot parameters

- **parameters assumed to be known** (-DH parameters
- link lengths } known)

 - kinematic description based, e.g., on Denavit-Hartenberg parameters ($\{\alpha_i, d_i, a_i, i = 1, \dots, N\}$ in case of all revolute joints), including link lengths (kinematic calibration)

- **uncertain** parameters that can be **identified** off-line
 - masses m_i , positions r_{ci} of CoMs, and inertia matrices I_i of each link, appearing in combinations (dynamic coefficients) $\Rightarrow p \ll 10 \times N$
- parameters that are **(slowly) varying** during operation
 - viscous F_{Vi} , dry F_{Di} , and stiction F_{Si} friction at each joint $\Rightarrow 1 \div 3 \times N$
- **unknown** and abruptly changing parameters
 - mass, CoM, inertia matrix of the **payload** (w.r.t. the tool center point)



when a payload is firmly **attached** to the robot E-E, only the 10 parameters of the last link are modified, influencing however most part of the robot dynamics



Goal of adaptive control

- given a twice-differentiable desired joint trajectory $q_d(t)$
 - with known desired velocity $\dot{q}_d(t)$ and acceleration $\ddot{q}_d(t)$
 - possibly obtained by kinematic inversion + joint interpolation
- execute this trajectory under large dynamic uncertainties
 - with a trajectory tracking error vanishing **asymptotically**

$$\lim_{t \rightarrow \infty} e = \phi \quad e = q_d - q \rightarrow 0 \quad \dot{e} = \dot{q}_d - \dot{q} \rightarrow 0$$

- guaranteeing **global stability**, no matter how far are the initial estimates of the unknown/uncertain parameters from their true values and how large is the initial trajectory error
- identification is **not** of particular concern: in general, the estimates of dynamic coefficients will not converge to the true ones!
- if this convergence is a specific extra requirement, then one should use (more complex) **indirect adaptive schemes**



Linear parameterization

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

- there exists always a (p -dimensional) **vector a** of **dynamic coefficients**, so that the robot model takes the **linear** form

$$Y(q, \dot{q}, \ddot{q}) a = u$$

- vector **a** contains only unknown or uncertain coefficients
- each component of **a** is in general a **combination** of the robot physical parameters (not necessarily all of them)
- the model **regression matrix Y** depends linearly on \ddot{q} , quadratically on \dot{q} (for the terms related to kinetic energy), and nonlinearly (trigonometrically) on q



Trajectory controllers based on model estimates

- inverse dynamics feedforward (**FFW**) + PD (**linear**) control

$$u = \underbrace{\hat{M}(q_d)\ddot{q}_d + \hat{S}(q_d, \dot{q}_d)\dot{q}_d + \hat{g}(q_d) + \hat{F}_V\dot{q}_d}_{\hat{u}_d} + K_P e + K_D \dot{e}$$

- (**nonlinear**) control based on feedback linearization (**FBL**)

$$u = \hat{M}(q)(\ddot{q}_d + K_P e + K_D \dot{e}) + \hat{S}(q, \dot{q})\dot{q} + \hat{g}(q) + \hat{F}_V\dot{q}$$

$$\boxed{\hat{M}, \hat{S}, \hat{g}, \hat{F}_V \quad \Leftrightarrow \quad \text{estimate } \hat{a}}$$

- approximate estimates of dynamic coefficients may lead to **instability** with **FBL** due to temporary 'non-positive' PD gains (e.g., $\hat{M}(q)K_P < 0!$)
- not easy** to turn these laws in **adaptive** schemes: inertia inversion/use of acceleration (FBL); bounds on PD gains (FFW)



A control law more easily made 'adaptive'

- nonlinear trajectory tracking control (without cancellations) having global asymptotic stabilization properties

$$u = \hat{M}(q)\ddot{q}_d + \hat{S}(q, \dot{q})\dot{q}_d + \hat{g}(q) + \hat{F}_V\dot{q}_d + K_P e + K_D \dot{e}$$

- a natural **adaptive version** would require ...

$\dot{\hat{a}} =$ designing a suitable **update law**
(in continuous time)

- without extra assumptions, it can be shown that joint velocities become eventually "clamped" to those of the desired trajectory (zero **velocity** error), but a residual **position** error may be left
- idea: **on-line modification** with a **reference velocity**

$$\dot{q}_d \rightarrow \boxed{\dot{q}_r = \dot{q}_d + \Lambda(q_d - q)} \quad \Lambda > 0$$

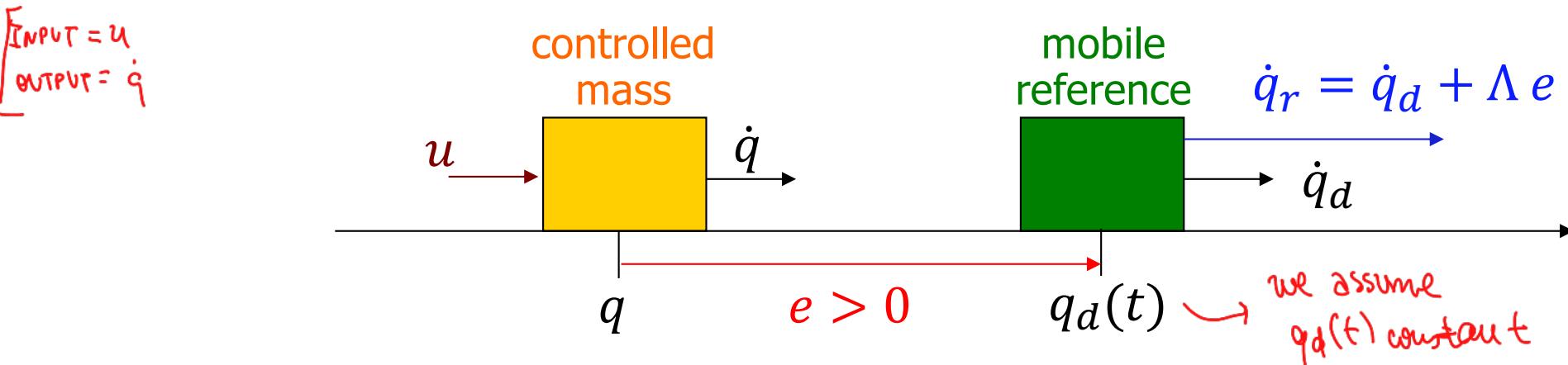
positive
definite

typically, $\Lambda = K_D^{-1}K_P$ (all matrices will be chosen **diagonal**)



Intuitive interpretation of \dot{q}_r

- elementary case
 - a mass 'lagging behind' a mobile reference ($e > 0$) at constant speed



→ 'enhanced' velocity error $s = \dot{q}_r - \dot{q} > \dot{q}_d - \dot{q} = \dot{e}$

$$u = K_D s = K_D(\dot{q}_r - \dot{q}) = K_D(\dot{q}_d + \Lambda e - \dot{q}) = K_D \dot{e} + \underbrace{K_D \Lambda e}_{K_P}$$

- a mass 'leading in front' of its mobile reference ($e < 0$)
- in a symmetric way, a 'reduced' velocity error will appear ($s < \dot{e}$)

$\tilde{e} = \gamma(q, \dot{q}, \ddot{q}_d, \dot{q}_d)$ we replace the desired velocities/accelerations with the reference velocities/accelerations.



Adaptive control law design

- substituting $\dot{q}_r = \dot{q}_d + \Lambda e$, $\ddot{q}_r = \ddot{q}_d + \Lambda \dot{e}$ in the previous nonlinear controller for trajectory tracking

$$u = \hat{M}(q)\ddot{q}_r + \hat{S}(q, \dot{q})\dot{q}_r + \hat{g}(q) + \hat{F}_V\dot{q}_r + K_P e + K_D \dot{e}$$

$$= Y(q, \dot{q}, \dot{q}_r, \ddot{q}_r)\hat{a} + K_P e + K_D \dot{e}$$

dynamic parameterization of
the control law using current estimates PD stabilization
(diagonal matrices, >0)
(note here the 4 arguments in $Y(\cdot)$!)

- update law for the estimates of the dynamic coefficients (\hat{a} becomes the p -dimensional state of the dynamic controller)

$$\dot{\hat{a}} = \Gamma Y^T(q, \dot{q}, \dot{q}_r, \ddot{q}_r)(\dot{q}_r - \dot{q})$$

$\underbrace{_s}_{s}$

$\Gamma > 0$ (diagonal)
estimation gains
(variation rate of estimates)

'modified' velocity error

estimation
gains



Asymptotic stability of trajectory error

Theorem

The introduced adaptive controller makes the tracking error along the desired trajectory **globally asymptotically stable**

$$e = q_d - q \rightarrow 0, \dot{e} = \dot{q}_d - \dot{q} \rightarrow 0$$

Proof

It doesn't matter which is the initial state, at the end we will have convergence

- a Lyapunov candidate for the closed-loop system (robot + dynamic controller) is given by

$$V = \frac{1}{2} s^T M(q) s + \frac{1}{2} e^T R e + \frac{1}{2} \tilde{a}^T \Gamma^{-1} \tilde{a} \geq 0$$

quadratic term in position error
quadratic term in acceleration error

$s = \dot{q}_r - \dot{q}$ ($= \dot{e} + \Lambda e$) $R > 0$ $\tilde{a} = a - \hat{a}$
modified velocity error *constant matrix* *error in parametric estimation*
 (to be specified later)

$$V = 0 \iff \hat{a} = a, q = q_d, s = 0 \quad (\Rightarrow \dot{q} = \dot{q}_d)$$

① We have to prove that $V \geq 0$



Proof (cont)

- the time derivative of V is

$$\dot{V} = \frac{1}{2} s^T \dot{M}(q) s + s^T M(q) \dot{s} + e^T R \dot{e} - \tilde{a}^T \Gamma^{-1} \dot{\tilde{a}}$$

since $\dot{\tilde{a}} = -\dot{\hat{a}}$ ($\dot{a} = 0$) \rightarrow we assume that the dynamic parameters are constant

- the closed-loop dynamics is given by

$$\begin{aligned} M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) + F_V \dot{q} &= \\ &= \widehat{M}(q)\ddot{q}_r + \widehat{S}(q, \dot{q})\dot{q}_r + \widehat{g}(q) + \widehat{F}_V \dot{q}_r + K_P e + K_D \dot{e} \end{aligned}$$

subtracting the two sides from $M(q)\ddot{q}_r + S(q, \dot{q})\dot{q}_r + g(q) + F_V \dot{q}_r$ leads to

$$\begin{aligned} M(q)\dot{s} + (S(q, \dot{q}) + F_V)s &= \\ &= \widetilde{M}(q)\ddot{q}_r + \widetilde{S}(q, \dot{q})\dot{q}_r + \widetilde{g}(q) + \widetilde{F}_V \dot{q}_r - K_P e - K_D \dot{e} \end{aligned}$$

with $\widetilde{M} = M - \widehat{M}$, $\widetilde{S} = S - \widehat{S}$, $\widetilde{g} = g - \widehat{g}$, $\widetilde{F}_V = F_V - \widehat{F}_V$



Proof (cont)

- from the property of **linearity in the dynamic coefficients**, it follows

$$\textcircled{M(q)\dot{s}} + (S(q, \dot{q}) + F_V)s = Y(q, \dot{q}, \dot{q}_r, \ddot{q}_r)\tilde{a} - K_P e - K_D \dot{e}$$

- substituting in \dot{V} , together with $\dot{\tilde{a}} = \Gamma Y^T s$, and using the skew-symmetry of matrix $\dot{M} - 2S$ we obtain

$$\begin{aligned}\dot{V} &= \frac{1}{2} s^T [\dot{M}(q) - 2S(q, \dot{q})] s - s^T F_V s + s^T Y \tilde{a} \\ &\quad - s^T (K_P e + K_D \dot{e}) + e^T R \dot{e} - \tilde{a}^T Y^T s \\ &= -s^T F_V s - s^T (K_P e + K_D \dot{e}) + e^T R \dot{e}\end{aligned}$$

- replacing $s = \dot{e} + \Lambda e$ and being $F_V = F_V^T$ (diagonal)

$$\dot{V} = -e^T (\Lambda^T F_V \Lambda + \Lambda^T K_P) e$$

a complete quadratic form in e, \dot{e} !

$$\xrightarrow{\hspace{1cm}} -e^T (2\Lambda^T F_V + \Lambda^T K_D + K_P - \textcircled{R}) \dot{e} - \dot{e}^T (F_V + K_D) \dot{e}$$



Proof (end)

- defining now (all matrices are **diagonal!**)

$$\Lambda = K_D^{-1} K_P > 0 \quad R = 2K_P (I + K_D^{-1} F_V) > 0$$

cancels the cross-term in $e^T (\dots) \dot{e}$ and leads to

$$\begin{aligned}\dot{V} &= -e^T \Lambda^T (F_V + K_D) \Lambda e - \dot{e}^T (F_V + K_D) \dot{e} \\ &= -e^T K_P K_D^{-1} (F_V + K_D) K_D^{-1} K_P e - \dot{e}^T (F_V + K_D) \dot{e} \leq 0\end{aligned}$$

and thus

$$\boxed{\dot{V} = 0 \iff e = \dot{e} = 0}$$

the thesis follows from Barbalat lemma + LaSalle theorem



the maximal invariant set of states $\subseteq \{\dot{V} = 0\}$ has **zero trajectory error** ($e = \dot{e} = 0$) and **a constant value** for \hat{a} , not necessarily the true one ($\tilde{a} \neq 0$)

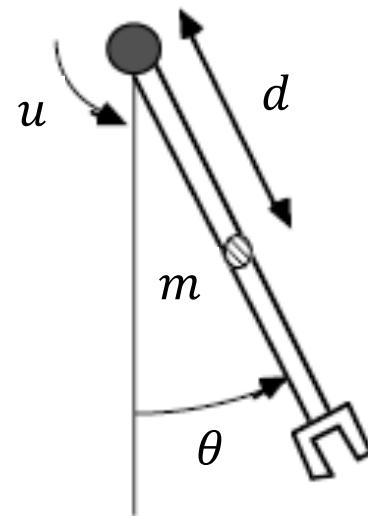


Remarks

- if the desired trajectory $q_d(t)$ is **persistently exciting**, then also the estimates of the dynamic coefficients converge to their true values
- **condition** of persistent excitation
 - for **linear** systems: # of frequency components in the desired trajectory should be at least **twice as large** as # of unknown coefficients
 - for **nonlinear** systems: the condition can be checked only **a posteriori** (a squared motion integral should always be positive bounded from below)
- in case of known absence of (viscous) friction ($F_V \equiv 0$), the same proof applies (a bit easier in the final part)
- the adaptive controller **does not require** the inverse of the inertia matrix (true or estimated), nor the actual robot acceleration (only the desired acceleration), nor further lower bounds on $K_P > 0, K_D > 0$
- adaptation can also be used **only for a subset** of dynamic coefficients, with the others being known ($Y_a = Y_{adapt} \hat{a}_{adapt} + Y_{known} a_{known}$)
- the **non-adaptive version** (using accurate estimates) is a static tracking controller based on the **passivity** property of robot dynamics



Case study: Single-link under gravity



model $I\ddot{\theta} + mg_0d \sin \theta + f_V \dot{\theta} = u$ (with friction)

linear parameterization

$$Y(\theta, \dot{\theta}, \ddot{\theta})a = [\ddot{\theta} \quad \sin \theta \quad \dot{\theta}] \begin{bmatrix} I \\ mg_0d \\ f_V \end{bmatrix} = u$$

adaptive controller

$$e = \theta_d - \theta \quad \Lambda > 0$$

$$\dot{\theta}_r = \dot{\theta}_d + \frac{k_P}{k_D} e$$

$$\gamma_i > 0, i = 1, 2, 3$$

$$u = \widehat{I} \ddot{\theta}_r + \widehat{mg_0d} \sin \theta + \widehat{f_V} \dot{\theta}_r + k_P e + k_D \dot{e}$$

$$\dot{\hat{a}} = \begin{pmatrix} \dot{\widehat{I}} \\ \dot{\widehat{mg_0d}} \\ \dot{\widehat{f_V}} \end{pmatrix} = \begin{pmatrix} \gamma_1 \ddot{\theta}_r \\ \gamma_2 \sin \theta \\ \gamma_3 \dot{\theta}_r \end{pmatrix} (\dot{\theta}_r - \dot{\theta})$$



Simulation data

- **real** dynamic coefficients

$$I = 7.5, \quad mg_0d = 6, \quad f_V = 1$$

- **initial** estimates

$$\widehat{I} = 5, \quad \widehat{mg_0d} = 5, \quad \widehat{f}_V = 2$$

- control parameters

$$k_P = 25, \quad k_D = 10, \quad \gamma_i = 5, \quad i = 1,2,3$$

- **test trajectories** (starting with $\theta(0) = 0, \dot{\theta}(0) = 0$)

- **first**

$$\theta_d(t) = -\sin t$$

Note: same test trajectories
used also for robust control

- **second**

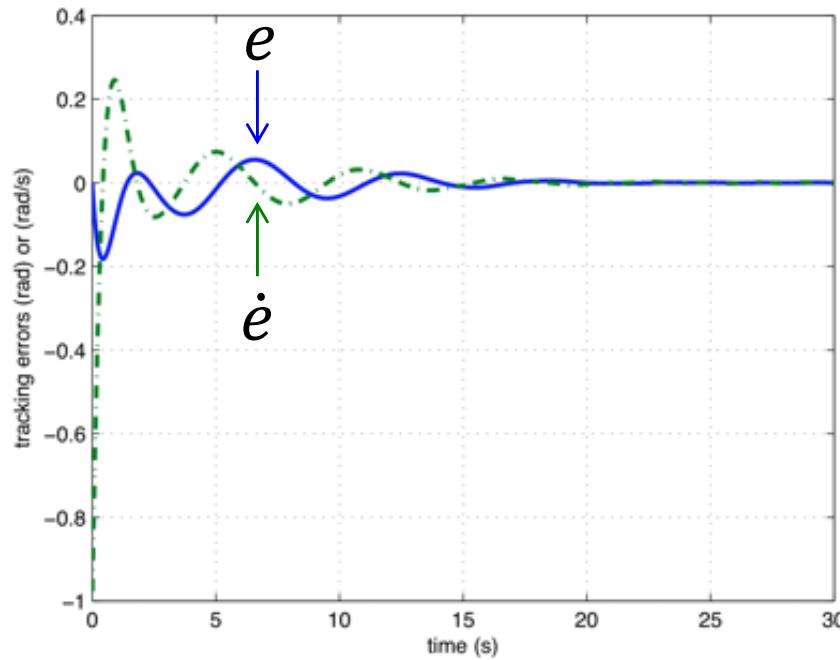
$\ddot{\theta}_d(t)$ = (periodic) bang-bang acceleration profile with
 $A = 1 \text{ rad/s}^2, \omega = 1 \text{ rad/s}$



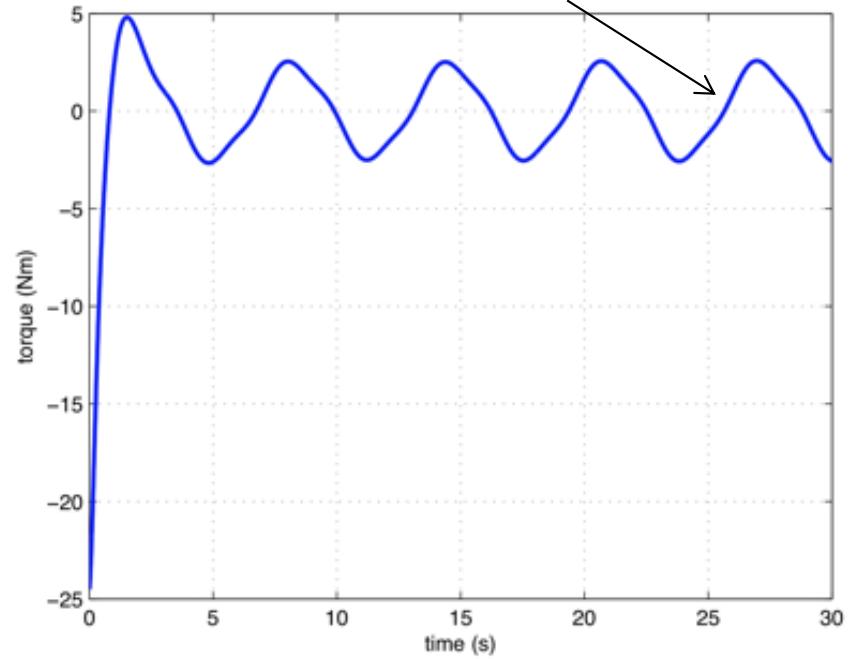
Results

first trajectory

note the nonlinear system dynamics
(no sinusoidal regime at steady state!)



position and velocity errors

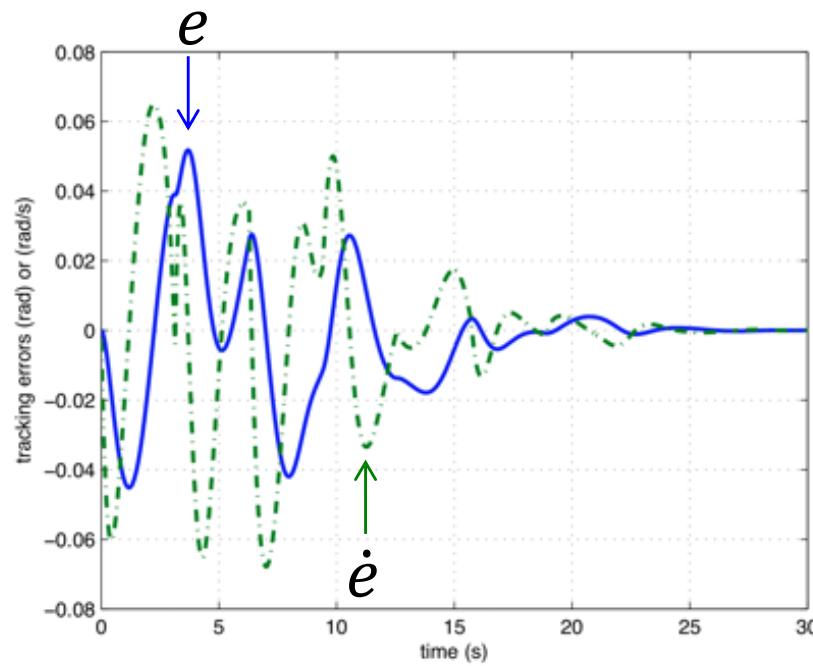


control torque

$$\theta_d(t) = -\sin t$$

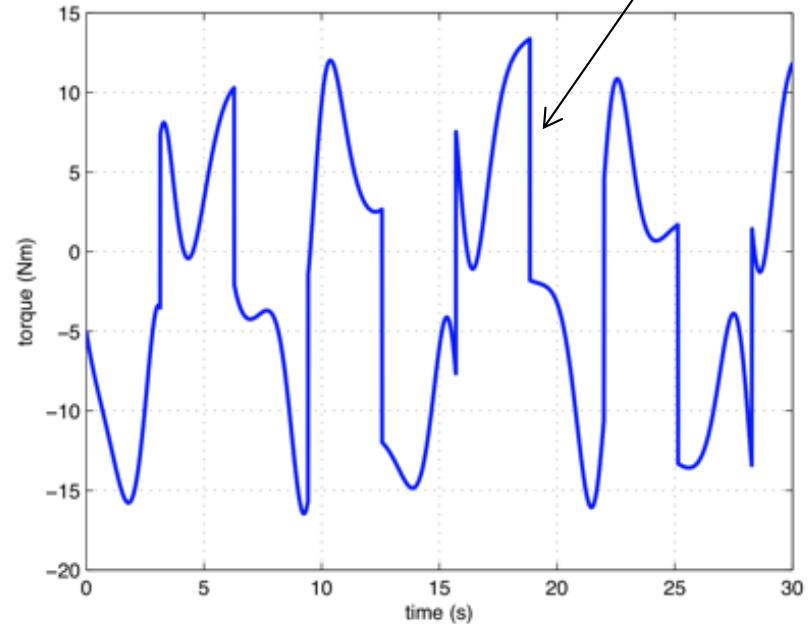


Results second trajectory



position and velocity errors

note the torque discontinuities
(due to those of the desired acceleration)

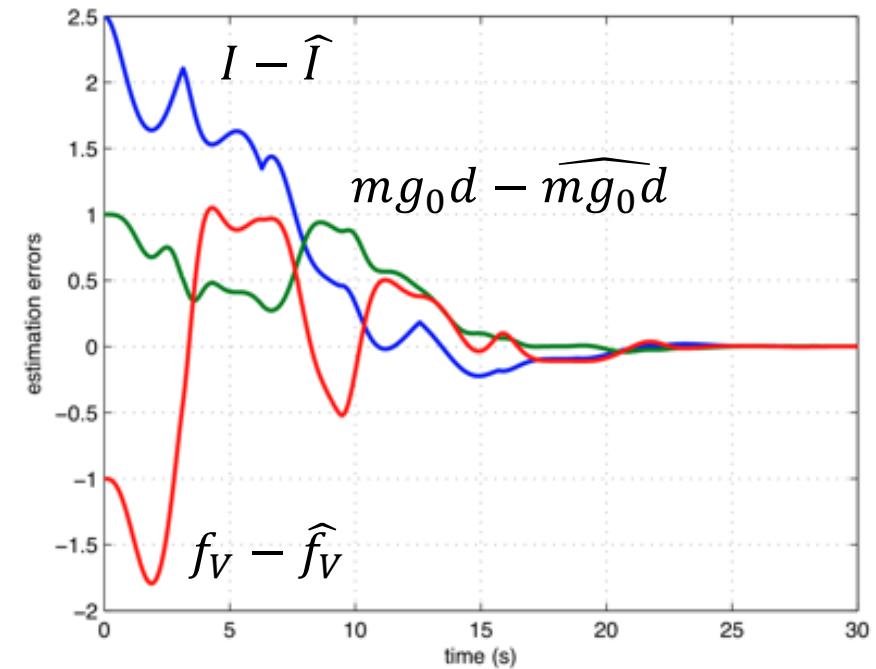
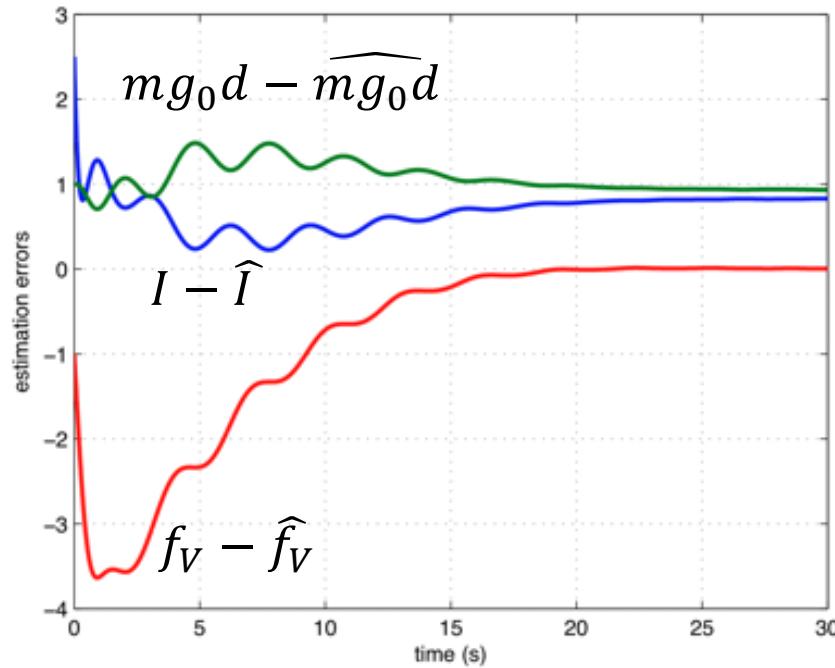


control torque

$$\ddot{\theta}_d(t) = \text{(periodic) bang-bang acceleration profile}$$



Estimates of dynamic coefficients



$$\text{errors } \tilde{a} = a - \hat{a}$$

first trajectory

only the estimate of the viscous
friction coefficient converges
to the true value

second trajectory

all three estimates of
dynamic coefficients converge
to their true values



A special case: Adaptive regulation

- adaptation in case q_d is **constant**
- **no special simplifications** for the presented adaptive control law (designed for the general tracking case...)

$$u = \hat{M}(q)\ddot{q}_r + \hat{S}(q, \dot{q})\dot{q}_r + \hat{g}(q) + \hat{F}_v\dot{q}_r + K_P e + K_D \dot{e}$$

$$\dot{\hat{a}} = \Gamma Y^T(q, \dot{q}, \dot{q}_r, \ddot{q}_r)(\dot{q}_r - \dot{q})$$

since $\dot{q}_r = \Lambda(q_d - q)$ and $\ddot{q}_r = -\Lambda\dot{q}$ **do not** vanish!

- a **different** case would be the availability of an adaptive version of the trajectory tracking controller

$$u = \hat{M}(q)\ddot{q}_d + \hat{S}(q, \dot{q})\dot{q}_d + \hat{g}(q) + \hat{F}_v\dot{q}_d + K_P e + K_D \dot{e}$$

since, when q_d collapses to a constant, **only the adaptation of the gravity term** would be left over (which is what one would naturally expect...)



An efficient adaptive regulator

- use a linear parameterization of the **gravity term** only

$$g(q) = G(q)a_g$$

with a p_g -dimensional vector a_g

- an adaptive regulator yielding **global asymptotic stability** of the equilibrium state $(q_d, 0)$ is provided by

$$u = G(q)\hat{a}_g + K_P(q_d - q) - K_D\dot{q}$$

$$\dot{\hat{a}}_g = \gamma G^T(q) \left(\frac{2e}{1 + 2\|e\|^2} - \beta\dot{q} \right), \quad \gamma > 0$$

where $e = q_d - q$, $K_P > 0$, $K_D > 0$ (symmetric), and $\beta > 0$ is chosen sufficiently **large**

(see paper by P. Tomei, IEEE TRA, 1991; available as extra material on the course web)



An adaptive regulator

Sketch of asymptotic stability analysis

- use the function

$$V = \frac{\beta}{2} (\dot{q}^T M(q) \dot{q} + e^T K_P e) - \frac{2\dot{q}^T M(q)e}{1 + 2\|e\|^2} + \frac{1}{2} (\hat{a}_g - a_g)^T (\hat{a}_g - a_g)$$

- a sufficient condition for V to be a **Lyapunov candidate** is that

$$\beta > \frac{2M_M}{\sqrt{M_m K_{P,m}}}$$

- a sufficient condition which guarantees **also** that

$$\dot{V} = \dots \leq -a\|e\|^2 - b\|\dot{q}\|^2 \leq 0, \quad a > 0, b > 0$$

is

$$\beta > \max \left\{ \frac{2M_M}{\sqrt{M_m K_{P,m}}}, \frac{1}{K_{D,m}} \left(\frac{K_{D,m}^2}{2K_{P,m}} + 4M_M + \frac{\alpha_S}{\sqrt{2}} \right) \right\}$$

$\|S(q, \dot{q})\| \leq \alpha_S \dot{q}$

Note: for any **symmetric, positive definite** matrix A

$$A_M = \lambda_{\max}(A) = \sqrt{\lambda_{\max}(A^T A)} = \|A\| \quad \text{and thus, e.g., } \frac{1}{2} \dot{q}^T M(q) \dot{q} \geq \frac{1}{2} M_m \|\dot{q}\|^2$$

$$A_m = \lambda_{\min}(A)$$

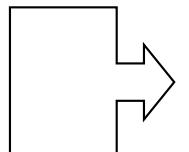


Robotics 2

Control in the Cartesian Space

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Regulation of robot Cartesian pose

- “PD +” type control for regulation problems
 - proportional to the Cartesian pose error, with a derivative term (on velocity) + cancellation/compensation of gravity in joint space
- robot
 - dynamics $M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = \dot{u}$ *neglect dissipative effects* dimension of spaces
 - kinematics $p = f(q) \rightarrow \dot{p} = J(q)\dot{q}$ joint = n Cartesian = m
pose e-e *Analytic Jacobian*
- goal: asymptotic stabilization of the end-effector pose

$$p = p_d, \dot{q} = \dot{q}_d = 0 \rightarrow \dot{p}_d = 0$$

Note: if $m = n$, then $\dot{q} = 0 \Leftrightarrow \dot{p} = 0$ up to singularities

if $m < n$, then the goal is not uniquely associated

to a complete robot state: $n - m$ joint coordinates are missing ...
simplifying of possibility to reach robot pose



A Cartesian regulation law

(*)

$$u = J^T(q)K_P(p_d - p) - K_D \dot{q} + g(q)$$

DESIRED
ACTUAL POSE
ERROR

damping action

$K_P, K_D > 0$
positive
def., usually
diagonal
 (symmetric)

Theorem

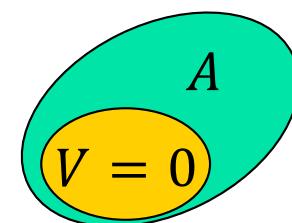
under the control law (*), the robot state will converge asymptotically to the set
$$\begin{aligned} A &= \{\dot{q} = 0, q: K_P(p_d - f(q)) \in N(J^T(q))\} \\ &\supseteq \{\dot{q} = 0, q: f(q) = p_d\} \end{aligned}$$

Proof

define $e_p = p_d - p$ (Cartesian error) and the associated Lyapunov-like candidate function

$$V = \frac{1}{2} \dot{q}^T M(q) \dot{q} + \frac{1}{2} e_p^T K_P e_p$$

with $V = 0 \Leftrightarrow (q, \dot{q}) \in \{\dot{q} = 0, q: f(q) = p_d\} \subseteq A$





Proof (cont)

differentiating $V = \frac{1}{2} \dot{q}^T M(q) \dot{q} + \frac{1}{2} e_p^T K_P e_P \geq 0$

$$\begin{aligned}\dot{V} &= \dot{q}^T (M\ddot{q} + \frac{1}{2} \dot{M}\dot{q}) - e_p^T K_P \dot{p} \\ &= \dot{q}^T (u - S\dot{q} - g + \frac{1}{2} \dot{M}\dot{q}) - e_p^T K_P \dot{p} \\ &= \dot{q}^T (J^T K_P e_P - K_D \dot{q} + g - g) - e_p^T K_P J \dot{q} \\ &= -\dot{q}^T K_D \dot{q} \leq 0\end{aligned}$$

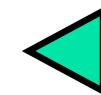
with $\dot{V} = 0 \Leftrightarrow \dot{q} = 0$

in this situation, the **closed-loop equations** become

$$M(q)\ddot{q} + g(q) = J^T(q)K_P e_P + g(q) \rightarrow \ddot{q} = M^{-1}(q)J^T(q)K_P e_P$$

$\rightarrow \ddot{q} = 0 \Leftrightarrow K_P e_P \in N(J^T(q))$

by applying LaSalle theorem, the thesis follows





Corollary

+1PICALLY WE STOP AT AEST

for a given initial state $(\underline{q(0)}, \dot{\underline{q}}(0))$, if the robot **does not encounter any singularity** of $J^T(q)$ (configurations where $\rho(J^T) < m \leq n$) during its motion, then there is **asymptotic stabilization** to one single state (when $m = n$) or to a set of states (when $m < n$) such that

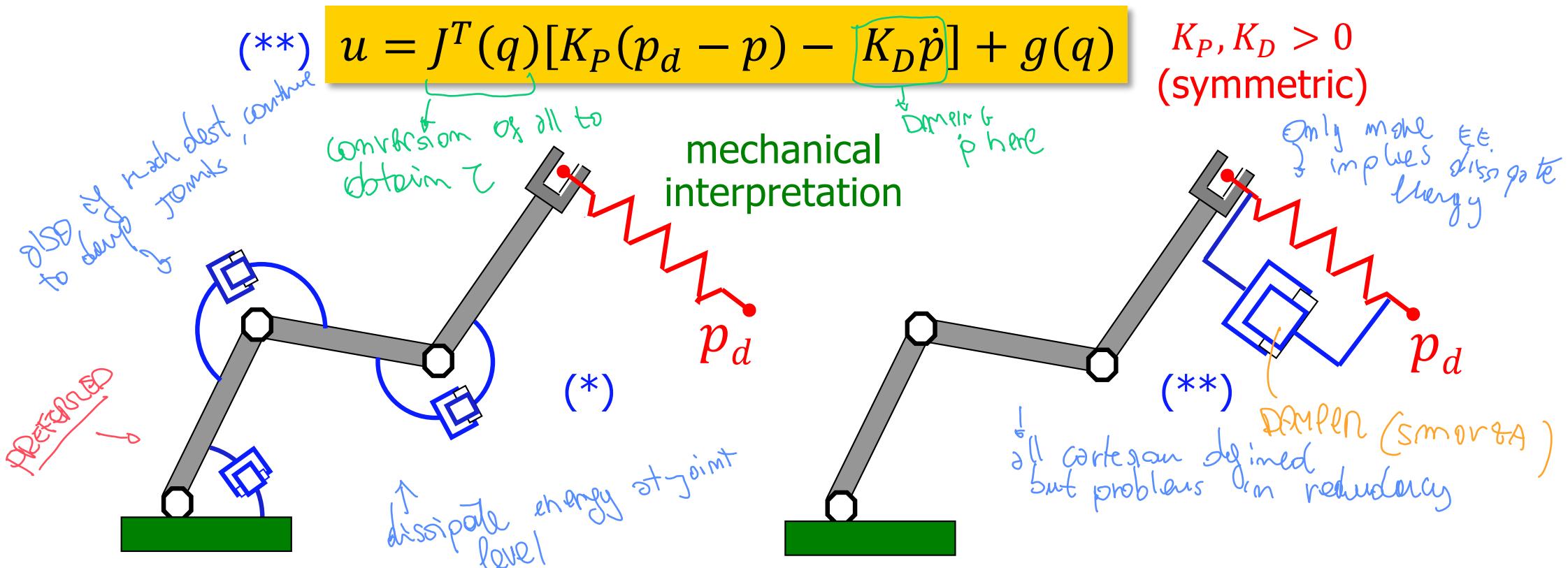
$$e_P = 0, \dot{q} = 0$$

Note: singular configurations q of $J^T(q)$ coincide with those of $J(q)$



A possible variant for regulation

“all Cartesian” PD control + gravity cancellation in joint space



J^T transforms the “virtual” **elastic**, for (*), or **visco-elastic**, for (**), force/torque acting on the end-effector into control torques at the joints



Feedback linearization in Cartesian space

robot

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

output

$$y = p, \quad p = f(q)$$

Cartesian
position/orientation

assume: $m = n$

algorithm

differentiate the output(s) as many times as needed up to the appearance of (at least one of) the input torque(s), then verify if it is possible to solve for the input = "inversion"

uniform
"relative degree"
 $\rho = 2$
for all outputs

$$\begin{aligned} y &= f(q) \\ \dot{y} &= J(q)\dot{q} \quad \text{from the dynamic model} \\ \ddot{y} &= J(q)\ddot{q} + \dot{J}(q)\dot{q} \\ &= J(q)M^{-1}(q)[u - c(q, \dot{q}) - g(q)] + \dot{J}(q)\dot{q} \end{aligned}$$

Theorem

for a non-redundant robot, it is possible to exactly linearize and decouple the dynamic behavior at the **Cartesian** level if and only if

$$\det J(q) \neq 0$$

↑
J non singular



Feedback linearization in Cartesian space (in the right coordinates!)

explained
at
↓

control law

$$u = M(q)J^{-1}(q)a + c(q, \dot{q}) + g(q) - M(q)J^{-1}(q)\dot{J}(q)\dot{q}$$
$$= \beta(q)a + \alpha(q, \dot{q})$$

→ $\ddot{\tilde{y}} = \ddot{p} = J(q)M^{-1}(q)[u - c(q, \dot{q}) - g(q)] + \dot{J}(q)\dot{q} = a$

p, \dot{p} are the so-called "**linearizing**" coordinates

closed-loop equations (in the **joint space**)

$$M^{-1} * M\ddot{q} + c + g = MJ^{-1}[a - \dot{J}\dot{q}] + c + g$$

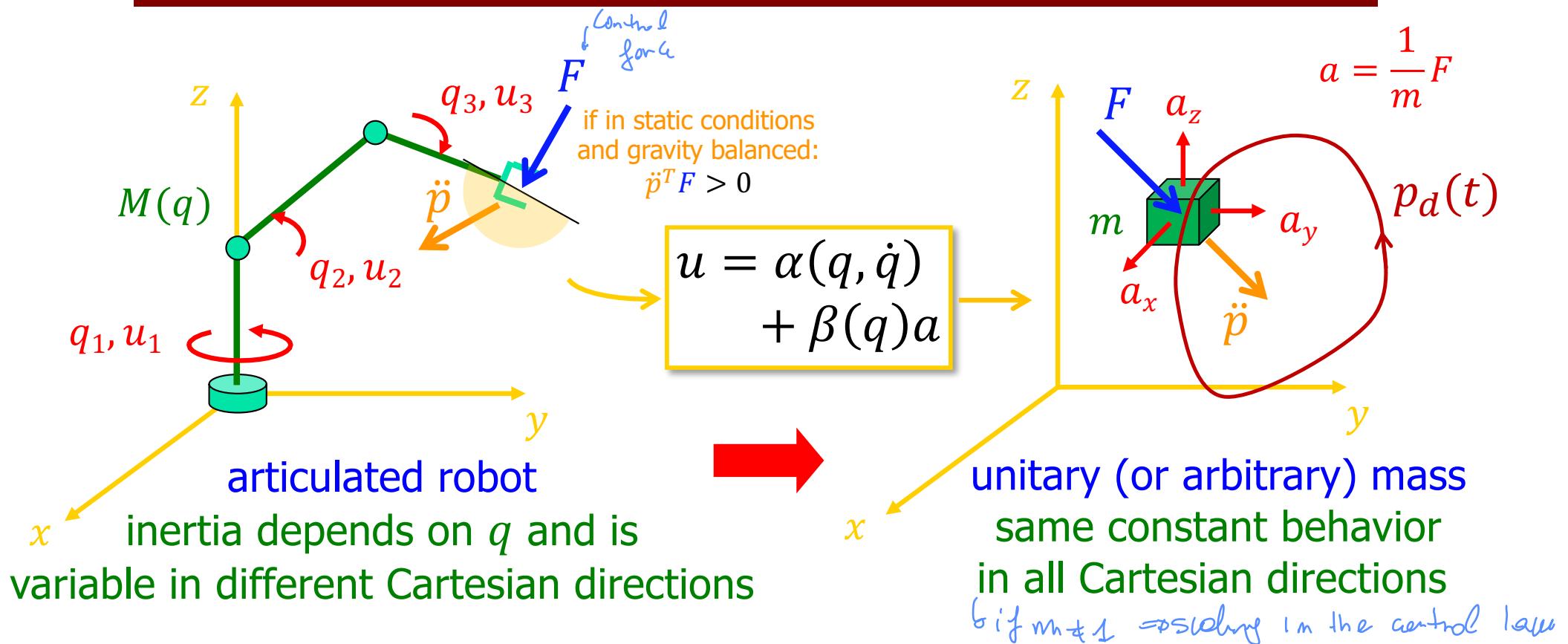
→ $\ddot{q} = J^{-1}(q)a - J^{-1}(q)\dot{J}(q)\dot{q}$

purely
kinematic
equations

(but still **nonlinear** and **coupled!!**)



Physical interpretation



when a force F is applied at the end-effector

- the uncontrolled robot will accelerate with \ddot{p} in a different direction
- on the other hand
- a mass m accelerates always in the **same** direction of the applied force F



Alternative derivation in purely Cartesian terms

the previous exact linearizing and decoupling law can be rewritten in **Cartesian terms** using a **control force/torque F**

$$u = M(q)J^{-1}(q)a + c(q, \dot{q}) - M(q)J^{-1}(q)\dot{J}(q)\dot{q} + g(q)$$

joint torque u is moved to the **Cartesian space** as $F = J^{-T}(q)u$ (for $m = n$)

$$\begin{aligned} F &= [J^{-T} MJ^{-1}]a \xrightarrow{\text{Cartesian inertia}} [JM^{-1}J^T]^{-1} = M_p(p) \\ &\quad + [J^{-T} c - J^{-T} MJ^{-1}\dot{J}\dot{q}] \xrightarrow{\text{Cartesian Coriolis/centrifugal terms}} \\ &\quad + [J^{-T} g] \xrightarrow{\text{Cartesian gravity}} \\ &= M_p a + c_p + g_p \end{aligned}$$

 this is the feedback linearization law applied to the **Cartesian dynamic model** of the robot

$$M_p(p)\ddot{p} + c_p(p, \dot{p}) + g_p(p) = F$$

$$\ddot{p} = a$$



Remarks - 1

- the design of a **Cartesian trajectory tracking control** is completed by **stabilizing** the tracking error in the **m independent** chains of double integrators, i.e., by setting

$$a_i = \ddot{p}_{di} + K_{Di}(\dot{p}_{di} - \dot{p}_i) + K_{Pi}(p_{di} - p_i) \quad \begin{matrix} \text{scalars} \\ K_{Pi} > 0, K_{Di} > 0 \\ i = 1, \dots, m \end{matrix}$$

- the transient behavior of the Cartesian error along a desired trajectory is **exponentially stable** (with arbitrary eigenvalues assigned by choosing the diagonal gains of K_P, K_D)
- for $p_d = \text{constant}$ (regulation task), the control law becomes

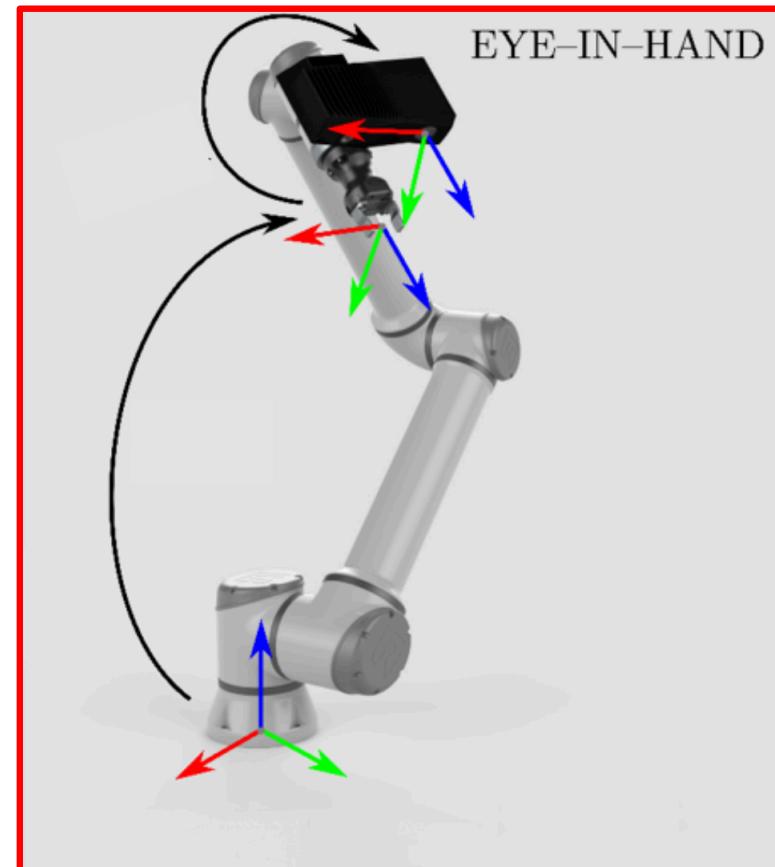
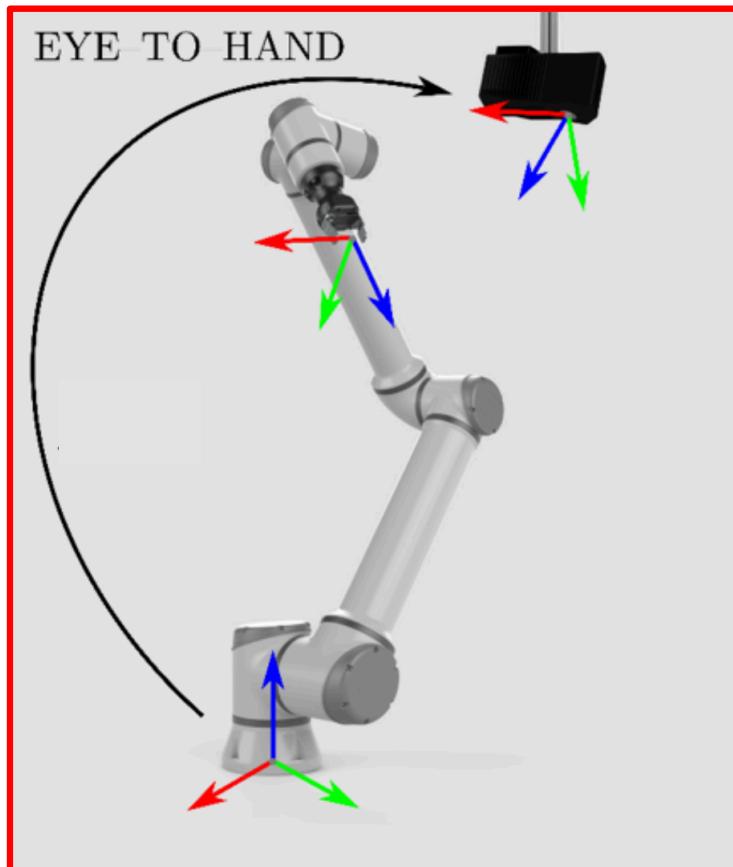
$$u = M(q)J^{-1}(q)[K_P e_P - K_D J(q)\dot{q}] + c(q, \dot{q}) + g(q) - M(q)J^{-1}(q)J(q)\dot{q}$$

which is computationally heavier than a control law designed directly for regulation, such as previous laws **(*)** or **(**)**, but it keeps the property of an exponentially stable transient error



Remarks - 2

- the Cartesian pose/velocity can either be directly **measured** by external sensors (cameras: eye-to-hand/eye-in-hand) or **computed** through direct and differential kinematics of the robot





Remarks - 3

- in **redundant** robots ($m < n$), by replacing $MJ^{-1} = (JM^{-1})^{-1}$ in the control law with some (weighted) pseudoinverse $(JM^{-1})_W^\#$, one still obtains **input-output** decoupling and linearization, but not exact linearization of the whole **state** dynamics
 - there is an additional internal dynamics left of dimension $n - m$



More on the redundant case ...

- suppose $m < n$, but with a Jacobian J of full rank m
- let the control law (with null-space torque term u_0) be defined as

$$u = (J(q)M^{-1}(q))_W^\# \left(a - j(q)\dot{q} + J(q)M^{-1}(q)(c(q, \dot{q}) + g(q)) \right) \\ + \left(I - (J(q)M^{-1}(q))_W^\# J(q)M^{-1}(q) \right) u_0$$

where $(JM^{-1})_W^\# = W^{-1}M^{-1}J^T(JM^{-1}W^{-1}M^{-1}J^T)^{-1}$

- three standard choices for $W > 0$

$$W = I \implies (JM^{-1})^\# = M^{-1}J^T(JM^{-2}J^T)^{-1}$$

$$W = M^{-1} \implies (JM^{-1})_{M^{-1}}^\# = J^T(JM^{-1}J^T)^{-1}$$

$$W = M^{-2} \implies (JM^{-1})_{M^{-2}}^\# = M J^T(JJ^T)^{-1} = MJ^\#$$

each associated control torque optimizes a different criterion (see the slides on redundant robots)

- all give the same $\ddot{p} = a$, with u_0 available for null-space control



Conclusions

- most of the control laws presented in the joint space (i.e., driven by a joint error) can be **translated** with relative ease to the Cartesian space, e.g.
 - regulation with constant gravity compensation
 - adaptive regulation
 - robust control for trajectory tracking
 - adaptive control for trajectory tracking
- the **main issues** are related to
 - kinematic singularities, both for the Jacobian transpose and the Jacobian inverse control laws: suitable modifications are needed to obtain **singularity robustness**
 - kinematic redundancy ($m < n$): use of a **stabilizing null-space torque** control is needed for the extra $n - m$ generalized coordinates (locally, $n - m$ joint variables)



Robotics 2

Robot Interaction with the Environment

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI



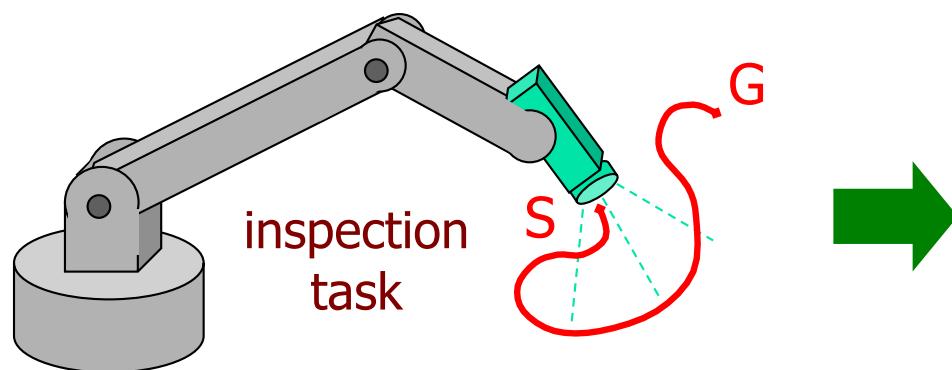


Robot-environment interaction

a robot (end-effector) may interact with the **environment**

- **modifying the state** of the environment (e.g., pick-and-place operations)
- **exchanging forces** (e.g., assembly or surface finishing tasks)

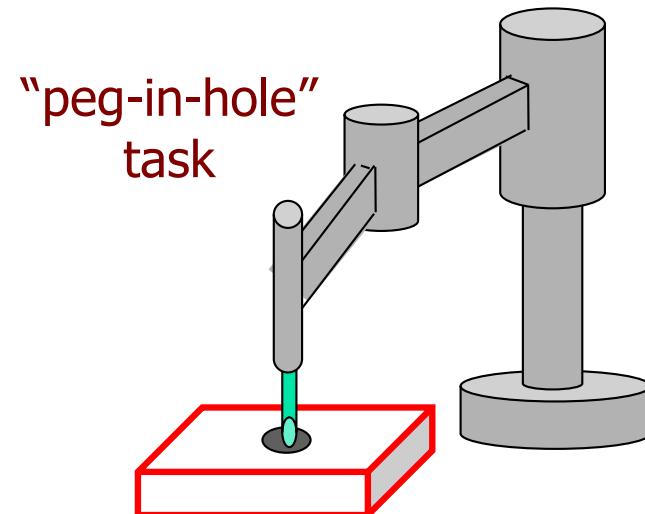
control of free motion



sensors: position (encoders)
at the joints* or
vision at the Cartesian level

*and velocity (by numerical differentiation
or, more rarely, with tachos)

control of compliant motion



sensors: as before +
6D force/torque
(at the robot wrist)



Robot compliance

PASSIVE

robot end-effector equipped with **mechatronic devices** that “comply” with the **generalized forces** applied at the TCP = Tool Center Point

RCC = Remote Center of Compliance device



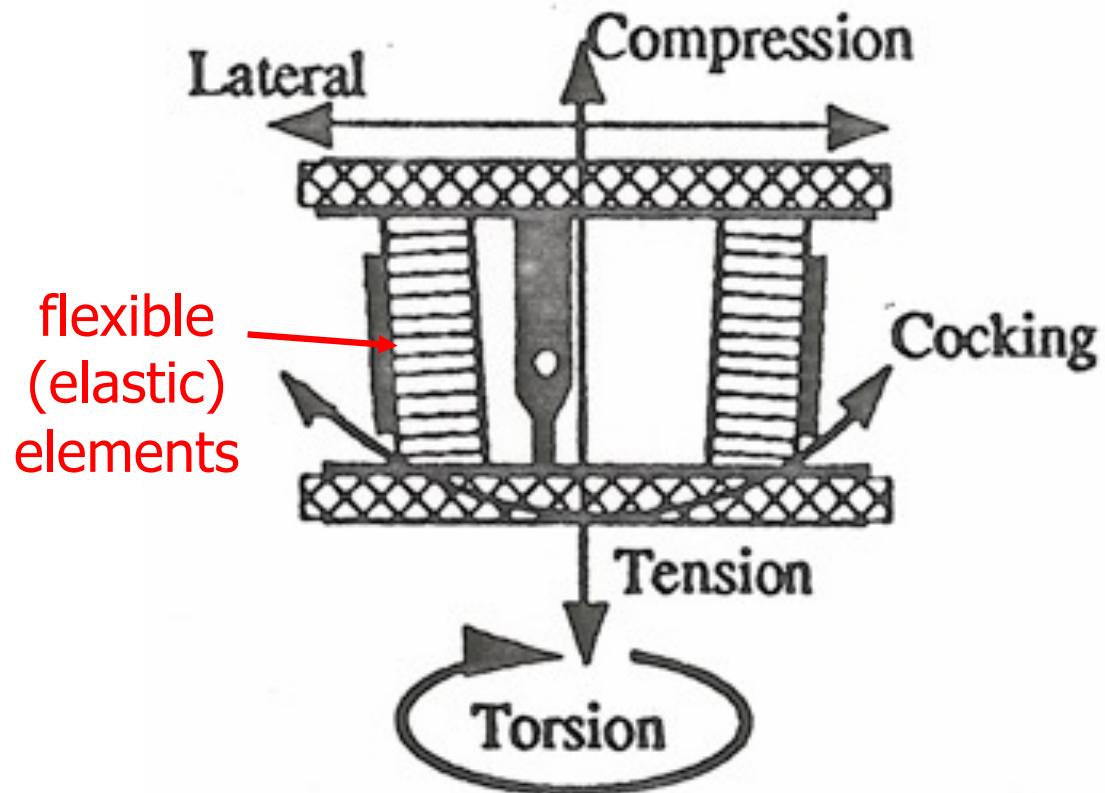
ACTIVE

robot is moved by a **control law** so as to react in a desired way to **generalized forces** applied at the TCP (typically measured by a F/T sensor)

- **admittance** control
contact forces \Rightarrow velocity commands
- **stiffness/compliance** control
contact displacements \Rightarrow force commands
- **impedance** control
contact displacements \Leftrightarrow contact forces



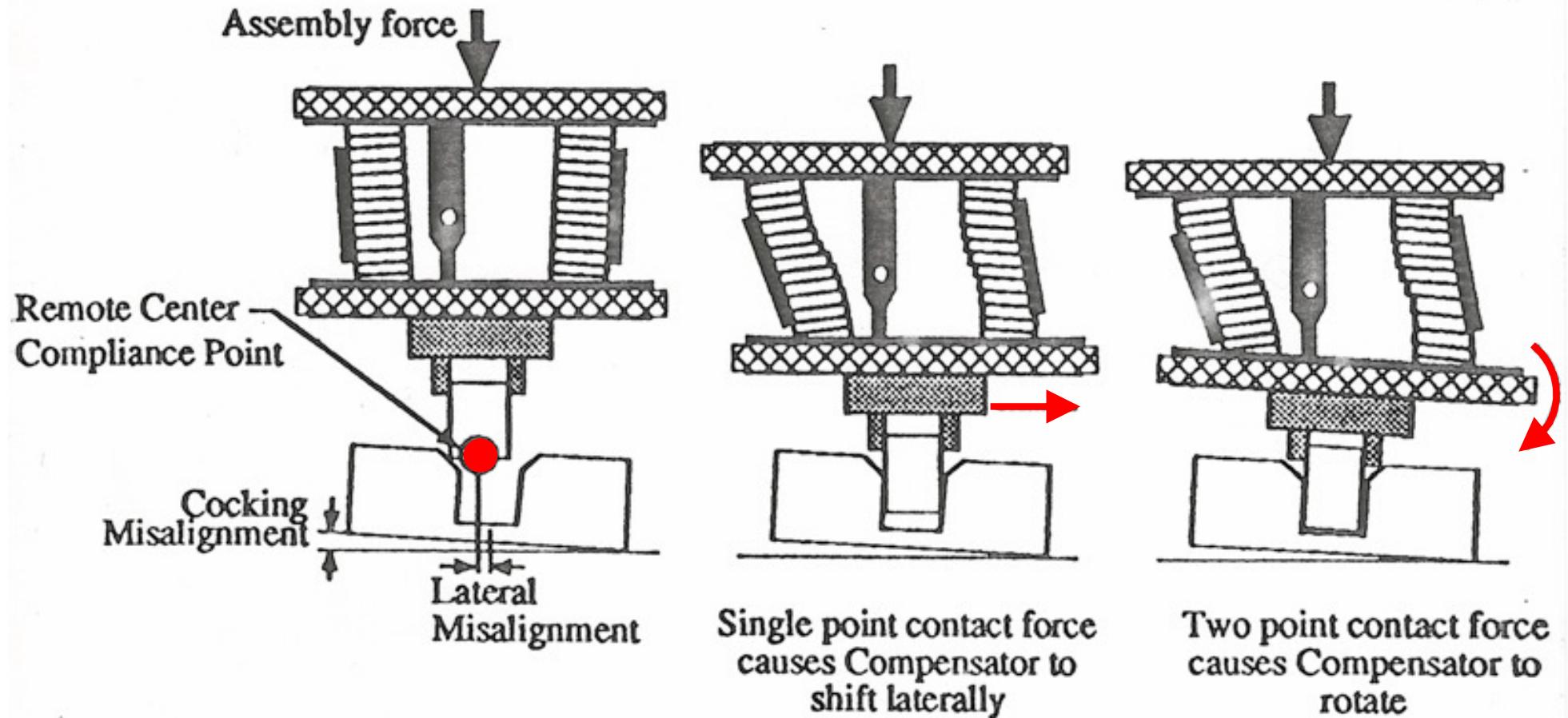
RCC device



RCC models of
different size
by ATI

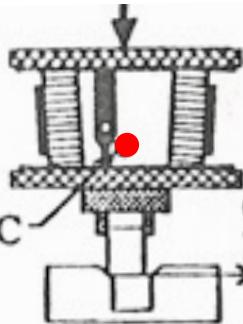


RCC behavior in case of misalignment errors in assembly tasks



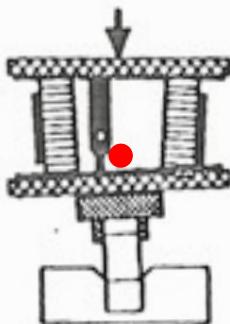


Effects of RCC positioning



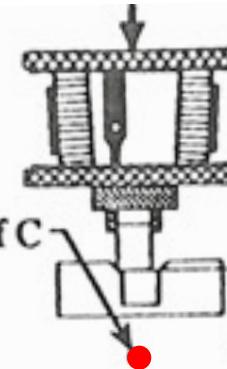
Contact
Force

C of C



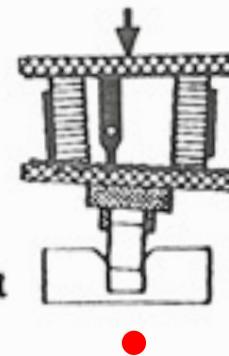
With the C of C far above the point of contact a lateral contact force causes the part to enter at an angle, causing a two point contact.

too high...



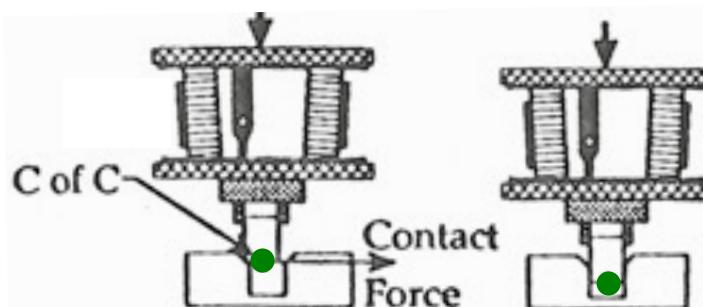
Contact
Force

C of C



With the C of C far below the point of contact the part enters at an angle causing two point contact

too low...



Contact
Force

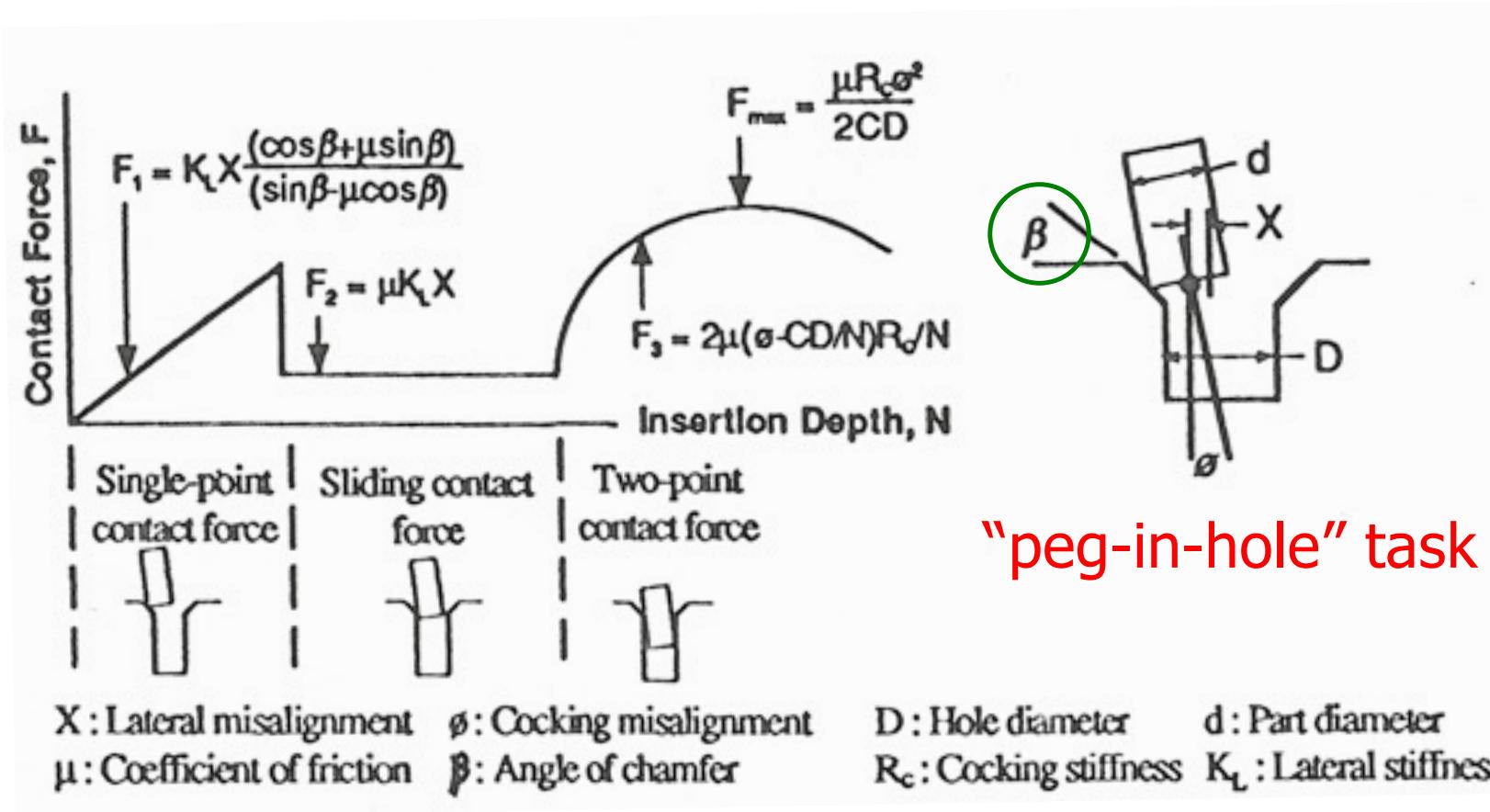
C of C

When the C of C is near the contact point the part enters correctly

correct!
(TCP = RCC)



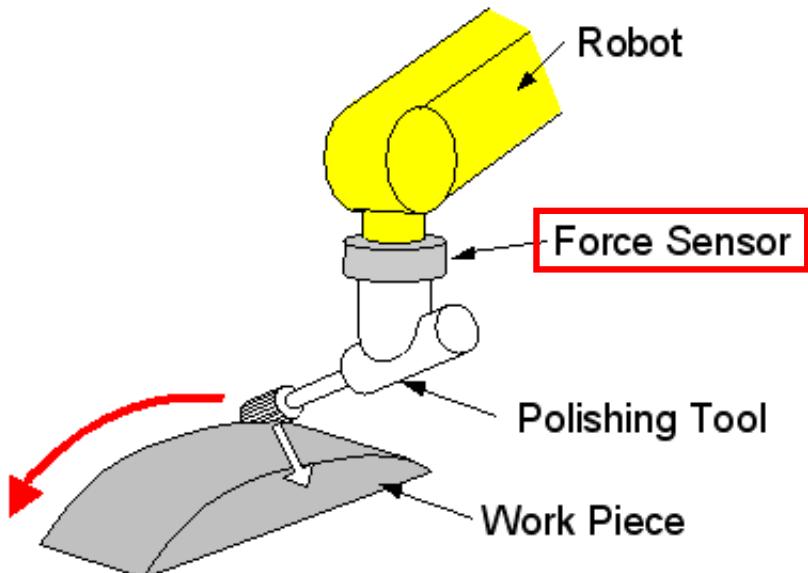
Typical evolution of assembly forces



chamfer angle β = to ease the insertion,
related also to the tolerances of the hole



Active compliance for contour following



Following with constant pushing force



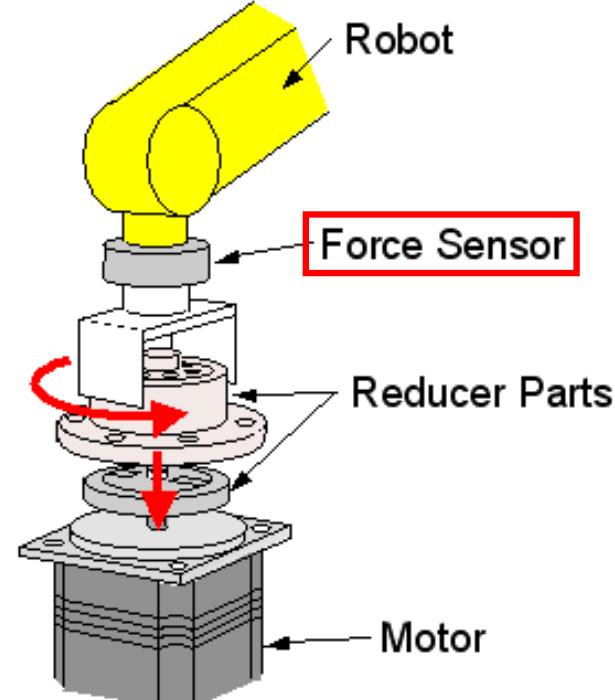
Washstand



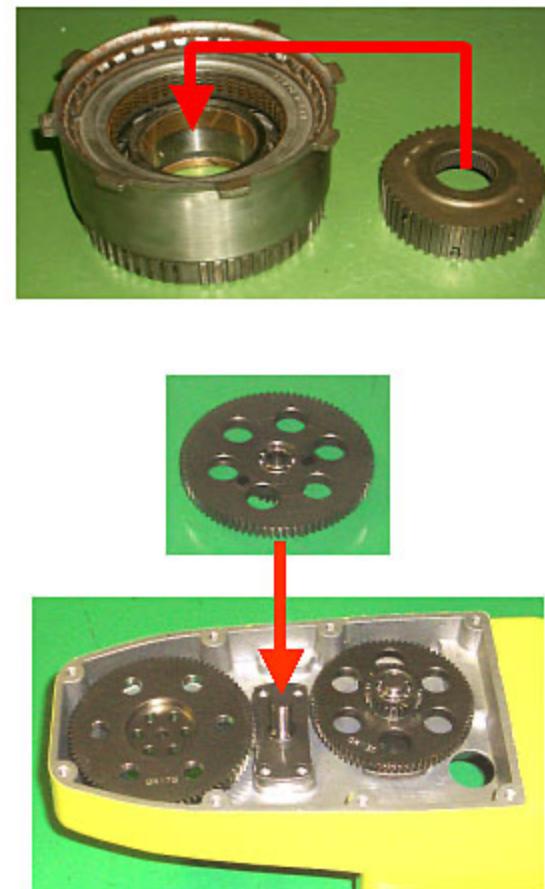
Metal Cabinet



Active compliance “matching” of mechanical parts



Phase matching by force sensing



Gear Parts

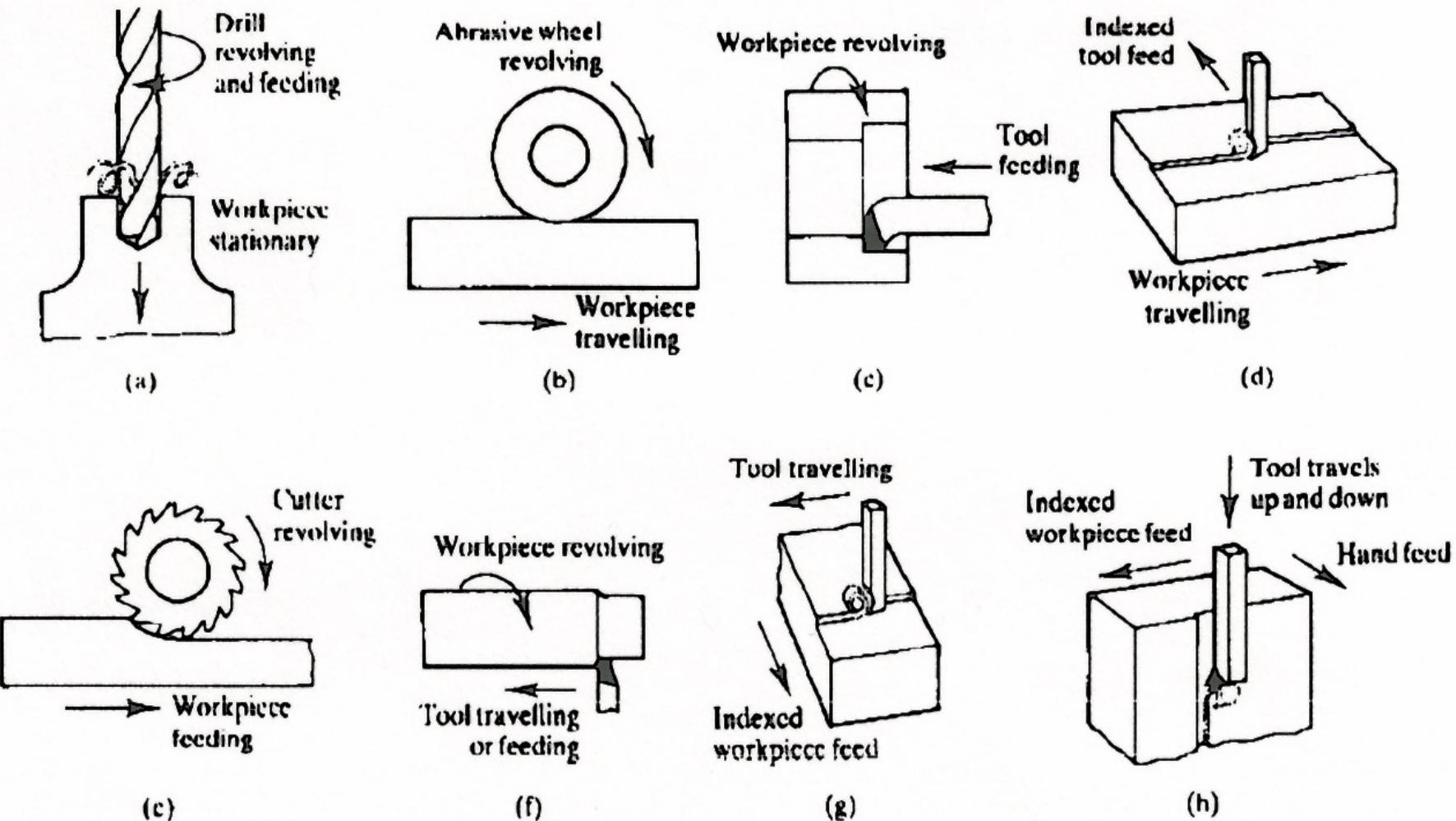


Tasks with environment interaction

- mechanical machining
 - deburring, surface finishing, polishing, assembly,...
- tele-manipulation
 - force feedback improves performance of human operators in master-slave (leader-follower) systems
- contact exploration for shape identification
 - force and velocity/vision sensor fusion allow 2D/3D geometric identification of unknown objects and their contour following
- dexterous robot hands
 - power grasp and fine in-hand manipulation require force/motion cooperation and coordinated control of the multiple fingers
- cooperation of multi-manipulator systems
 - the environment includes one or more other robots with their own dynamic behaviors
- physical human-robot interaction
 - humans as active, dynamic environments that need to be handled under full safety premises ...

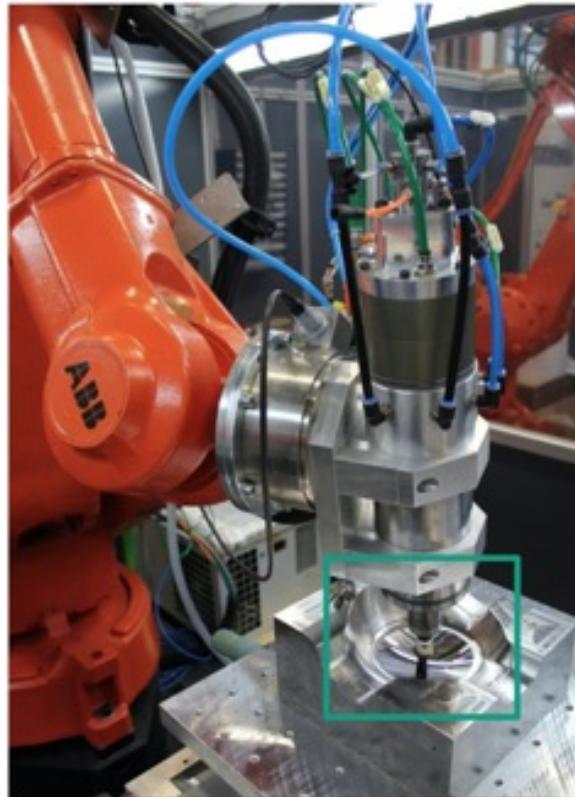


Examples of mechanical machining

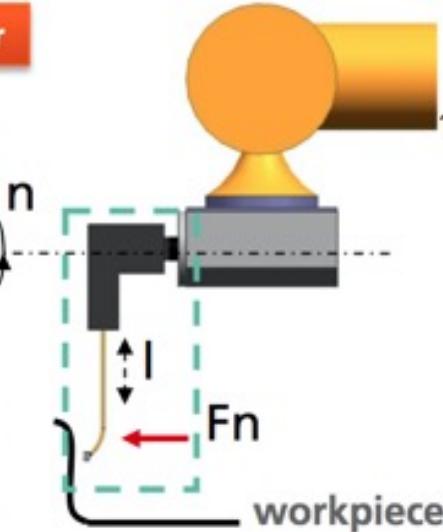
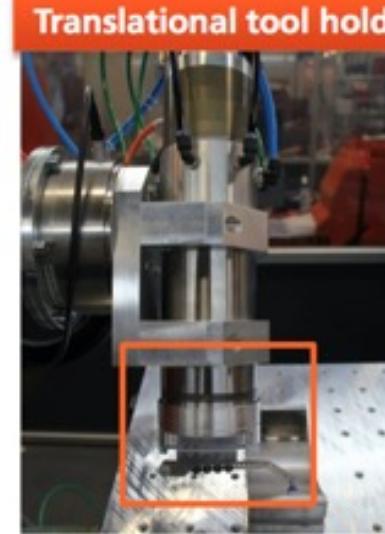




Abrasive finishing of surfaces



Translational tool holder

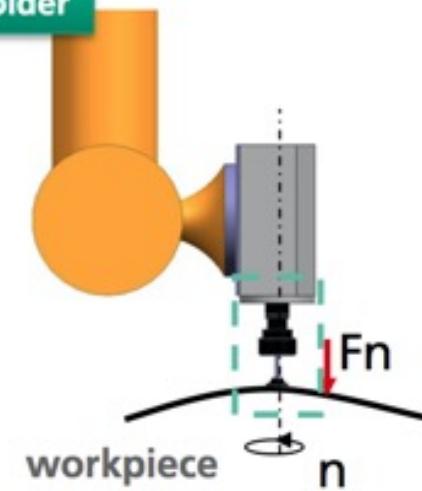


Rotational tool holder



Main properties:

- synchronous motor
- rotation : 100 - 36.000 rpm
- power : 6 kW
- mass : 16 kg
- automated tool exchanger
- pneumatic canals for the force control (x3)





Abrasive finishing of surfaces

video



technological processes: cold forging of surfaces
and hammer peening by pneumatic machine



Non-contact surface finishing

video



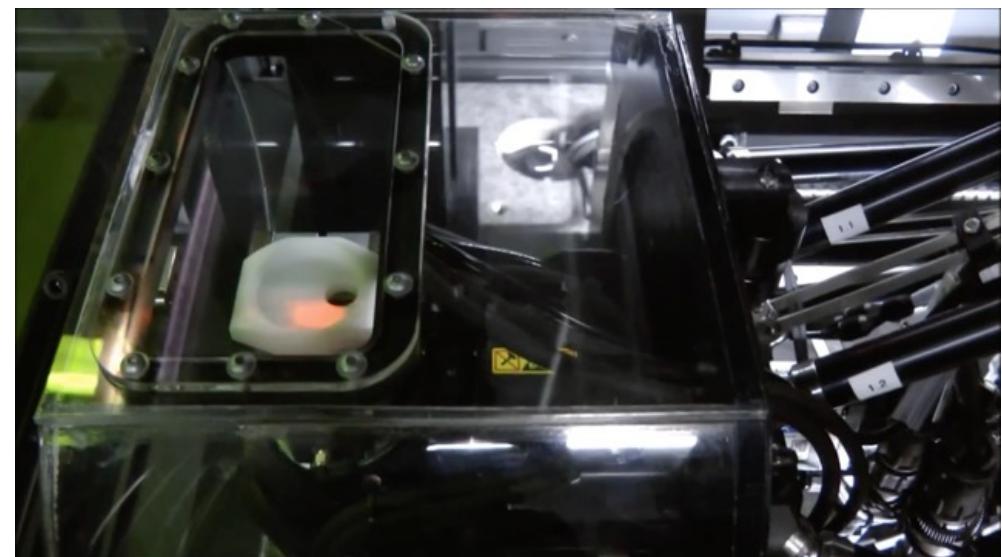
Fluid Jet technology



H2020 EU project for the
Factory of the Future (FoF)

Pulsed Laser technology

video





In all cases ...

- for physical interaction tasks, the **desired motion** specification and execution should be integrated with complementary data for the **desired force**
 → **hybrid force/motion** planning and control objectives
- the exchanged forces/torques at the contact(s) with the environment can be explicitly **set under control** or simply **kept limited** in an indirect way



Evolution of control approaches

a bit of history from the late 70's-mid '80s ...

- **explicit control of forces/torques only [Whitney]**
 - used in quasi-static operations (assembly) in order to avoid deadlocks during part insertion
- **active admittance and compliance control [Paul, Shimano, Salisbury]**
 - contact forces handled through position (**stiffness**) or velocity (**damping**) control of the robot end-effector
 - robot reacts as a compressed **spring** (with **damper**) in selected/all directions
- **impedance control [Hogan]**
 - a desired dynamic behavior is imposed to the robot-environment interaction, e.g., a “model” with forces acting on a **mass-spring-damper**
 - mimics the human arm behavior moving in an unknown environment
- **hybrid force-motion control [Mason]**
 - decomposes the **task space** in complementary sets of directions where **either** force **or** motion is controlled, based on
 - a **purely kinematic** robot model [Raibert, Craig]
 - the actual **dynamic model** of the robot [Khatib]



appropriate for fast and accurate motion in dynamic interaction...

We must introduce
new constraints
to Lagrangian formulation
modified.

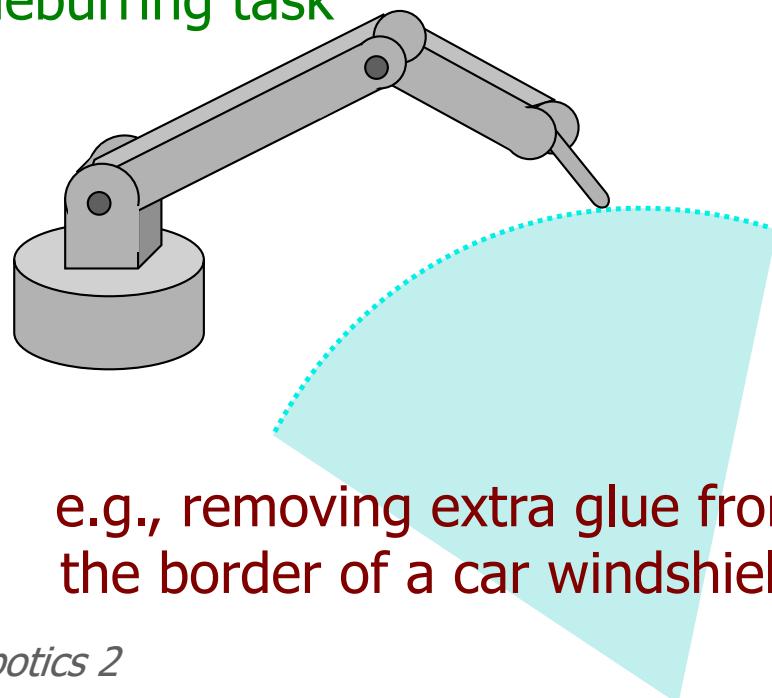


Interaction tasks of interest

interaction tasks with the environment that require

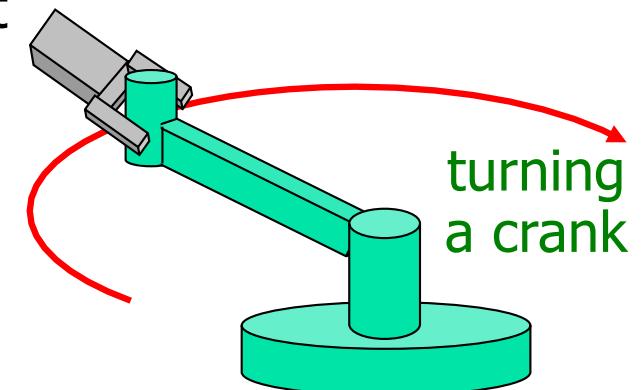
- accurate following/reproduction by the robot end-effector of desired trajectories (even at high speed) defined on the surface of objects
- control of forces/torques applied at the contact with environments having low (**soft**) or high (**rigid**) stiffness

deburring task



e.g., removing extra glue from
the border of a car windshield

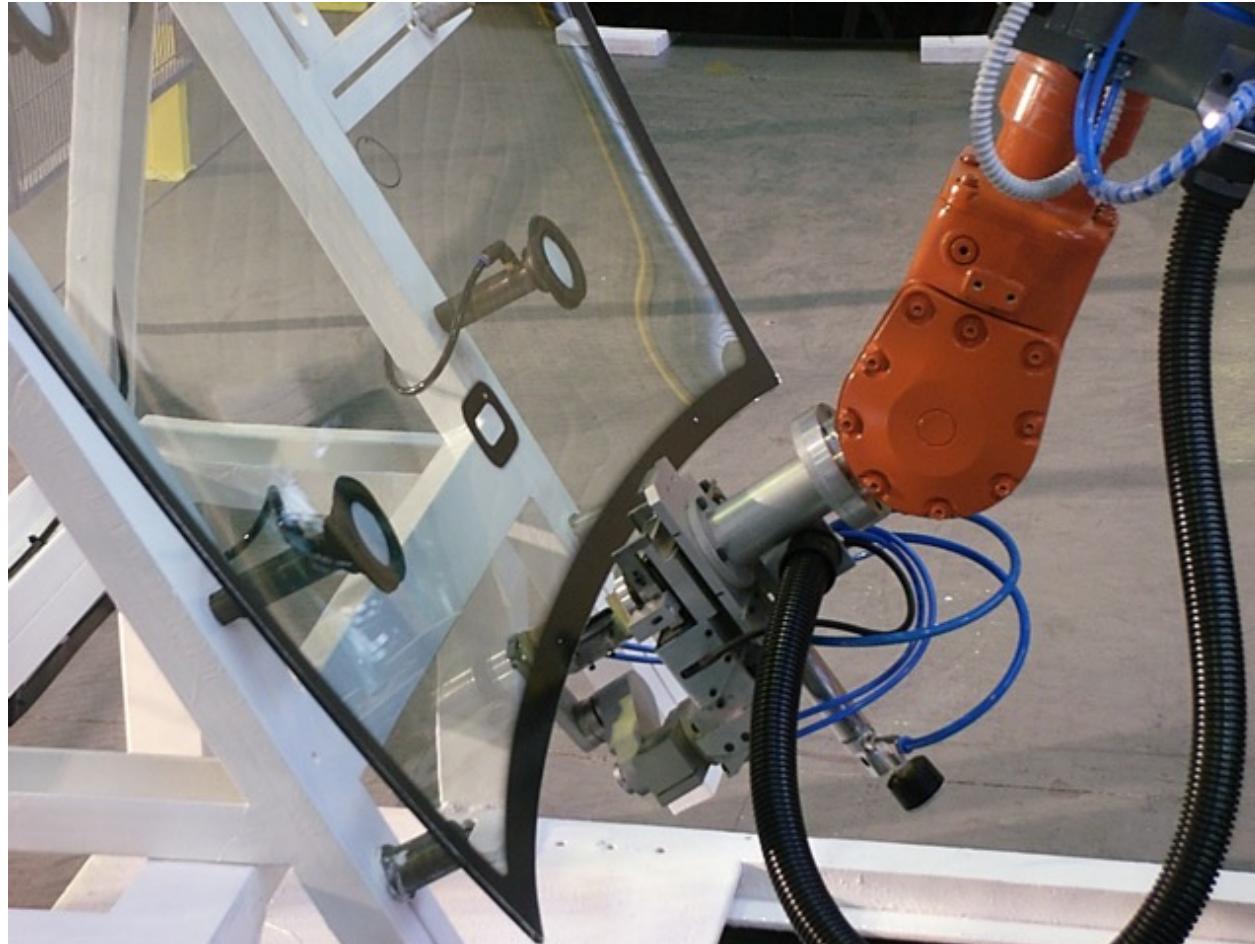
robot



e.g., opening a door



Robotized deburring of windshields



c/o ABB Excellence Center in Cecchina (Roma), 2002



main difference is the type of environment

Impedance vs. Hybrid control

environment model (\leftrightarrow domain of control application)

impedance control

- environment = mechanical system undergoing **small but finite deformations**
- contact forces arise as the result of a balance of two **coupled dynamic systems** (robot+environment)
- ➔ desired dynamic characteristics are assigned to the force/motion interaction

no force w/o pos error

hybrid force/motion control

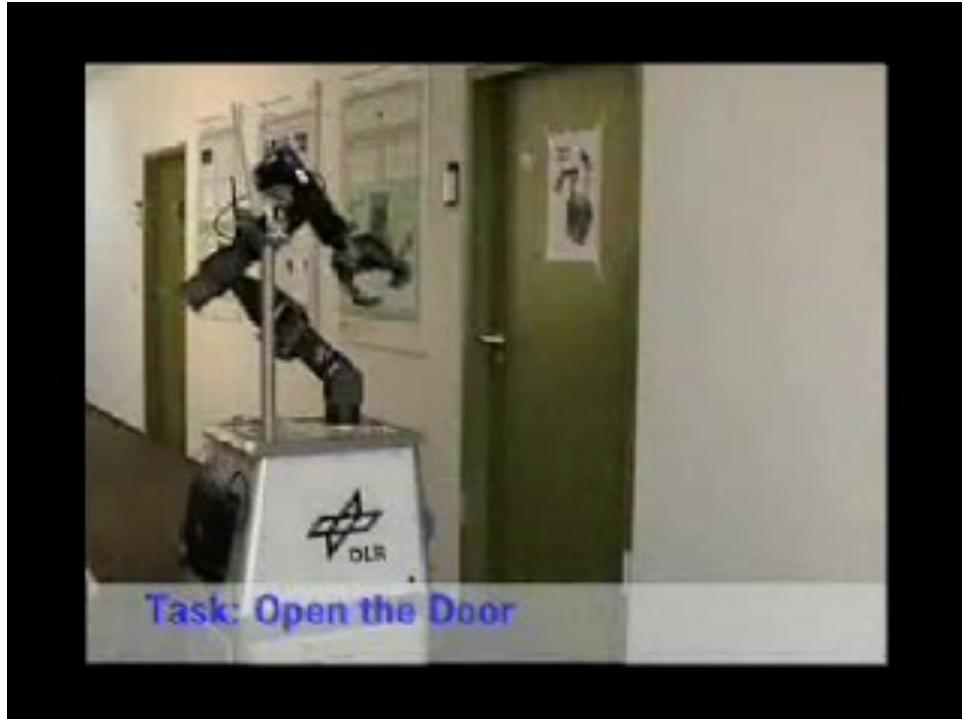
- a **rigid environment** reduces the degrees of freedom of the robot when in (bi-/uni-lateral) contact
- contact forces result from attempts to violate **geometric constraints** imposed by the environment
- ➔ task space is decomposed in sets of directions where **only motion** or **only reaction forces** are feasible

- the required **level of knowledge** about the environment geometry is only **apparently** different between the two control approaches
- however, **measuring contact forces** may not be needed in impedance control, while it always necessary in hybrid force/motion control

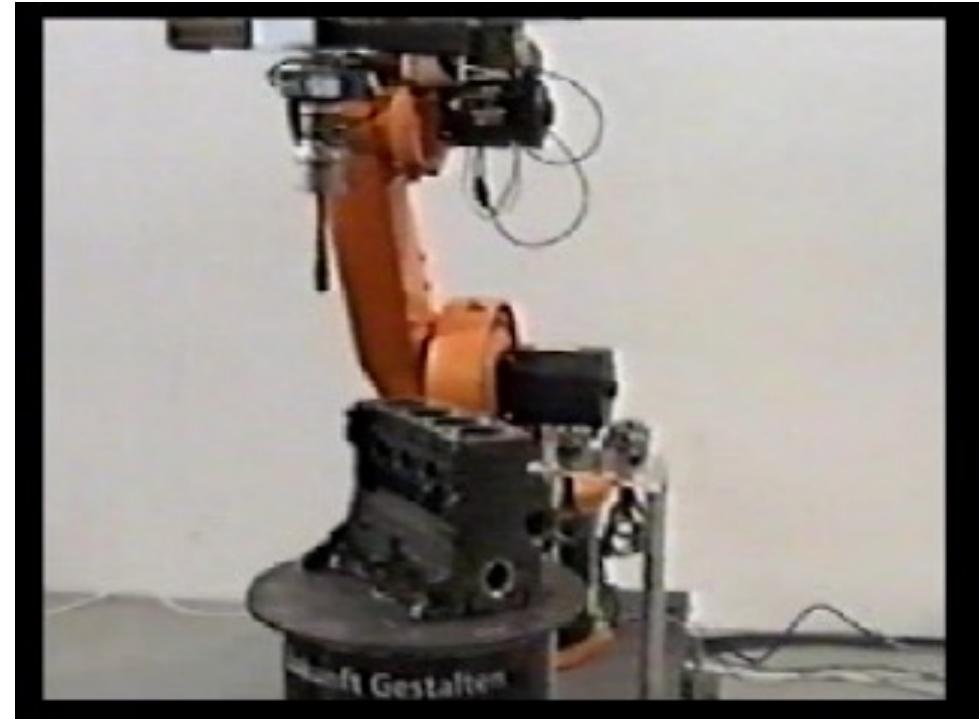


Impedance vs. Hybrid control

- opening a door with a mobile manipulator under **impedance control**
- piston insertion in a motor based on **hybrid control** of force-position (visual)



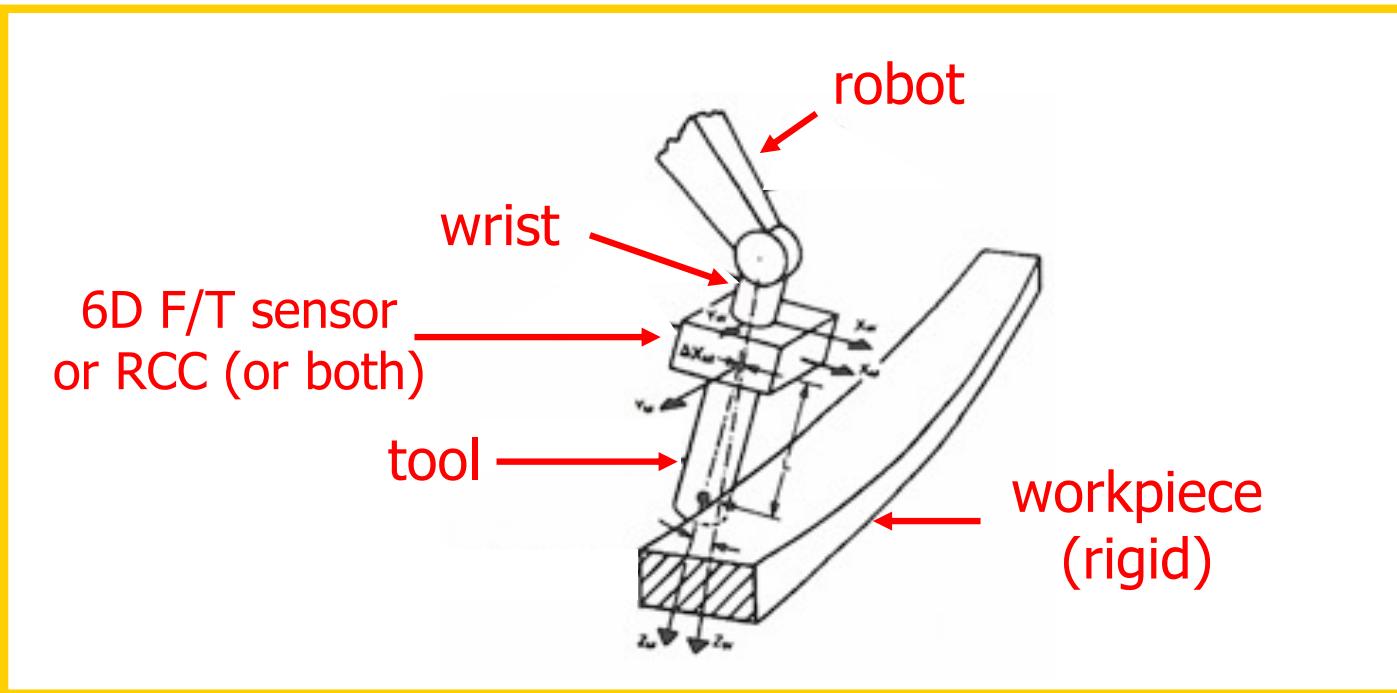
video



video



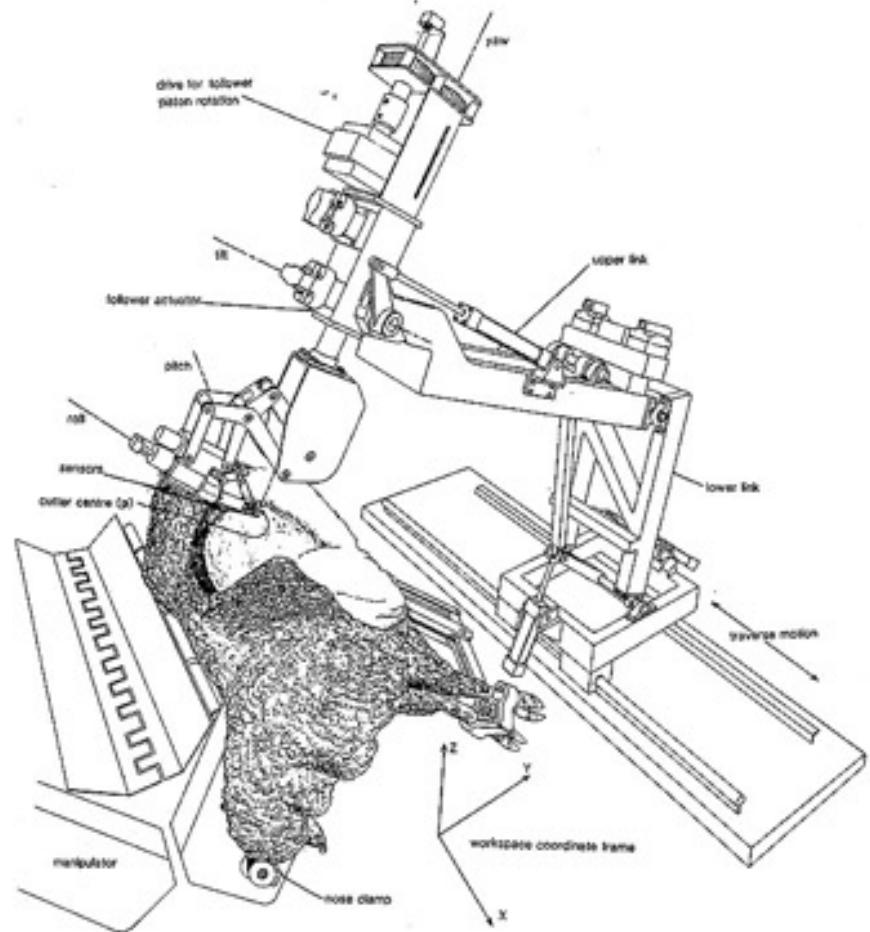
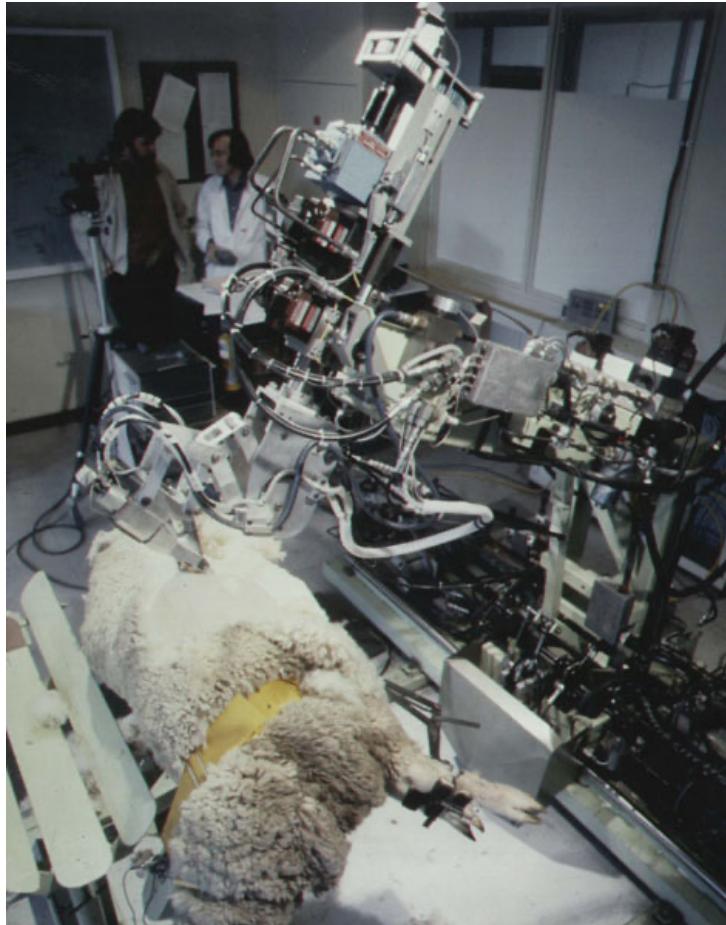
A typical constrained situation ...



the robot end-effector follows in a stable and accurate way the geometric profile of a **very stiff** workpiece, while applying a desired contact force



An unusual compliant situation ...

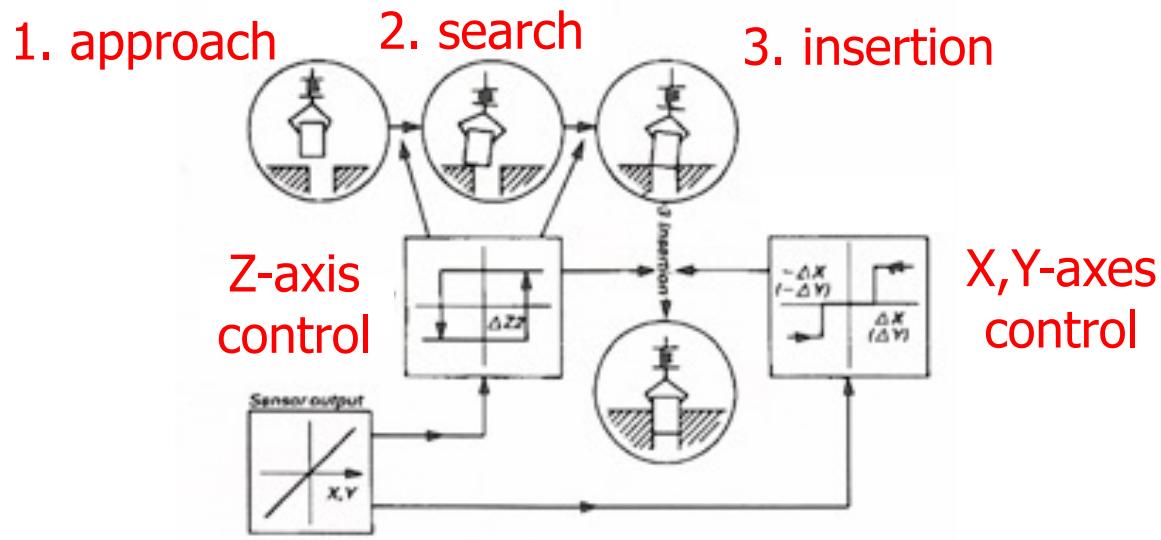


Trevelyan (AUS): Oracle robotic system in a test dated 1981

...is the sheep happy?



A mixed interaction situation



processing/reasoning on force measurements
leads to a sequence of **fine motions**
⇒ correct completion of insertion task with
help of (sufficiently large) passive compliance

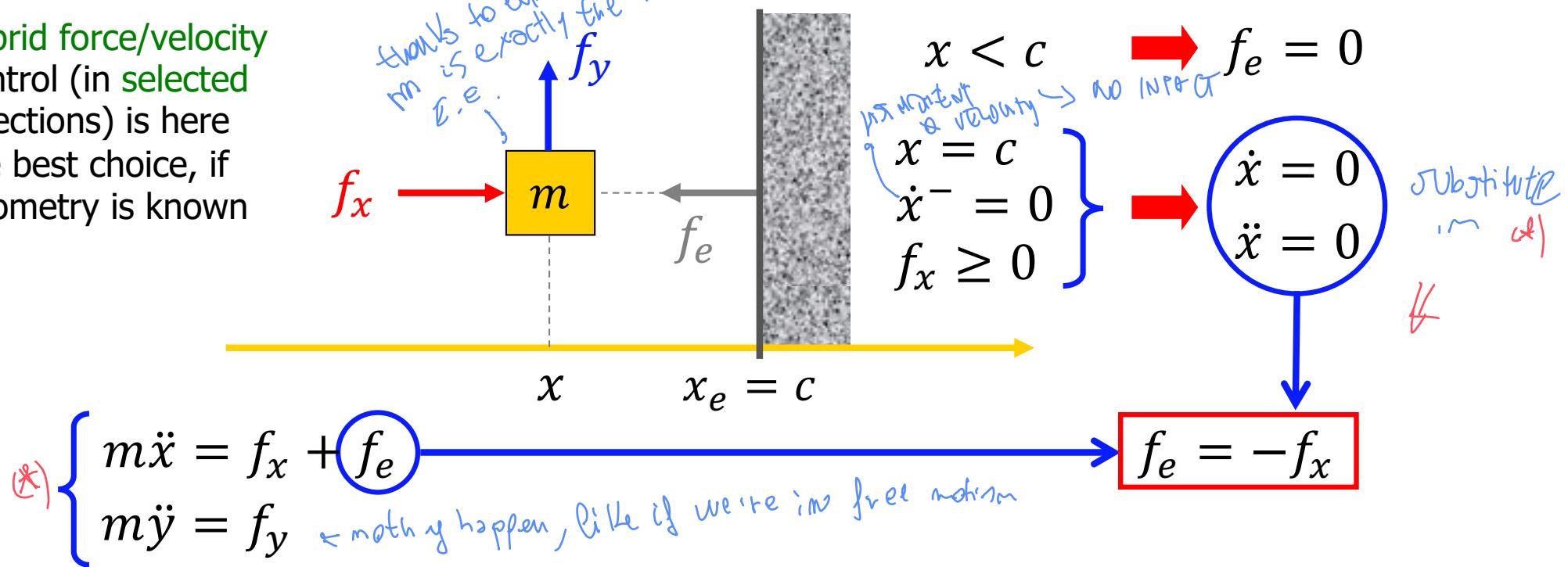


Ideally constrained contact situation

a first possible modeling choice for very stiff environments



hybrid force/velocity control (in selected directions) is here the best choice, if geometry is known



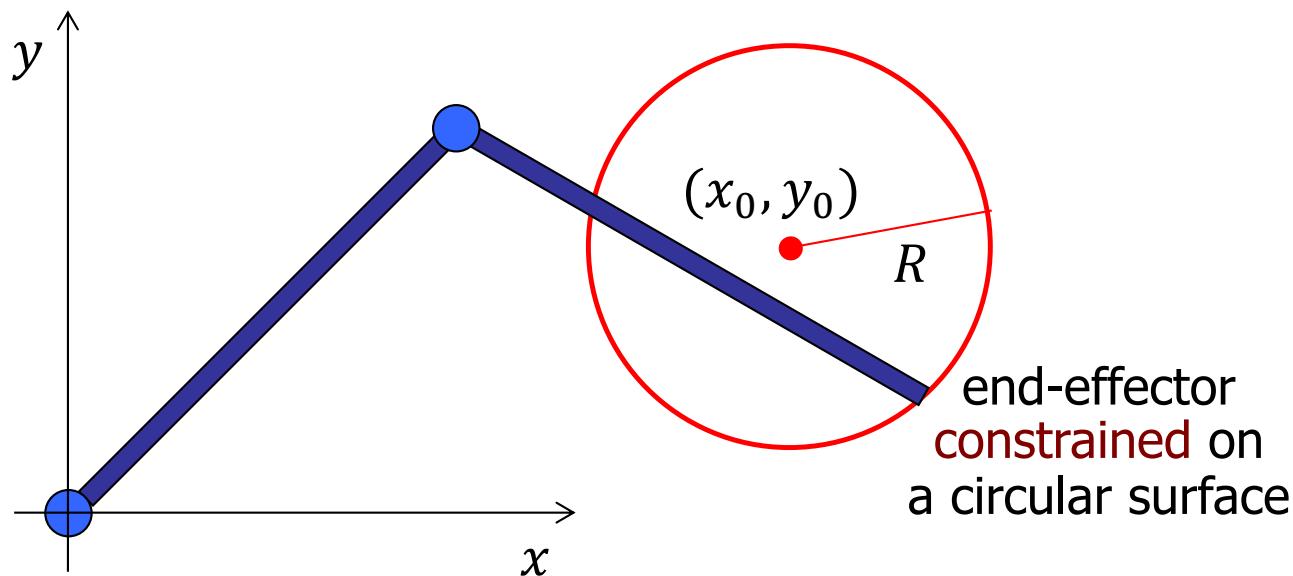
"ideal" = robot (here, a Cartesian mass) + environment are both **infinitely STIFF** (and **no friction** at the contact)

if a possible **impact** ($x = c, \dot{x}^- > 0$) is purely "elastic" (i.e., with conservation of total momentum and total kinetic energy) $\Rightarrow \dot{x}^+ = -\dot{x}^-$ (f_e is an impulse!)



In more complex situations

- how can we describe **more complex contact situations**, where the **end-effector** of an articulated robot (not yet reduced to a Cartesian mass via feedback linearization control) is **constrained** to move **on an environment surface** with nonlinear geometry?
- example: a planar 2R robot with end-effector moving on a circle





Constrained robot dynamics - 1

- consider a robot in free space described by its Lagrange **dynamic model** and a **task output function** (e.g., the end-effector pose)

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

$$r = f(q)$$

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

if m = n

*↓ m blocking
the robot*

- suppose that the task variables are subject to $m < n$ (bilateral) **geometric constraints** in the general form $k(r) = 0$ and define

DEFINE CONSTRAINT $\rightarrow h(q) = k(f(q)) = 0$

- the **constrained robot dynamics** can be derived using again the Lagrange formalism, by defining an **augmented Lagrangian** as

$$L_a(q, \dot{q}, \lambda) = L(q, \dot{q}) + \lambda^T h(q) = T(q, \dot{q}) - U(q) + \lambda^T h(q)$$

where the **Lagrange multipliers** λ (a m -dimensional vector) can be interpreted as the **generalized forces** that arise at the contact when attempting to violate the constraints



Constrained robot dynamics - 2

- applying the **Euler-Lagrange equations** in the extended space of generalized coordinates q and multipliers λ yields

$$\frac{d}{dt} \left(\frac{\partial L_a}{\partial \dot{q}} \right)^T - \left(\frac{\partial L_a}{\partial q} \right)^T = \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}} \right)^T - \left(\frac{\partial L}{\partial q} \right)^T - \left(\frac{\partial}{\partial q} (\lambda^T h(q)) \right)^T = u$$

$$\left(\frac{\partial L_a}{\partial \lambda} \right)^T = h(q) = 0 \quad \xleftarrow{\text{contact forces do NOT produce work}}$$

→ [
$$\begin{aligned} M(q)\ddot{q} + c(q, \dot{q}) + g(q) &= u + A^T(q)\lambda \\ \text{subject to} \quad h(q) &= 0 \end{aligned}$$
] (★) like J^T

where we defined the **Jacobian of the constraints** as the matrix

$$A(q) = \frac{\partial h(q)}{\partial q}$$

that will be assumed of **full row rank** ($= m$) for convenience



Constrained robot dynamics - 3

- we can **eliminate the appearance of the multipliers** as follows
 - differentiate the constraints twice w.r.t. time \rightarrow because it must work at any level
- $$h(q) = 0 \Rightarrow \dot{h} = \frac{\partial h(q)}{\partial q} \dot{q} = A(q)\dot{q} = 0 \Rightarrow \ddot{h} = A(q)\ddot{q} + \dot{A}(q)\dot{q} = 0$$
- substitute the joint accelerations from the dynamic model (★)
(dropping dependencies)

$$AM^{-1}(u + A^T\lambda - c - g) + \dot{A}\dot{q} = 0$$

- solve for the multipliers

the inertia-weighted
pseudoinverse of the
constraint Jacobian A

$$\begin{aligned} \lambda &= (AM^{-1}A^T)^{-1}(AM^{-1}(c + g - u) - \dot{A}\dot{q}) \\ &= (A_M^\#)^T(c + g - u) - (AM^{-1}A^T)^{-1}\dot{A}\dot{q} \end{aligned}$$

invertible $m \times m$ matrix,
when A is full rank \Rightarrow invertible

to be replaced in the dynamic model...

constraint
forces λ are
uniquely
determined
by the robot
state (q, \dot{q})
and **input u !!**



Constrained robot dynamics - 4

- the final **constrained dynamic model** can be rewritten as

$$M(q)\ddot{q} = \left[I - A^T(q)(A_M^\#(q))^T \right] (u - c(q, \dot{q}) - g(q)) - M(q)A_M^\#(q)\dot{A}(q)\dot{q}$$

dynamically consistent projection matrix

where $A_M^\#(q) = M^{-1}(q)A^T(q)(A(q)M^{-1}(q)A^T(q))^{-1}$ and with

$$\lambda = (A_M^\#(q))^T(c(q, \dot{q}) + g(q) - u) - (A(q)M^{-1}(q)A^T(q))^{-1}\dot{A}(q)\dot{q}$$

- if the robot state $(q(0), \dot{q}(0))$ at time $t = 0$ satisfies the constraints, i.e.,

$$h(q(0)) = 0, \quad A(q(0))\dot{q}(0) = 0$$

then the robot evolution described by the above dynamics will be consistent with the constraints **for all $t \geq 0$** and **for any $u(t)$**

- this is a useful **simulation model** (constrained **direct** dynamics)

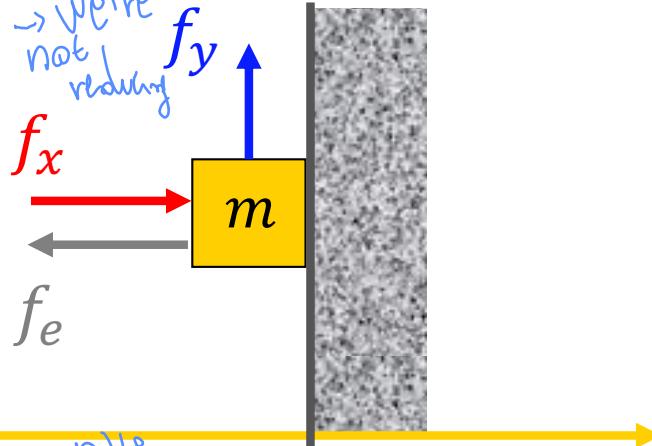


Example – ideal mass constrained robot dynamics

2 generalised coord.

$$q = \begin{pmatrix} x \\ y \end{pmatrix}$$

$$u = \begin{pmatrix} f_x \\ f_y \end{pmatrix}$$



$$M\ddot{q} = u$$

robot dynamics
in free motion

$$M = \begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix}$$

for $x = c \Rightarrow$ we're in the wall

$$h(q) = x - c = 0 \Rightarrow A(q) = \begin{pmatrix} 1 & 0 \end{pmatrix} \Rightarrow A_M^\#(q) = \dots = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$\left(I - A^T(q)(A_M^\#(q))^T \right) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

dynamically consistent
projection matrix

$$\lambda = -(A_M^\#(q))^T u = -(1 \ 0) u = -f_x$$

multiplier (contact force f_e)

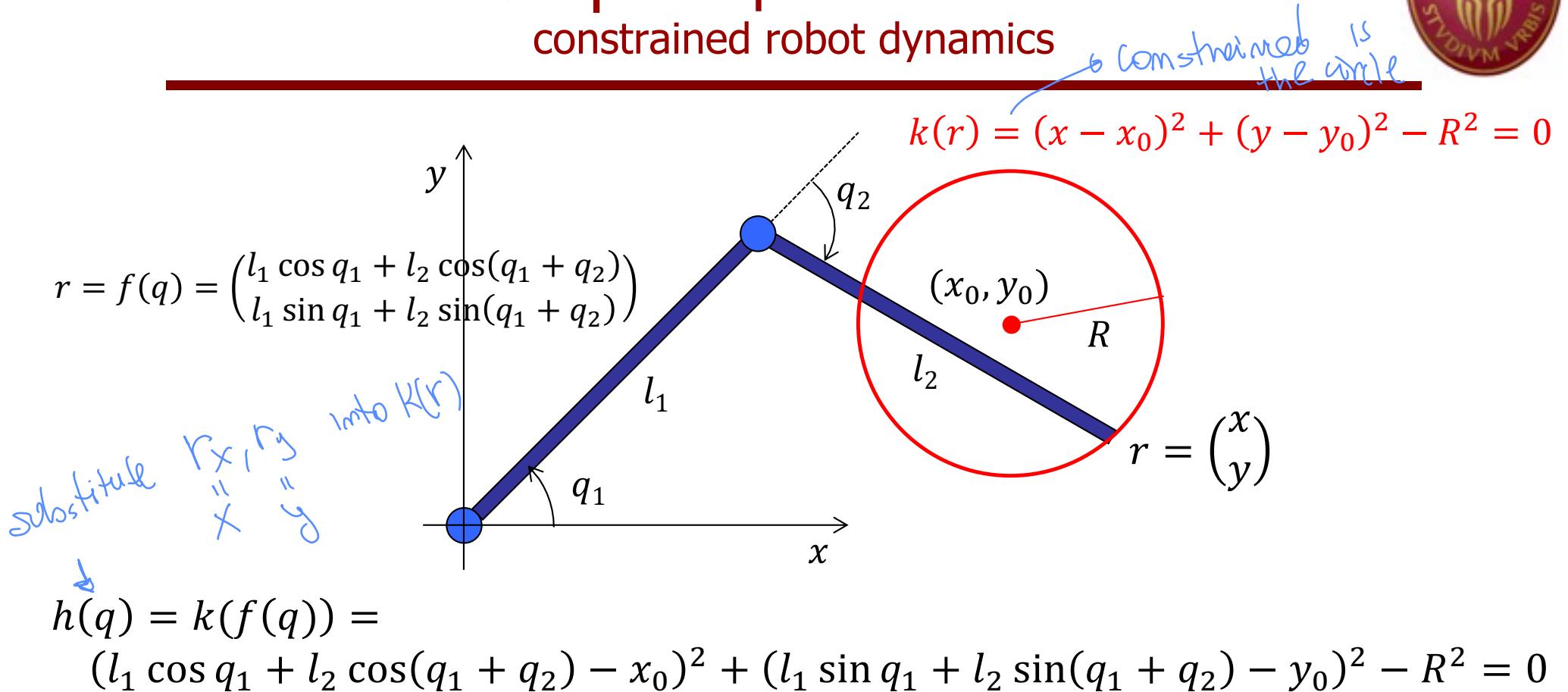
$$M \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} = M\ddot{q} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} u = \begin{pmatrix} 0 \\ f_y \end{pmatrix}$$

constrained
robot dynamics



Example – planar 2R robot

constrained robot dynamics



$$\dot{h} = \frac{\partial k}{\partial r} \frac{\partial r}{\partial q} \dot{q} = [2(x - x_0) \quad 2(y - y_0)] J_r(q) \dot{q}$$

$$= [2(l_1 c_1 + l_2 c_{12} - x_0) \quad 2(l_1 s_1 + l_2 s_{12} - y_0)] J_r(q) \dot{q} = A(q) \dot{q}$$



Reduced robot dynamics - 1

- by imposing m constraints $h(q) = 0$ on the n generalized coordinates q , it is also possible to **reduce** the description of the constrained robot dynamics to a **$n - m$ dimensional** configuration space
- start from constraint matrix $A(q)$ and **select** a matrix $D(q)$ such that

$$\begin{pmatrix} A(q) \\ D(q) \end{pmatrix} \text{ is a } \begin{matrix} \text{nonsingular} \\ n \times n \end{matrix} \text{ matrix} \quad \rightarrow \quad \begin{pmatrix} A(q) \\ D(q) \end{pmatrix}^{-1} = (E(q) \quad F(q))$$

- define the $(n - m)$ -dimensional vector of **pseudo-velocities** ν as the linear combination (at a given q) of the robot generalized velocities

$$\nu = D(q)\dot{q} \quad \rightarrow \quad \dot{\nu} = D(q)\ddot{q} + \dot{D}(q)\dot{q}$$

- inverse relationships (from “pseudo” to “generalized” velocities and accelerations) are given by

$$\dot{q} = F(q)\nu \quad \ddot{q} = F(q)\dot{\nu} - (E(q)\dot{A}(q) + F(q)\dot{D}(q))F(q)\nu$$

↔ properties of **block products** in inverse matrices have been used for eliminating the appearance of \dot{F} (often F is only known **numerically**)

Reduced robot dynamics – 2

whiteboard ...



$$\begin{pmatrix} A(q) \\ D(q) \end{pmatrix}^{-1} = (E(q) \quad F(q)) \quad \text{a number of properties from this definition...}$$

two matrix inverse products

two matrix inverse products

$$\begin{pmatrix} A(q) \\ D(q) \end{pmatrix} (E(q) \quad F(q)) = \begin{pmatrix} A(q)E(q) & A(q)F(q) \\ D(q)E(q) & D(q)F(q) \end{pmatrix} = \begin{pmatrix} I_{m \times m} \\ 0 \\ \textcolor{blue}{I}_{(n-m) \times (n-m)} \end{pmatrix}$$

three useful identities!

$$(E(q) \quad F(q)) \begin{pmatrix} A(q) \\ D(q) \end{pmatrix} = E(q)A(q) + F(q)D(q) = I_{n \times n}$$

 differentiating w.r.t. time $\dot{E}A + E\dot{A} + \dot{F}D + F\dot{D} = 0$

from pseudo-velocity $\nu = D(q)\dot{q}$
 since F is a right inverse of the
 full row rank matrix D ($DF = I$)

$$\dot{q} = F(q)v \quad D\dot{q} = DFv \\ = v)$$

→ differentiating w.r.t. time $\dot{q} = F(q)v$

$$\ddot{q} = F\dot{v} + \dot{F}v = F\dot{v} + (\dot{F}D)\dot{q} \stackrel{(\triangleleft)}{=} F\dot{v} - (\dot{E}A + E\dot{A} + F\dot{D})Fv$$

$$= F(q)\dot{v} - (E(q)\dot{A}(q) + F(q)\dot{D}(q))F(q)v$$



Reduced robot dynamics - 3

- consider again the dynamic model (★), dropping dependencies

$$M\ddot{q} + c + g = u + A^T \lambda$$

- since $AE = I$, multiplying on the left by E^T isolates the multipliers

$$E^T(M\ddot{q} + c + g - u) = \lambda$$

- since $AF = 0$, multiplying on the left by F^T eliminates the multipliers

$$F^T M \ddot{q} = F^T(u - c - g)$$

- substituting in the latter the generalized accelerations and velocities with the pseudo-accelerations and pseudo-velocities leads finally to

invertible
 $(n-m) \times (n-m)$ positive definite matrix \rightarrow $(F^T M F) \dot{v} = F^T(u - c - g + M(E\dot{A} + F\dot{D})Fv)$

which is the reduced $(n - m)$ -dimensional dynamic model

- similarly, the expression of the multipliers becomes

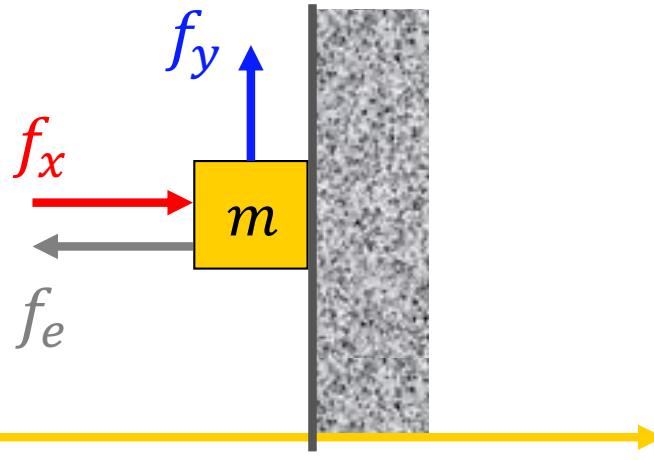
$$\lambda = E^T(MF\dot{v} - M(E\dot{A} + F\dot{D})Fv + c + g - u) \quad (\S)$$



Example – ideal mass reduced robot dynamics

$$q = \begin{pmatrix} x \\ y \end{pmatrix}$$

$$u = \begin{pmatrix} f_x \\ f_y \end{pmatrix}$$



$$M \ddot{q} = u$$

robot dynamics
in free motion

$$M = \begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix}$$

$$h(q) = x - c = 0 \Rightarrow A = \begin{pmatrix} 1 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} A \\ D \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} E & F \end{pmatrix}$$

➡ $v = D\dot{q} = \dot{y}$ pseudo-velocity

$$\lambda = E^T(MFv - u)$$

$$= (1 \ 0) \left(\begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \ddot{y} - \begin{pmatrix} f_x \\ f_y \end{pmatrix} \right) = -(1 \ 0) \begin{pmatrix} f_x \\ f_y \end{pmatrix} = -f_x$$

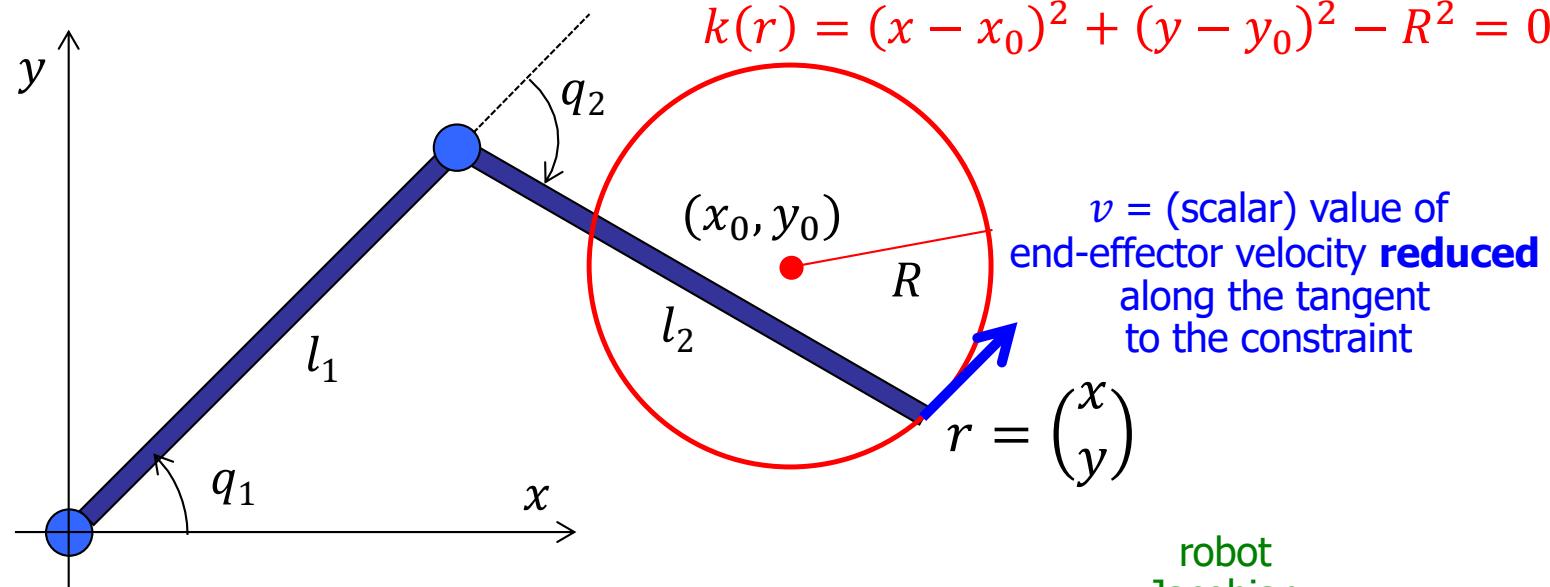
$$(F^T M F)v = (0 \ 1) \begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} v = m \ddot{y} = f_y = F^T u$$

multiplier
(contact
force f_e)

reduced
robot dynamics



Example – planar 2R robot reduced robot dynamics



$$A(q) = [2(x - x_0) \quad 2(y - y_0)] J_r(q)$$

$$= [2(l_1 c_1 + l_2 c_{12} - x_0) \quad 2(l_1 s_1 + l_2 s_{12} - y_0)] J_r(q)$$

a feasible selection of matrix $D(q)$

$$D(q) = \left[-\frac{1}{2}(y - y_0) \quad \frac{1}{2}(x - x_0) \right] J_r(q) \rightarrow \det \begin{pmatrix} A(q) \\ D(q) \end{pmatrix} = R^2 \cdot \det J_r(q) \neq 0$$

$$\begin{pmatrix} A(q) \\ D(q) \end{pmatrix}^{-1} = (E(q) \quad F(q)) \rightarrow \boxed{v} = D(q)\dot{q} \rightarrow \dot{q} = F(q)v = J_r^{-1}(q) \begin{pmatrix} -2(y - y_0)/R^2 \\ 2(x - x_0)/R^2 \end{pmatrix} v$$

a scalar



Control based on reduced robot dynamics

- the reduced $n - m$ dynamic expressions are more compact but also more complex and less used for simulation purposes than the n -dimensional constrained dynamics
- however, they are useful for **control design** (reduced **inverse** dynamics)
- in fact, it is straightforward to verify that the **feedback linearizing** control law

$$u = (c + g - M(E\dot{A} + F\dot{D})F\nu) + MFu_1 - A^T u_2$$

applied to the **reduced robot dynamics** and to the **expression (§) of the multipliers** leads to the closed-loop system

$$\dot{\nu} = u_1 \quad \lambda = u_2$$

Note: these are **exactly** in the form of the ideal mass example of slide #24, with $\nu = \dot{y}$, $u_1 = f_y/m$, $\lambda = f_e$, $u_2 = -f_x$ (being $n = 2$, $m = 1$, $n - m = 1$)



Compliant contact situation

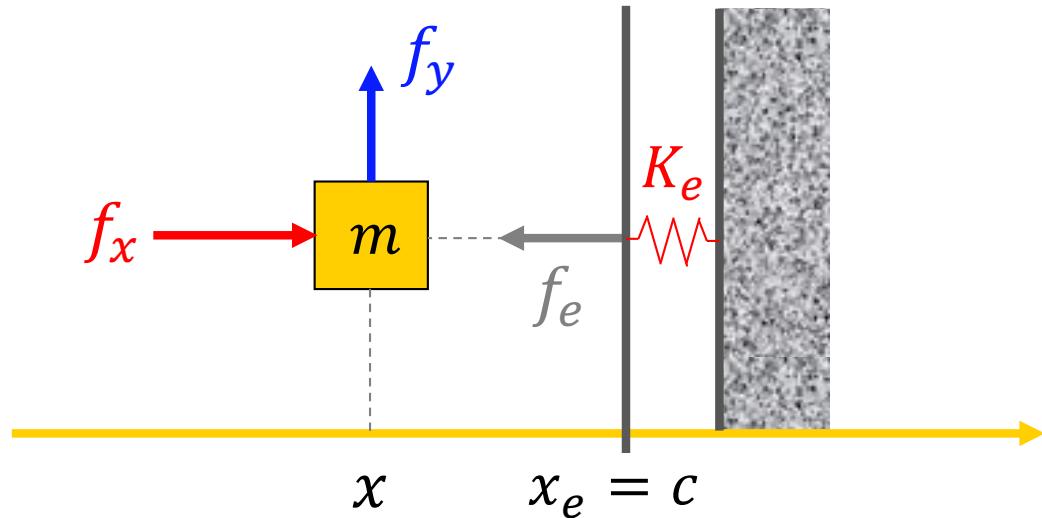
a second possible modeling choice for softer environments

← there is continuity of phenomena

↳ easier to simulate

compliance/impedance control (in all directions) is here a good choice that allows to handle

- uncertain position
- uncertain orientation of the wall



$$\begin{cases} m\ddot{x} = f_x + f_e \\ m\ddot{y} = f_y \end{cases} \quad \begin{cases} x < c & \rightarrow f_e = 0 \\ x \geq c & \rightarrow f_e = K_e(x - x_e) \end{cases}$$

↳ for a proportional deformation

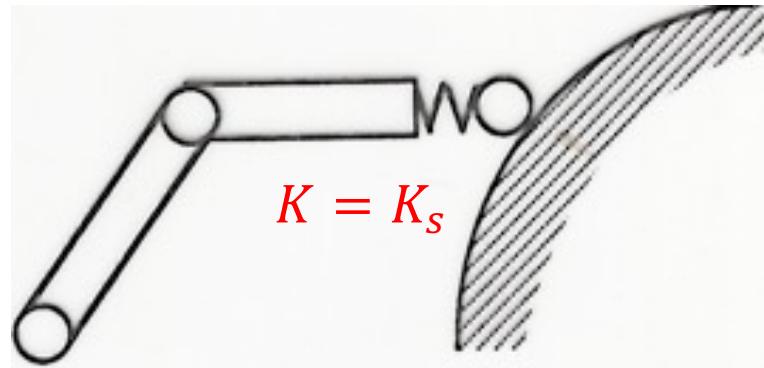
with $K_e > 0$ being the **stiffness** of the environment



Robot-environment contact types

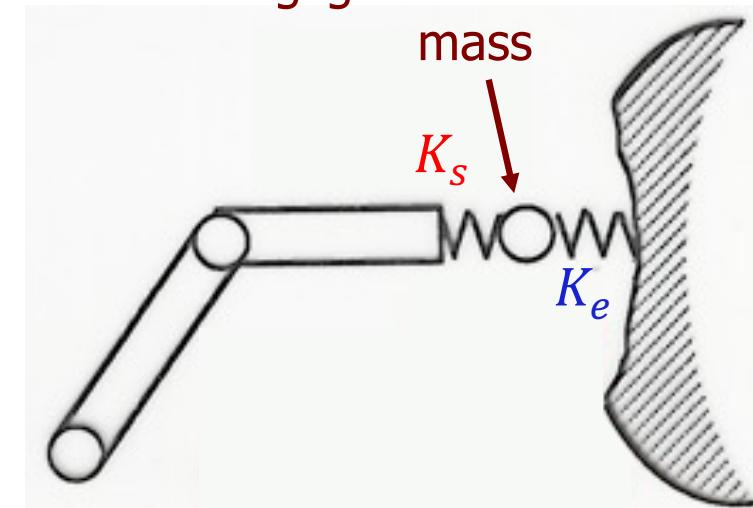
modeled by a single elastic constant

compliant
force sensor

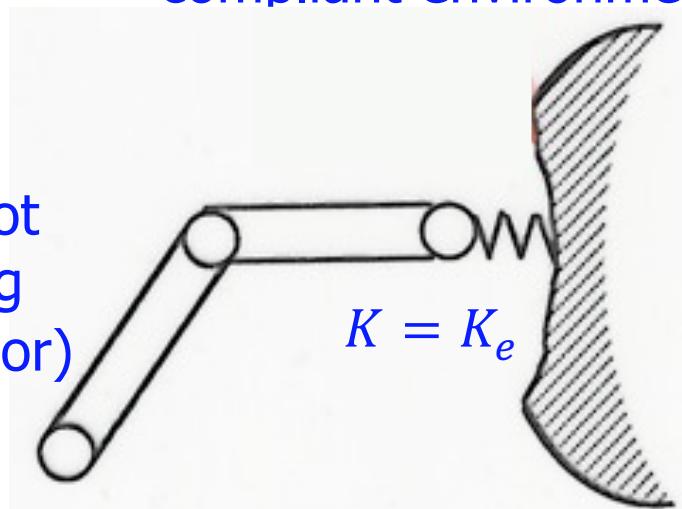


rigid environment

negligible intermediate
mass



rigid robot
(including
force sensor)



compliant environment

$$\frac{1}{K} = \frac{1}{K_s} + \frac{1}{K_e} \rightarrow K = \frac{K_s K_e}{K_s + K_e}$$

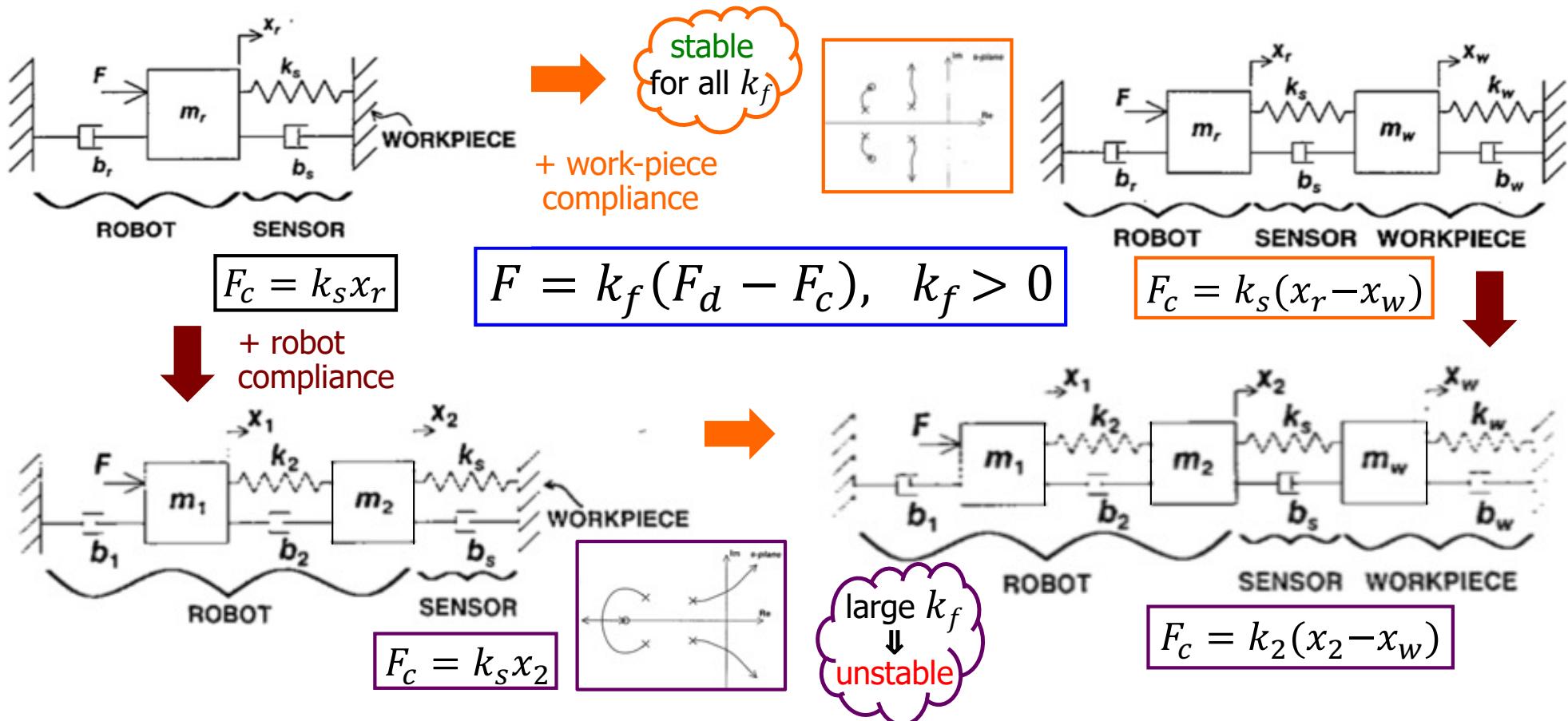
series of springs =
sum of compliances
(inverse of stiffnesses)



Force control

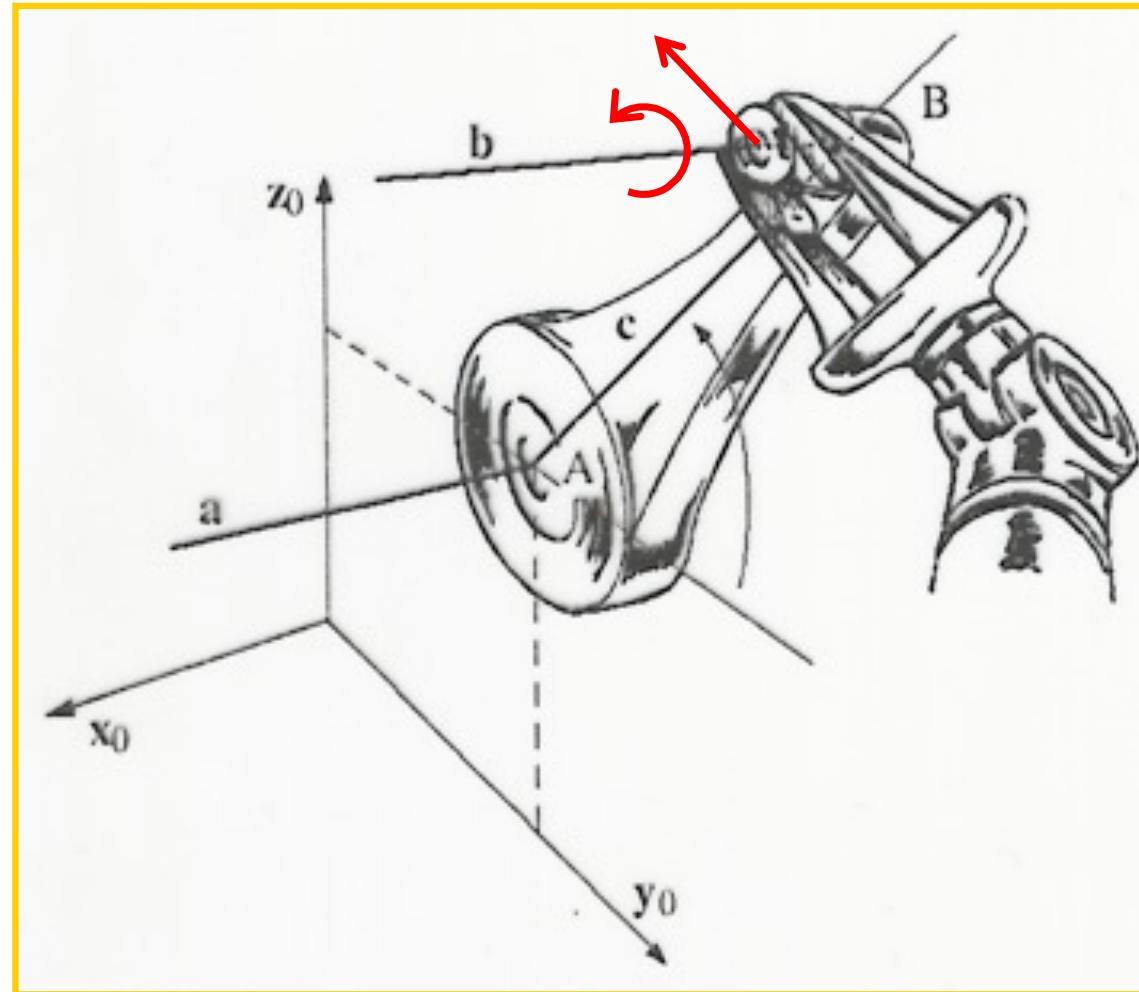
1-dof robot-environment linear dynamic models

- with a **force sensor** (having stiffness k_s and damping b_s) measuring the contact force F_c
- stability** analysis of a **proportional** control loop for regulation of the contact force (to a desired constant value F_d) can be made using the **root-locus method** (for a varying k_f)
- by including/excluding **work-piece compliance** and/or robot (transmission) compliance





Tasks requiring hybrid control



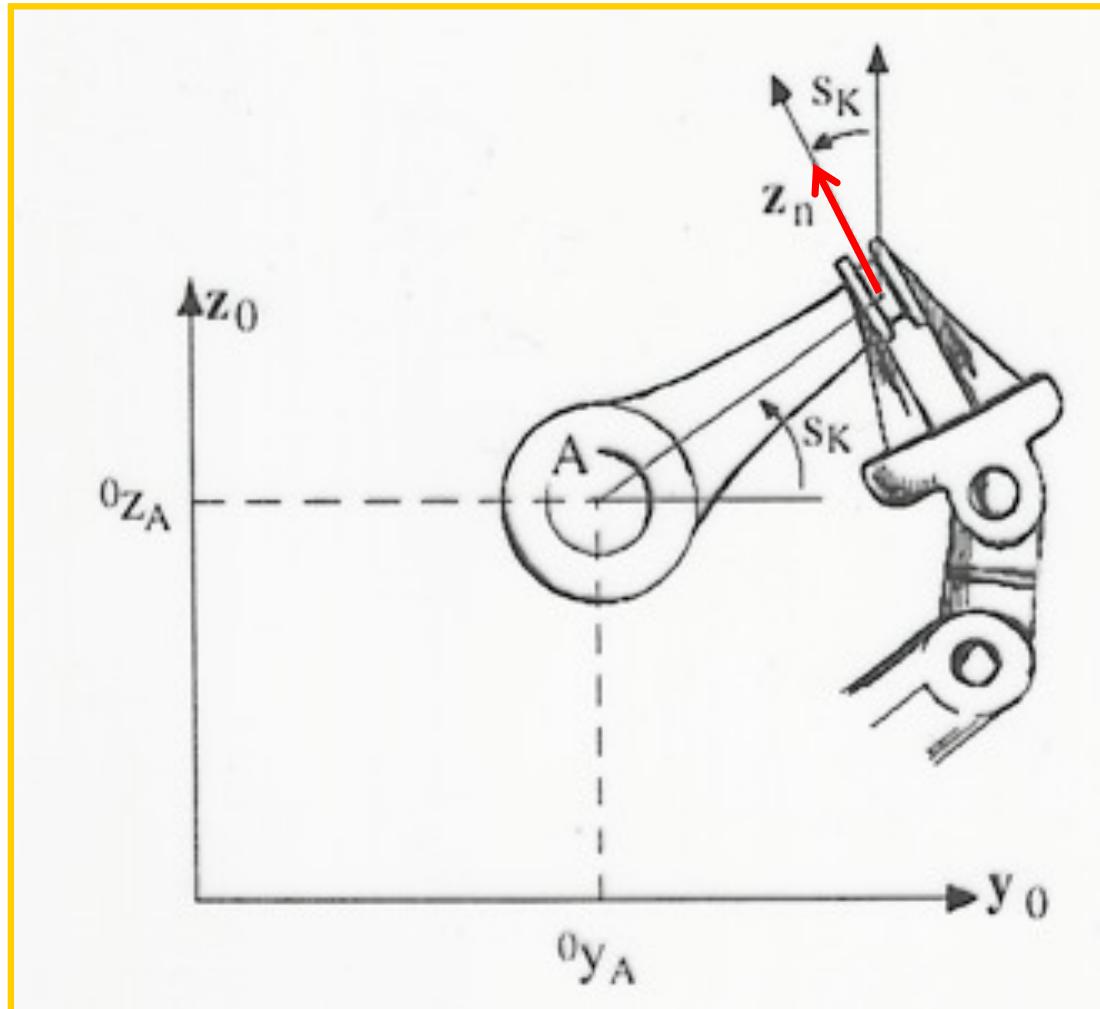
two generalized directions of instantaneous free motion at the contact:
tangential velocity & angular velocity around handle axis

↑
four directions of generalized reaction forces at the contact

the robot should turn a crank having a **free-spinning** handle



Tasks requiring hybrid control



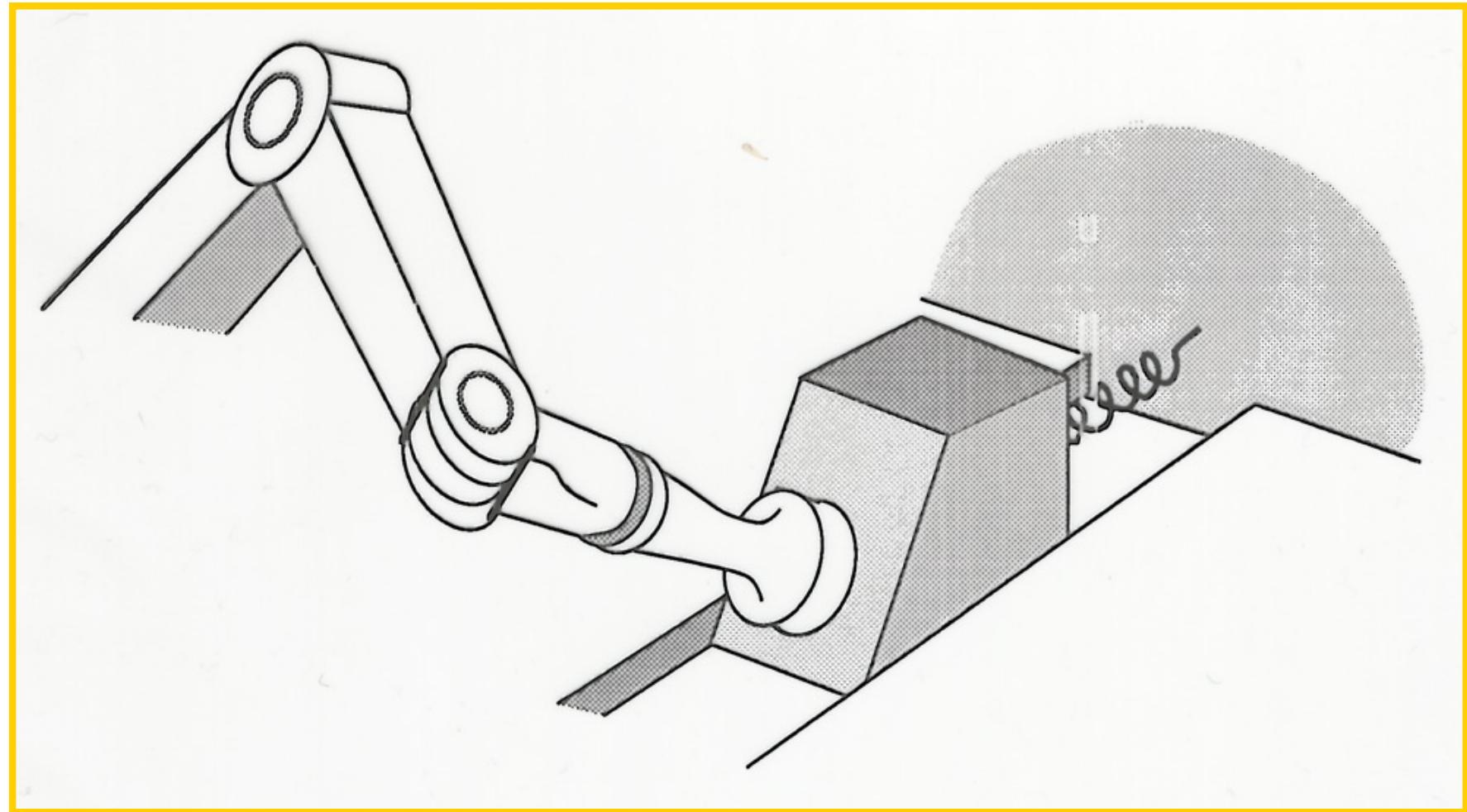
one direction only
of instantaneous
free motion
at the contact:
tangential velocity

↔
five directions
of generalized
reaction forces
at the contact

the robot should turn a crank
having a **fixed handle**



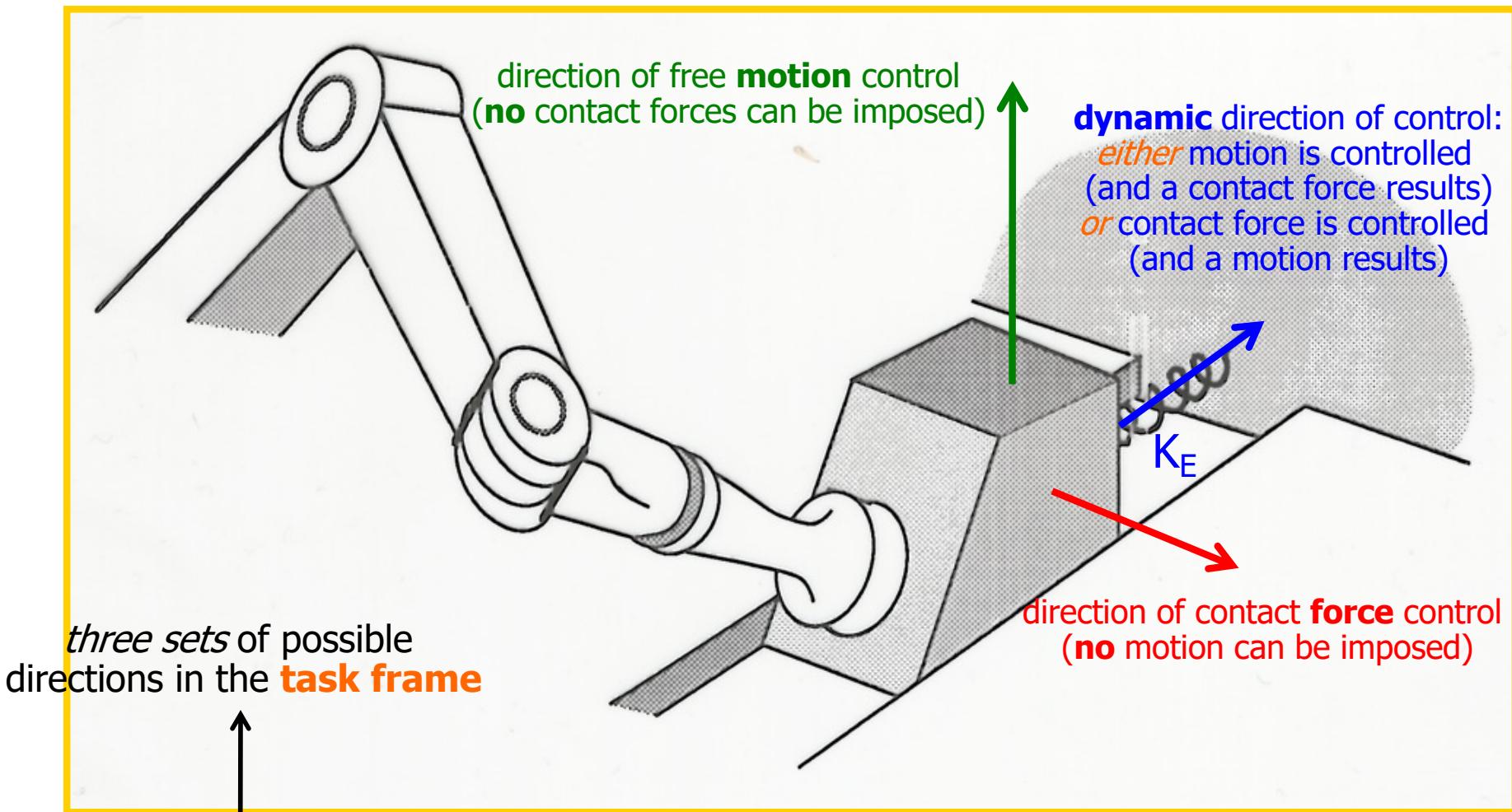
Tasks requiring hybrid control



the robot should push a mass
elastically coupled to a wall and constrained in a guide



Tasks requiring hybrid control



generalized hybrid modeling and control for dynamic environments

A. De Luca, C. Manes: IEEE Trans. Robotics and Automation, vol. 10, no. 4, 1994



Robotics 2

Impedance Control

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





mass + spring
+ damper
we chose model
✓

Impedance control

- imposes a desired **dynamic behavior** to the interaction between robot end-effector and environment
- the desired performance is specified through a **generalized dynamic impedance**, namely a complete set of **mass-spring-damper** equations (typically chosen as linear and decoupled, but also nonlinear)
- a model describing how reaction forces are generated in association with environment deformation is **not** explicitly required
- suited for tasks in which **contact forces** should be “kept small”, while their accurate regulation is not mandatory
- since a control loop based on **force error** is missing, **contact forces** are only indirectly assigned **by controlling position**
- the choice of a specific stiffness in the impedance model along a Cartesian direction results in a **trade-off** between contact forces and position accuracy in that direction

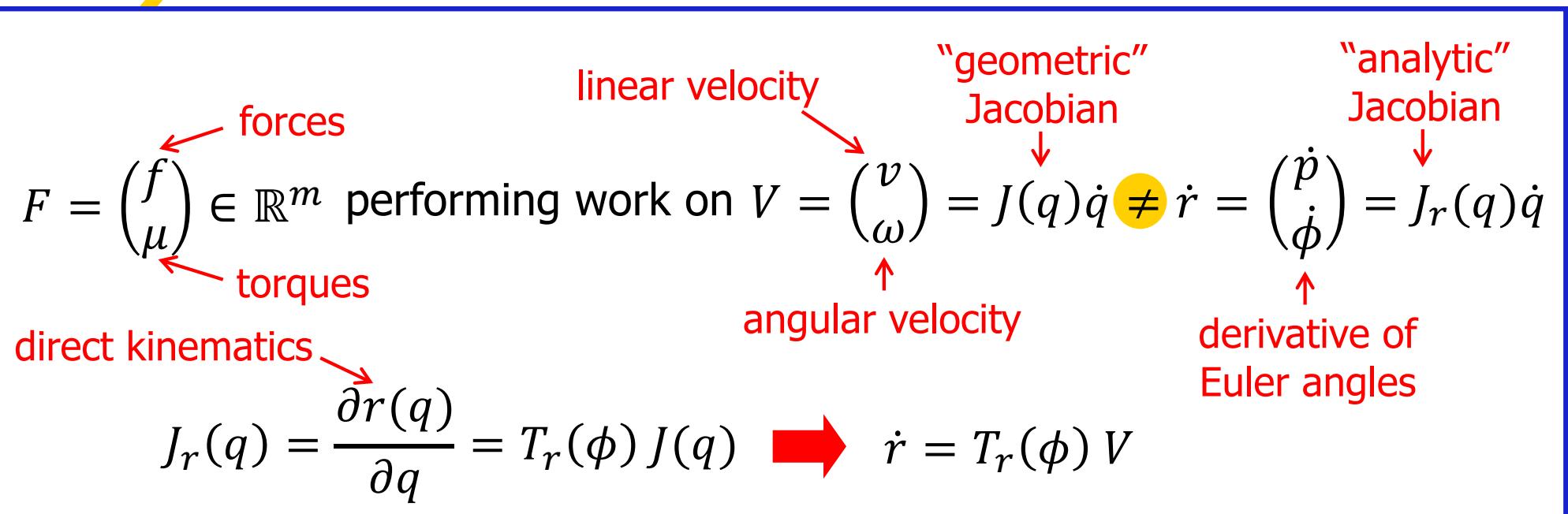


Dynamic model of a robot in contact

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

$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = u + J^T(q)F$$

forces due to contact
generalized Cartesian force



$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = u + J_r^T(q)F_r$$

PREFERRED
with

$$F_r = T_r^{-T}(\phi) F$$

generalized forces performing work on \dot{r}

transformation based on virtual work

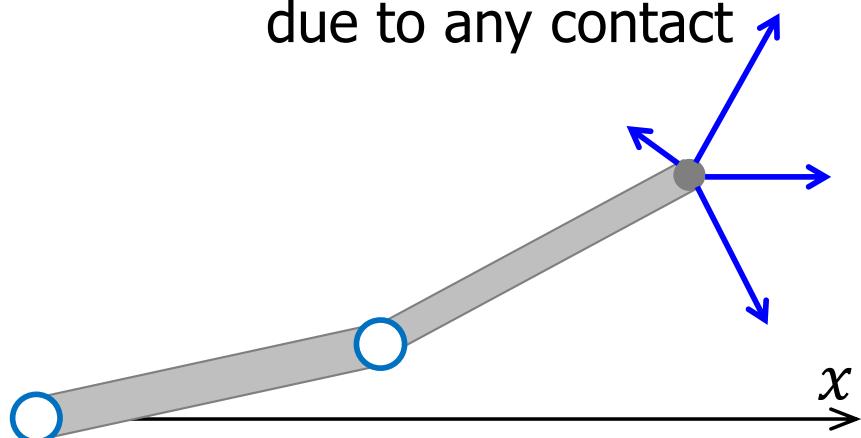


Contact forces vs. constraint forces

whiteboard...

$$M(q)\ddot{q} + \dots = u + J^T(q)F$$

every possible force F
due to any contact



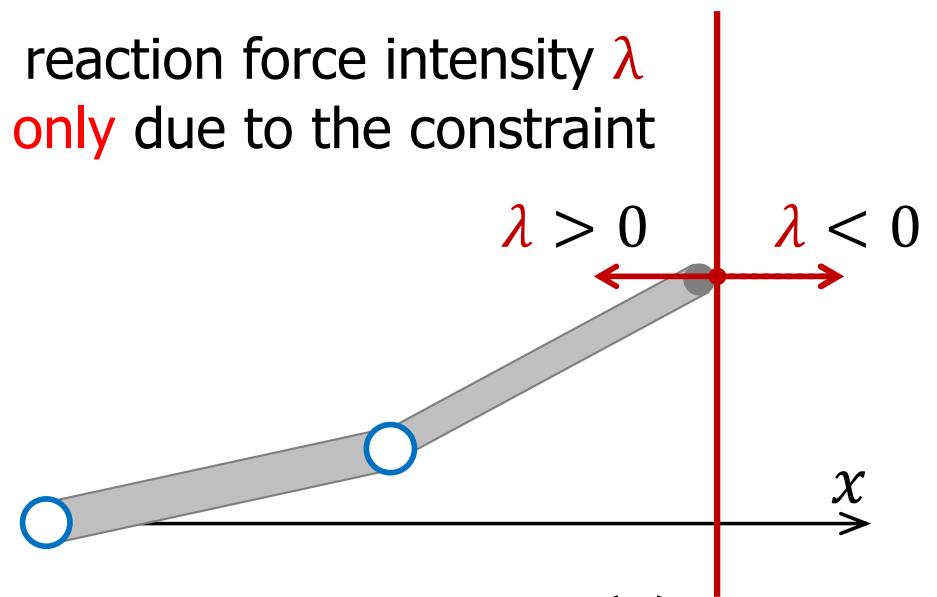
$$p = \begin{pmatrix} p_x(q) \\ p_y(q) \end{pmatrix} = r(q)$$

$$\dot{p} = \frac{\partial r(q)}{\partial q} \dot{q} = J(q) \dot{q}$$

$$\Rightarrow F = \begin{pmatrix} F_x \\ F_y \end{pmatrix}$$

$$M(q)\ddot{q} + \dots = u + A^T(q)\lambda$$

reaction force intensity λ
only due to the constraint



$$p_x(q) = k$$

$$\dot{p}_x = (1 \quad 0) J(q) \dot{q} = A(q) \dot{q} = 0$$

$$A^T(q)\lambda = J^T(q) \begin{pmatrix} 1 \\ 0 \end{pmatrix} \lambda = J^T(q) \begin{pmatrix} -F_x \\ 0 \end{pmatrix}$$

$$\Rightarrow F_x = -\lambda$$



Dynamic model in Cartesian coordinates

assuming
 $n = m$

$$M_r(q)\ddot{r} + S_r(q, \dot{q})\dot{r} + g_r(q) = J_r^{-T}(q)u + F_r$$

with

$$M_r(q) = J_r^{-T}(q)M(q)J_r^{-1}(q) = (J_r(q)M^{-1}(q)J_r^T(q))^{-1}$$

$$S_r(q, \dot{q}) = J_r^{-T}(q)S(q, \dot{q})J_r^{-1}(q) - M_r(q)\dot{J}_r(q)J_r^{-1}(q)$$

$$g_r(q) = J_r^{-T}(q)g(q)$$

... and the usual structural properties

- $M_r > 0$, if J_r is non-singular
- $\dot{M}_r - 2S_r$ is skew-symmetric, if $\dot{M} - 2S$ satisfies the same property
- the Cartesian dynamic model of the robot can be linearly parameterized in terms of a set of dynamic coefficients



Design of the control law

designed in two steps:

1. feedback linearization in the Cartesian space (with force measure)

$$u = J_r^T(q)[M_r(q)a + S_r(q, \dot{q})\dot{r} + g_r(q) - F_r]$$

$$\rightarrow \ddot{r} = a \quad \text{closed-loop system}$$

2. imposition of a dynamic impedance model

most of the times
it is "decoupled"
(diagonal matrices)

$$M_m(\ddot{r} - \ddot{r}_d) + D_m(\dot{r} - \dot{r}_d) + K_m(r - r_d) = F_r$$

*how do I choose these elements?
these matrices?*

desired (apparent) inertia (> 0) desired damping (≥ 0) desired stiffness (> 0) external forces from the environment

is realized by choosing

$$a = \ddot{r}_d + M_m^{-1}[D_m(\dot{r}_d - \dot{r}) + K_m(r_d - r) + F_r]$$

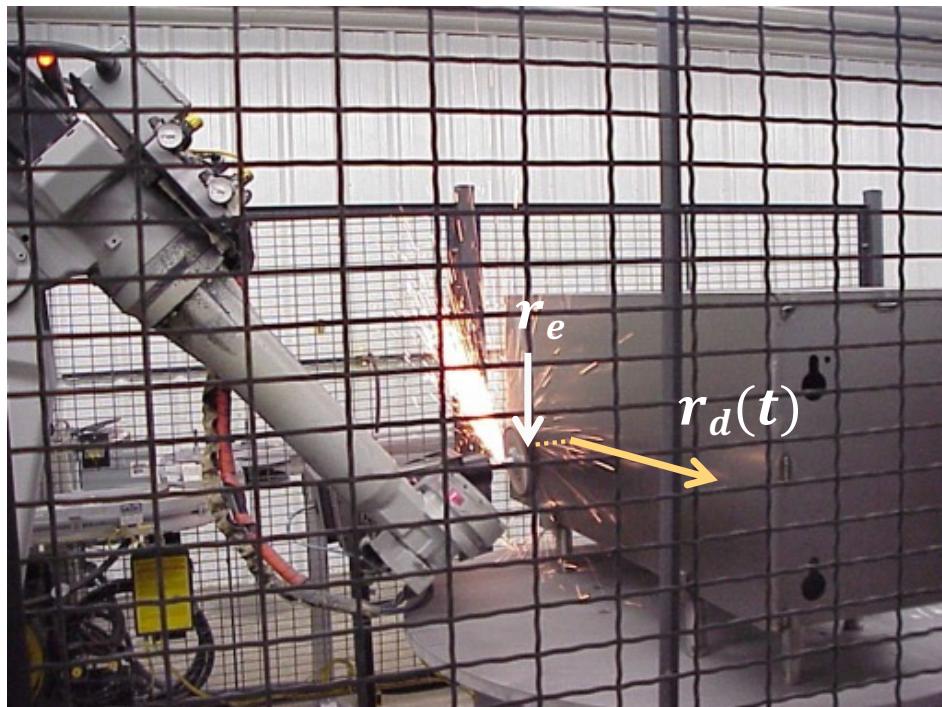
Note: $r_d(t)$ is the desired motion, which typically "slightly penetrates" inside the compliant environment (inducing contact forces)...



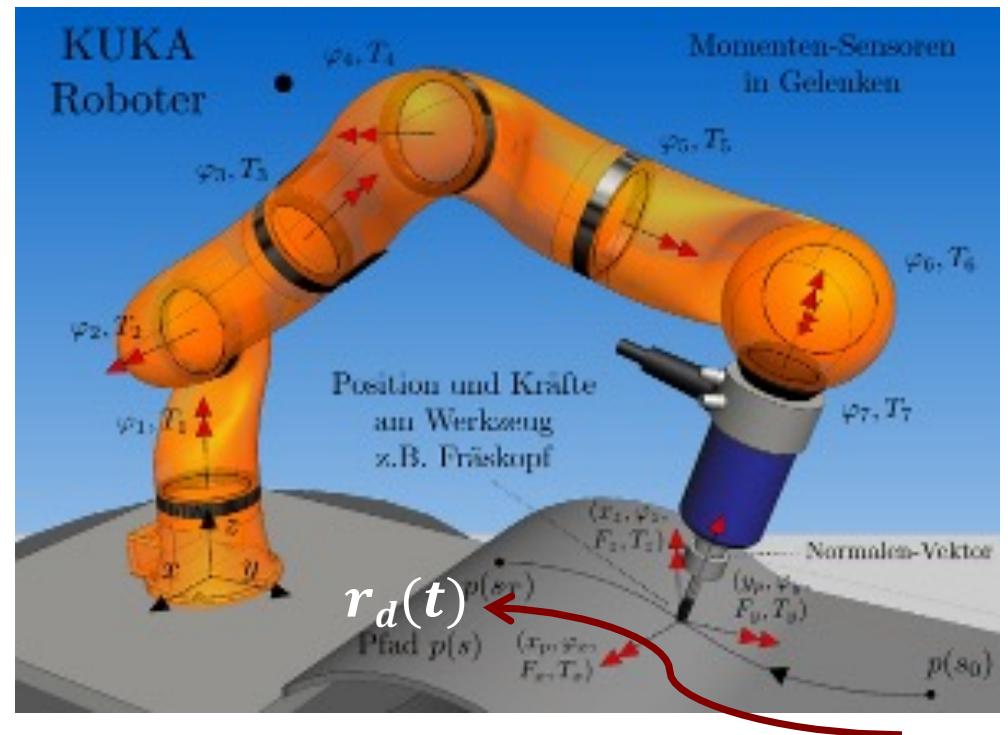
Examples of desired reference r_d in impedance/compliance control

$$M_m(\ddot{r} - \ddot{r}_d) + D_m(\dot{r} - \dot{r}_d) + K_m(r - r_d) = F_r$$

the desired motion $r_d(t)$ is slightly inside
the environment (keeping thus contact)



robot in grinding task



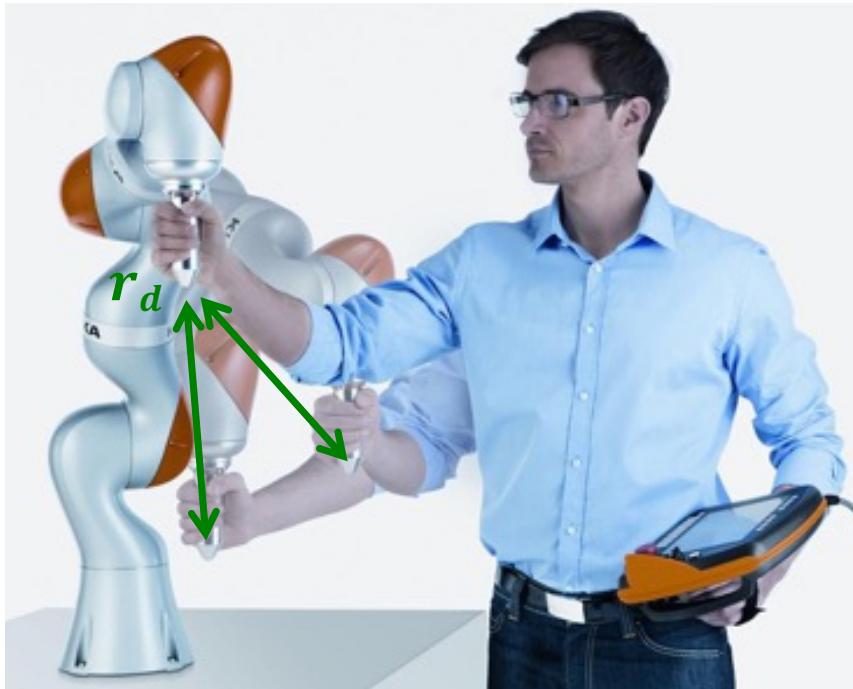
robot writing on a surface



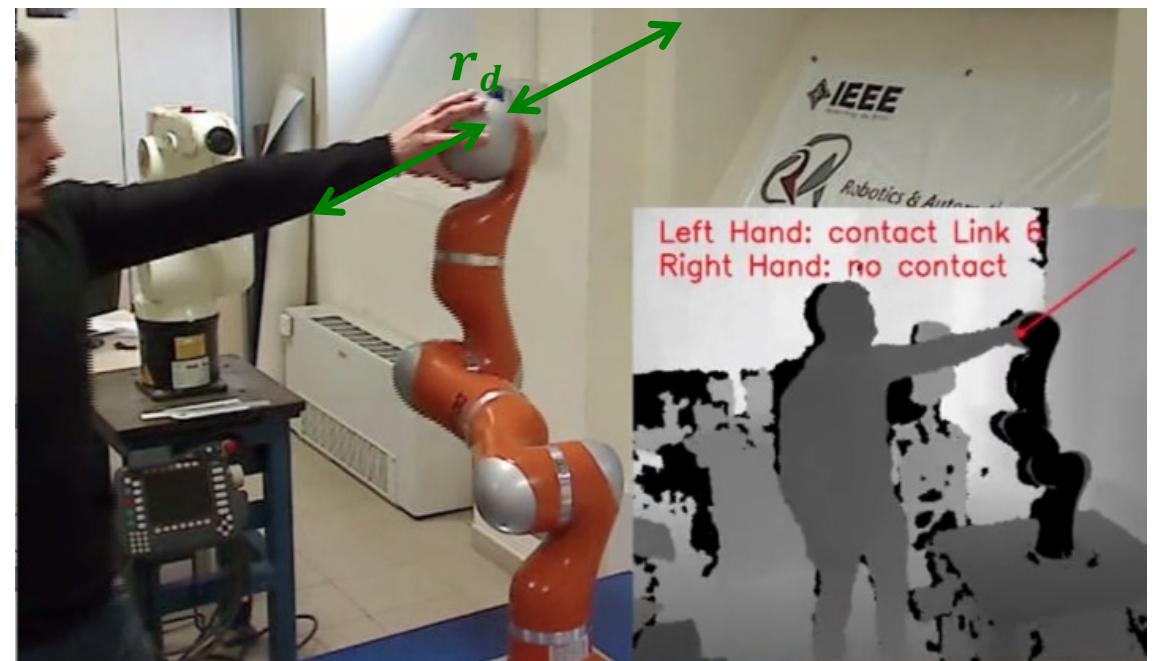
Examples of desired reference x_d in impedance/compliance control

$$M_m(\ddot{r} - \cancel{\dot{r}_d}) + D_m(\dot{r} - \cancel{\dot{r}_d}) + K_m(r - r_d) = F_r$$

constant desired pose r_d is the free Cartesian rest position in a human-robot interaction task



KUKA iiwa robot with human operator



KUKA LWR robot in pHRI (collaboration)



Control law in joint coordinates

substituting and simplifying...

$$u = M(q)J_r^{-1}(q)\{\ddot{r}_d - \dot{J}_r(q)\dot{q} + M_m^{-1}[D_m(\dot{r}_d - \dot{r}) + K_m(r_d - r)]\} \\ + S(q, \dot{q})\dot{q} + g(q) + J_r^T(q)[M_r(q)M_m^{-1} - I]F_r$$

matrix weighting the measured contact forces

- the following identity holds for the term involving contact forces

$$J_r^T(q)[M_r(q)M_m^{-1} - I]F_r = [M(q)J_r^{-1}(q)M_m^{-1} - J_r^T(q)]F_r$$

which eliminates from the control law also the appearance of the last remaining Cartesian quantity (the Cartesian inertia matrix)

- while the control design is based on dynamic analysis and desired (impedance) behavior described in the Cartesian space, the final control implementation is always at the robot joint level



Choice of the impedance model

rationale ...

- adapt/match to the dynamic characteristics of the environment (in particular, of its estimated stiffness) in a complementary way *→ more stiff env.
= more soft robot*
- avoid large impact forces due to uncertain geometric characteristics (position, orientation) of the environment *↳ where we enter in contact?*
- mimic the behavior of a human arm
 - fast and stiff in "free" motion, slow and compliant in "guarded" motion

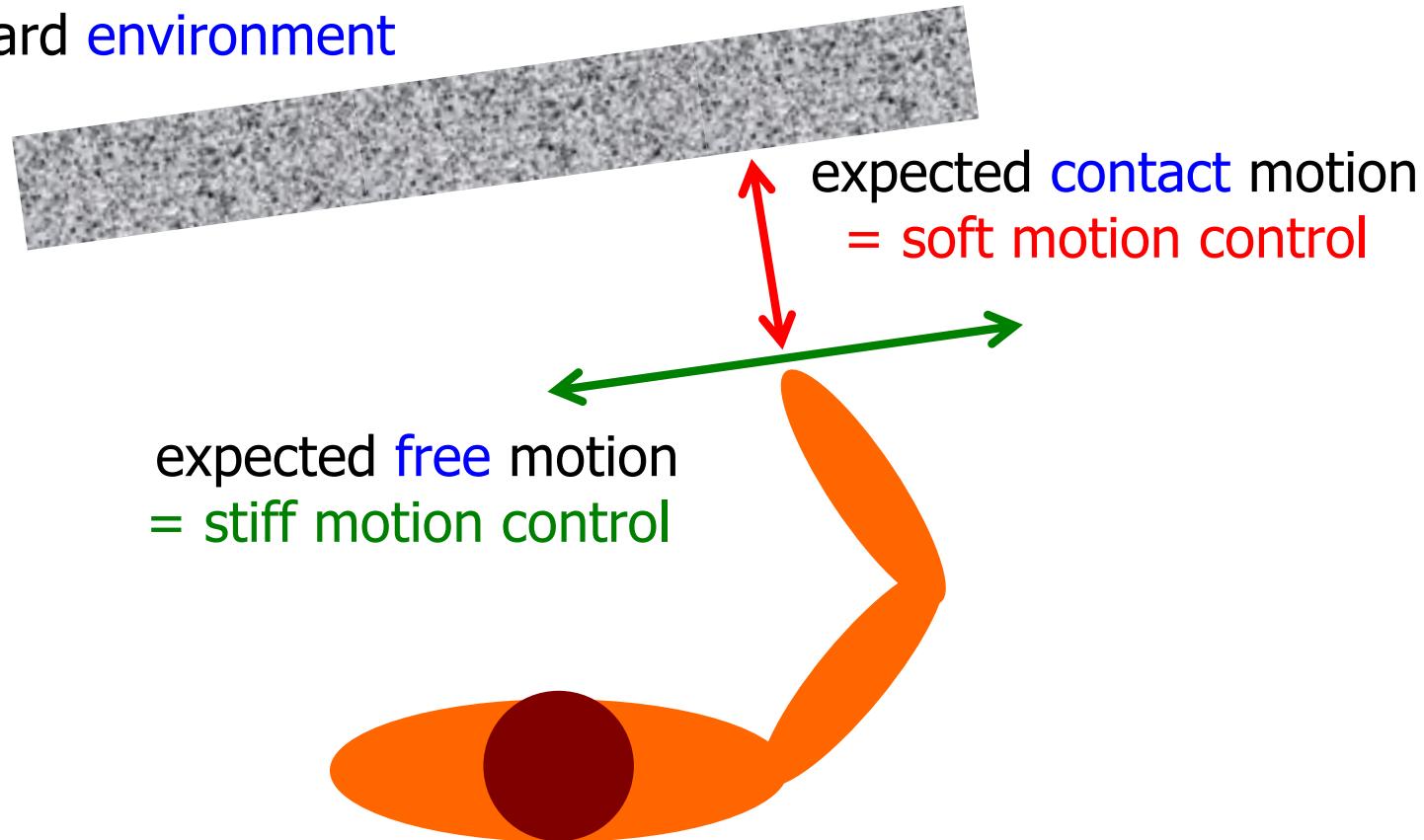


- large $M_{m,i}$ and small $K_{m,i}$ in Cartesian directions where contact is foreseen (→ low contact forces)
- large $K_{m,i}$ and small $M_{m,i}$ in Cartesian directions that are supposed to be free (→ good tracking of desired motion trajectory)
- damping coefficients $D_{m,i}$ are used then to shape transient behaviors



Human arm behavior

hard environment



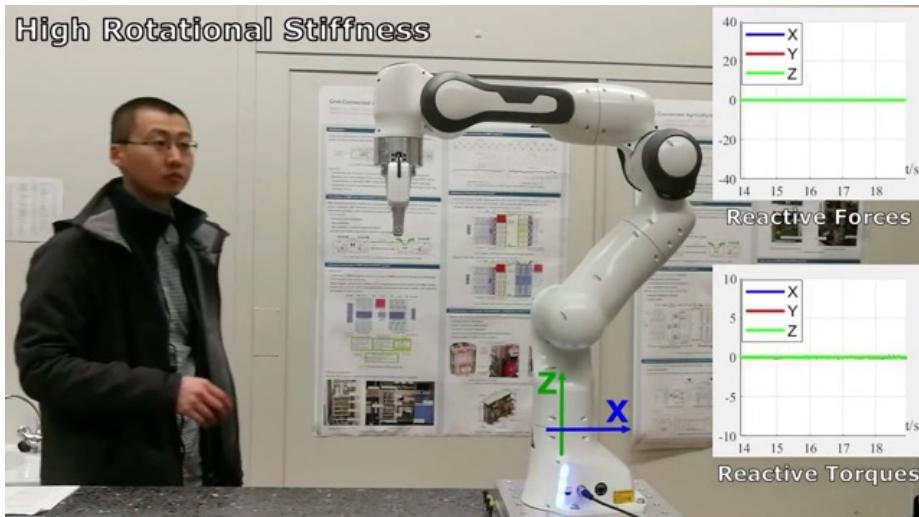
in the selected i -th Cartesian direction:

the **stiffer** is the environment, the **softer** is the chosen model stiffness $K_{m,i}$

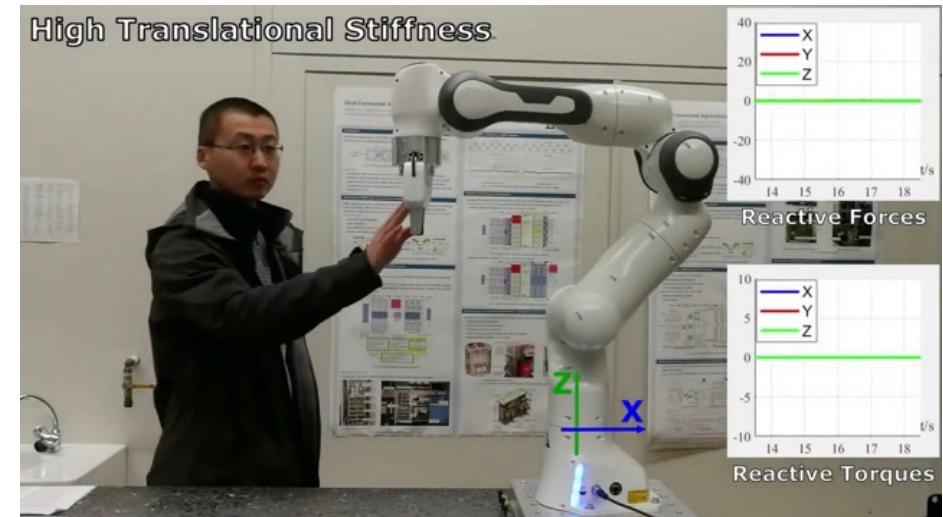
Experiments with impedance control human interaction with a Panda robot



7R Franka Emika Panda robot



high rotational $K_{m,\phi}$
low $K_{m,p}$
(compliant in position)



high translational $K_{m,p}$
low $K_{m,\phi}$
(compliant in orientation)

trajectory tracking with
physical interaction
(uniformly compliant)



LRS – RPTU
Kaiserslautern



A notable simplification - 1

choose the apparent inertia **equal to** the natural Cartesian inertia of the robot

$$M_m = M_r(q) = J_r^{-T}(q)M(q)J_r^{-1}(q) = (J_r(q) M^{-1}(q) J_r^T(q))^{-1}$$

then, the control law becomes

$$\begin{aligned} u = & M(q)J_r^{-1}(q)\{\ddot{r}_d - \dot{J}_r(q)\dot{q}\} + S(q, \dot{q})\dot{q} + g(q) \\ & + J_r^T(q)[D_m(\dot{r}_d - \dot{r}) + K_m(r_d - r)] \end{aligned}$$

WITHOUT contact force feedback! (a F/T sensor is no longer needed...)



this is a **pure motion control** law applied also during interaction,
but designed so as to keep **limited contact forces** at the end-effector level
(as before, K_m is chosen as a function of the **expected** environment stiffness)



A notable simplification - 2

technical issue: if the impedance model (now, nonlinear) is still supposed to represent a real mechanical system, then in correspondence to a desired non-constant inertia ($M_r(q)$) there should be Coriolis and centrifugal terms...



$$M_r(q)(\ddot{r} - \ddot{r}_d) + (S_r(q, \dot{q}) + D_m)(\dot{r} - \dot{r}_d) + K_m(r - r_d) = F_r$$

nonlinear impedance model ("only" gravity terms disappear)

redoing computations, the control law becomes

$$\begin{aligned} u = & M(q)J_r^{-1}(q)\{\ddot{r}_d - J_r(q)J_r^{-1}(q)\dot{r}_d\} + S(q, \dot{q})J_r^{-1}(q)\dot{r}_d + g(q) \\ & + J_r^T(q)[D_m(\dot{r}_d - \dot{r}) + K_m(r_d - r)] \end{aligned}$$

which is indeed slightly more complex, but has the following advantages:

- guarantee of asymptotic convergence to zero tracking error (on $r_d(t)$)
when $F_r = 0$ (no contact situation) \Rightarrow Lyapunov + skew-symmetry of $\dot{M}_r - 2S_r$
- further simplifications when r_d is constant



Cartesian regulation revisited

(without contact, $F_r = 0$)

when r_d is constant ($\dot{r}_d = 0, \ddot{r}_d = 0$), from the previous expression we get the control law

$$u = g(q) + J_r^T(q)[K_m(r_d - r) - D_m\dot{r}] \quad (\star)$$

Cartesian PD control with gravity cancellation...

when $F_r = 0$ (absence of contact), we know already that this control law ensures asymptotic stability of r_d , provided $J_r(q)$ has full rank

proof
(alternative)

Lyapunov candidate $V_1 = \frac{1}{2} \dot{r}^T M_r(q) \dot{r} + \frac{1}{2} (r_d - r)^T K_m (r_d - r)$

$$\dot{V}_1 = \dot{r}^T M_r(q) \ddot{r} + \frac{1}{2} \dot{r}^T \dot{M}_r(q) \dot{r} - \dot{r}^T K_m (r_d - r) = \dots = -\dot{r}^T D_m \dot{r} \leq 0$$

using skew-symmetry of $\dot{M}_r - 2S_r$ and $g_r = J_r^{-T} g$



Cartesian stiffness control (with contact, $F_r \neq 0$)

not sure
positive definite ↗

when $F_r \neq 0$, convergence to r_d is not assured
(it may not even be a closed-loop equilibrium...)

- for analysis, assume an elastic contact model for the environment
$$F_r = K_e(r_e - r) \quad \text{with stiffness } K_e \geq 0 \text{ and rest position } r_e$$
- closed-loop system behavior

Lyapunov candidate

$$V_2 = \frac{1}{2} \dot{r}^T M_r(q) \dot{r} + \frac{1}{2} (r_d - r)^T K_m (r_d - r) + \frac{1}{2} (r_e - r)^T K_e (r_e - r)$$

$$= V_1 + \frac{1}{2} (r_e - r)^T K_e (r_e - r)$$

$$\begin{aligned} \dot{V}_2 &= \dot{r}^T M_r(q) \ddot{r} + \frac{1}{2} \dot{r}^T \dot{M}_r(q) \dot{r} - \dot{r}^T K_m (r_d - r) - \dot{r}^T K_e (r_e - r) \\ &= \dots = -\dot{r}^T D_m \dot{r} + \dot{r}^T (F_r - K_e (r_e - r)) = -\dot{r}^T D_m \dot{r} \leq 0 \end{aligned}$$



Stability analysis (with $F_r \neq 0$)

when $\dot{r} = \ddot{r} = 0$, at a closed-loop system **equilibrium** it is

$$K_m(r_d - r) + K_e(r_e - r) = 0$$

which has the **unique** solution

$$r = (K_m + K_e)^{-1}(K_m r_d + K_e r_e) =: r_E$$

(check that the Lyapunov candidate V_2 has in fact its **minimum** in r_E !)

LaSalle \rightarrow r_E **asymptotically stable equilibrium**

$$r_E \approx \begin{cases} r_e & \text{for } K_e \gg K_m \text{ (rigid environment)} \\ r_d & \text{for } K_m \gg K_e \text{ (rigid controller)} \end{cases}$$



at **steady state**
the contact force is
 $F_r = K_e(r_e - r_E)$

Note: the Cartesian stiffness control law (\star) is often called **compliance control** in the literature

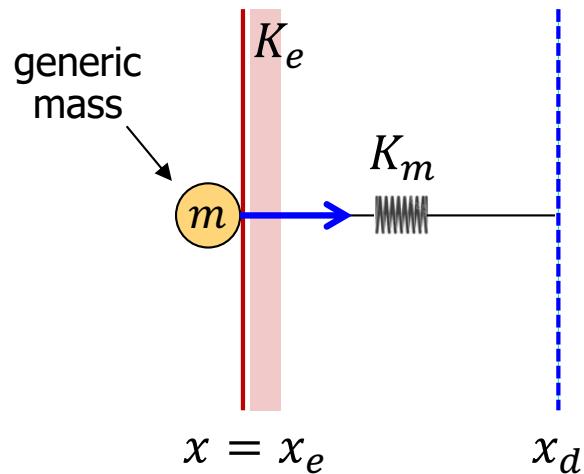


Equilibrium condition

whiteboard...

$$\mathbf{r}_E = (K_m + K_e)^{-1}(K_m r_d + K_e r_e)$$

let $r = x \in \mathbb{R}$

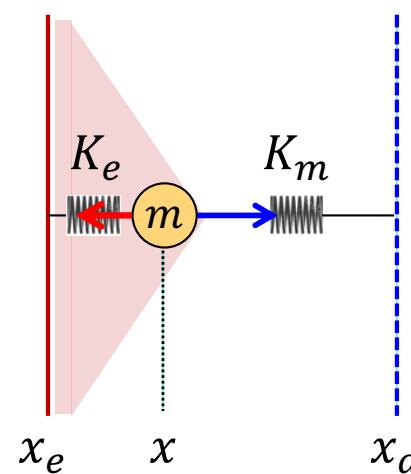


at the initial contact

$$m\ddot{x} = F_c - D_m \dot{x}$$

$$F_c = K_m(x_d - x_e) > 0$$

part of the Cartesian control force



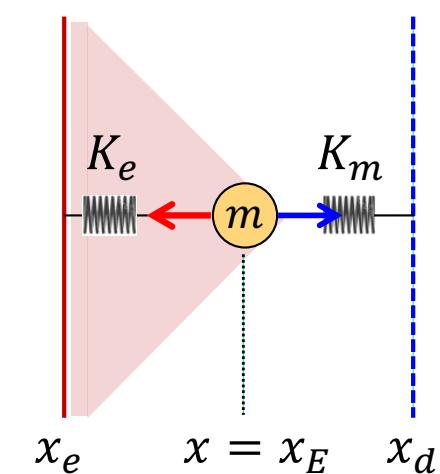
during the transient

$$m\ddot{x} = F_c + F_r - D_m \dot{x}$$

$$F_c = K_m(x_d - x) > 0$$

$$F_r = -K_e(x - x_e) < 0$$

$$|F_c| \neq |F_r|$$



at steady-state (equilibrium)

$$0 = F_c + F_r$$

$$F_c = K_m(x_d - x_E) > 0$$

$$F_r = -K_e(x_E - x_e) < 0$$

$$\rightarrow x_E = \frac{K_m x_d + K_e x_e}{K_m + K_e}$$



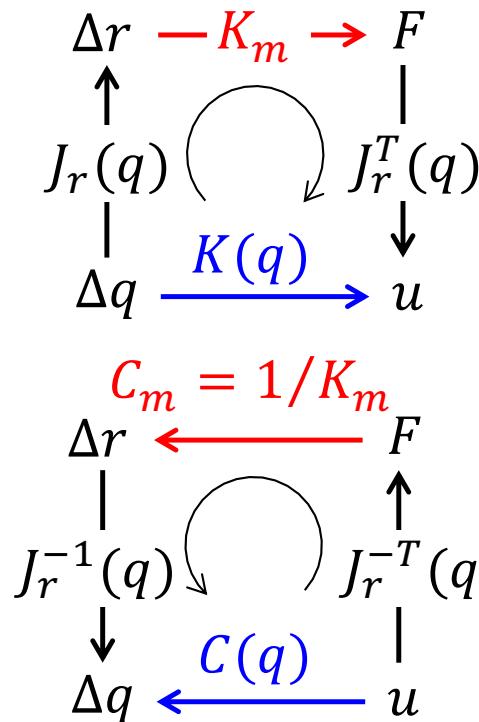
Active equivalent of RCC device

- IF**
- displacements from the desired position r_d are **small**, namely

$$(r_d - r) \approx J_r(q)(q_d - q)$$
 - $g(q) = 0$ (gravity compensated), $D_m = 0$ (or $\dot{r} \approx 0$, i.e., small enough)

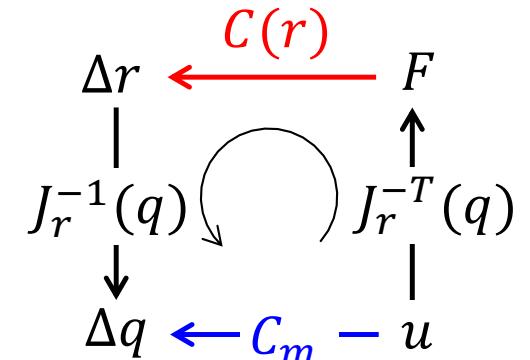
THEN

$$u = J_r^T(q)K_mJ_r(q)(q_d - q) = K(q)(q_d - q)$$



constant Cartesian-level stiffness K_m
(or compliance $C_m = 1/K_m$)
corresponds to
variable joint-level stiffness $K(q)$
(or compliance $= C(q)$)

... and vice versa



this is the “active” counterpart of a
Remote Center of Compliance (RCC) device



Admittance control

- in some cases, we don't have access to low-level robot torque (or motor current) commands \Rightarrow **closed control architecture**
- for handling the interaction with the environment, one uses often **admittance control**: contact forces $F_c \Rightarrow$ velocity commands \dot{q}
- **implementation** (with compliant matrices C)
 - in **joint** space or in **Cartesian (task)** space – with **singularity** issues ...
 - at the **velocity** or **incremental position** level

$$u_c = J^T(q)F_c \rightarrow \dot{q} = C_q u_c \rightarrow \boxed{\dot{q} = C_q J^T(q)F_c} \quad C_q \geq 0$$

\updownarrow
 Δq (to be added to the current q)

$$F_c \rightarrow \dot{r} = C_r F_c \rightarrow \boxed{\dot{q} = J^{-1}(q)C_r F_c} \quad C_r \geq 0$$

\updownarrow
(in case of redundancy) $J^\#(q)$

Experiments with admittance control human interaction with a KUKA LWR robot



7R KUKA LWR4+ robot

handling of task singularities
through **performance constraints**

admittance control at **any contact point**
without using a force/torque sensor

video



ICRA 2016, University of Patras

video



Sep 2013, DIAG Laboratory of Robotics



Robotics 2

Hybrid Force/Motion Control

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Hybrid force/motion control

- we consider **contacts/interactions** between a robot and a stiff environment that **naturally constrains** the end-effector motion
- **compared** to an approach using the constrained/reduced robot dynamics with (bilateral) **geometric constraints**, the **differences** are
 - the hybrid control law is designed in **ideal conditions**, but now unconstrained directions of motion and constrained force directions are defined in a more direct way using a **task frame formalism**
 - all **non-ideal conditions** (compliant surfaces, friction at the contact, errors in contact surface orientation) are handled explicitly in the control scheme by a **geometric filtering of the measured quantities**
 - considering only signal components that should appear in certain directions based on the nominal task model, and treating those that should not be there as **disturbances** to be rejected
- the hybrid control law avoids to introduce conflicting behaviors (force vs. motion control) in any of the task space directions!!

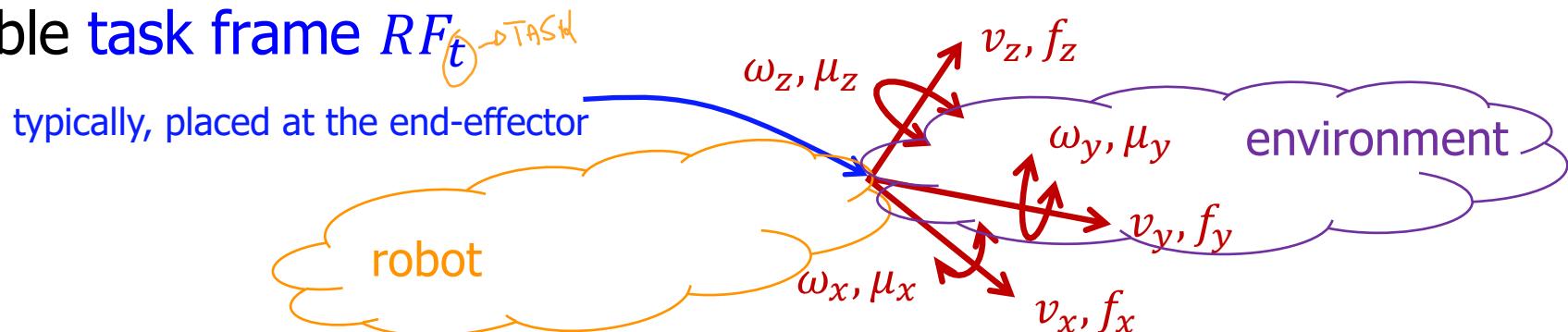


$\text{twist} = \begin{bmatrix} v \\ w \end{bmatrix} \in \mathbb{R}^6$
 $\text{wrench} = \begin{bmatrix} f \\ \mu \end{bmatrix} \in \mathbb{R}^6$

12 components
 only 6 of them are naturally constrained
 Comes from geometry of construction
 ALWAYS 6

Natural constraints

- in **ideal conditions** (robot and environment are perfectly rigid, contact is frictionless), **two sets of generalized directions** can be defined in the **task space**
 - end-effector motion** (v/ω) is prohibited along/around **$6 - k$ directions** (since the environment reacts there with forces/torques)
 - reaction forces/torques** (f/μ) are absent along/around **k directions** (where the environment does not prevent end-effector motions)
- these constraints have been called the **natural constraints** on motion and force associated to the task geometry
- the two sets of directions are characterized through the axes of a suitable **task frame** RF_t

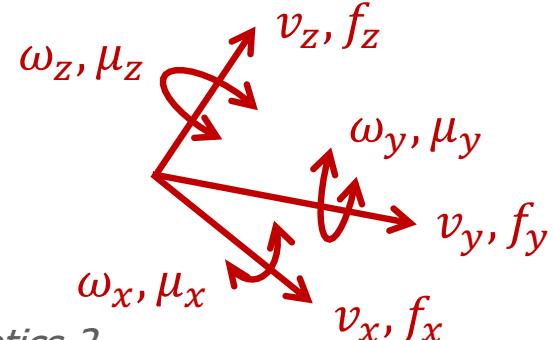




Artificial constraints

Bot twist & wrench
 p imposed by control law
 it does it
 move in source
 (idle and
 torque)
 Orthogonality
 and twists
 and wrenches

- the way **task execution** should be performed can be expressed in terms of so-called **artificial constraints** that specify the desired values (to be imposed by the control law)
 - for the **end-effector velocities** (v/ω) along/around k directions where feasible motions can occur
 - for the **contact forces/torques** (f/μ) along/around $6 - k$ directions where admissible reactions of the environment can occur
- the two sets of directions are **complementary** (they cover the 6D generalized task space) and mutually **orthogonal**, while the **task frame** can be **time-varying** ("moves with task progress")
 - directions are intended as 6D **screws**: twists $V = (v^T \omega^T)^T$ and wrenches $F = (f^T \mu^T)^T$

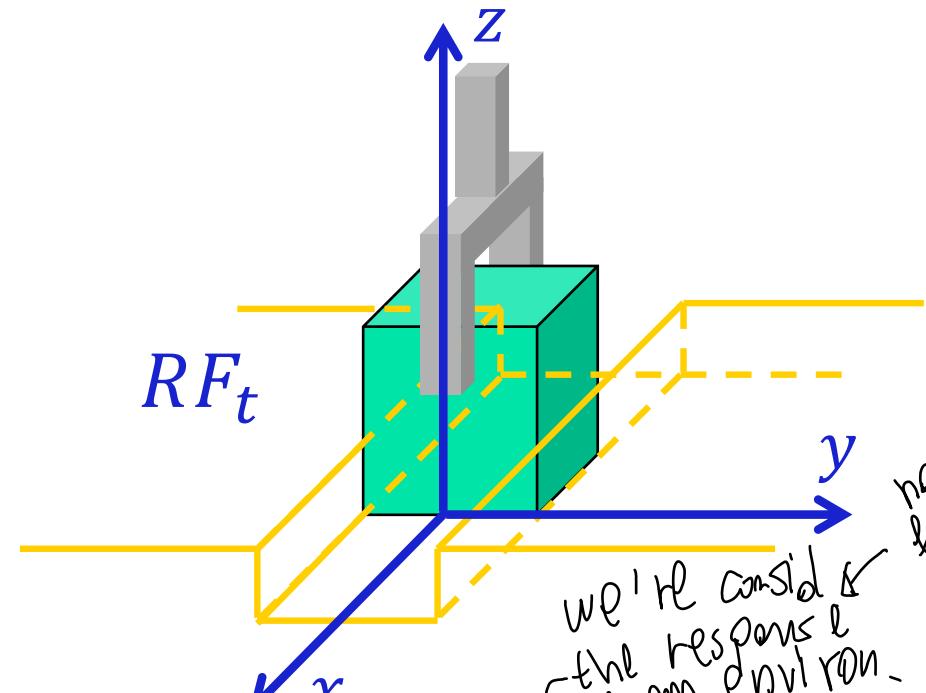


$$F^T V = 0 \Leftrightarrow \text{orthogonality}$$

but **ill-defined** (don't use it!) for $V_1^T V_2$ or $F_1^T F_2$



Task frame and constraints - example 1



v = linear velocity
 ω = angular velocity
 f = force
 μ = torque

$$6 - k = 4 \quad \left\{ \begin{array}{l} \\ \\ \\ \end{array} \right.$$

$$k = 2 \quad \left\{ \begin{array}{l} \\ \\ \\ \end{array} \right.$$

task: slide the cube along a guide

more than 1 constraint

no reaction force environment

natural (geometric) constraints

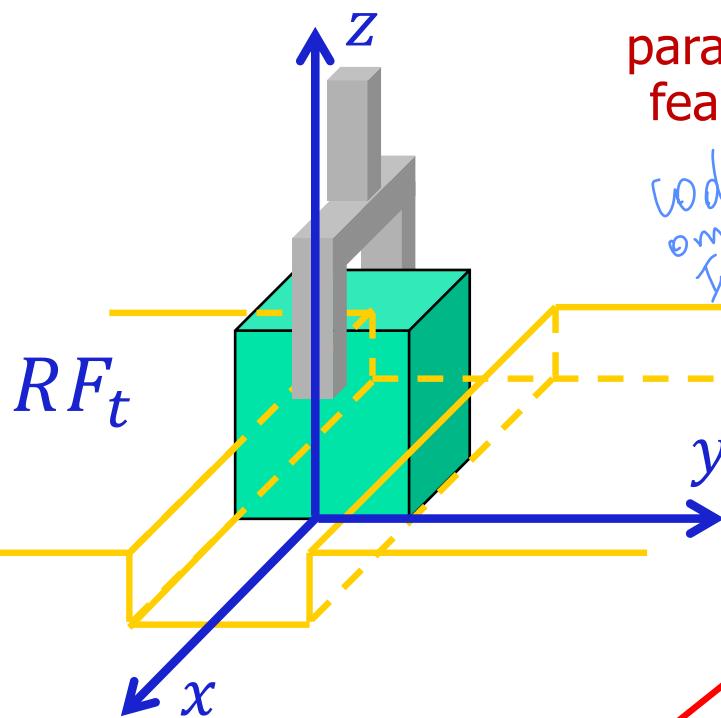
$$\left. \begin{array}{l} v_y = v_z = 0 \\ \omega_x = \omega_z = 0 \\ f_x = \mu_y = 0 \end{array} \right\} \begin{array}{l} 6 - k = 4 \\ k = 2 \end{array}$$

artificial constraints
(to be imposed by the control law)

$$\begin{array}{l} f_y = f_{y,des} (= 0) \text{ (to avoid internal stress)} \\ \mu_x = \mu_{x,des} (= 0), \mu_z = \mu_{z,des} (= 0) \\ f_z = f_{z,des} \text{ (to keep contact)} \\ \omega_y = \omega_{y,des} = 0 \text{ (to slide and not to roll !!)} \\ v_x = v_{x,des} \end{array}$$



Selection of directions - example 1



parametrization of feasible reactions

$$\begin{pmatrix} f \\ \mu \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_y \\ f_z \\ \mu_x \\ \mu_z \end{pmatrix} = Y \begin{pmatrix} f_y \\ f_z \\ \mu_x \\ \mu_z \end{pmatrix}$$

parametrization of feasible motions

*(v)
(ω)*
codification
on wheel
can move

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

here, constant and unitary
("selection" of columns from
the 6×6 identity matrix)

$$T^T Y = 0$$

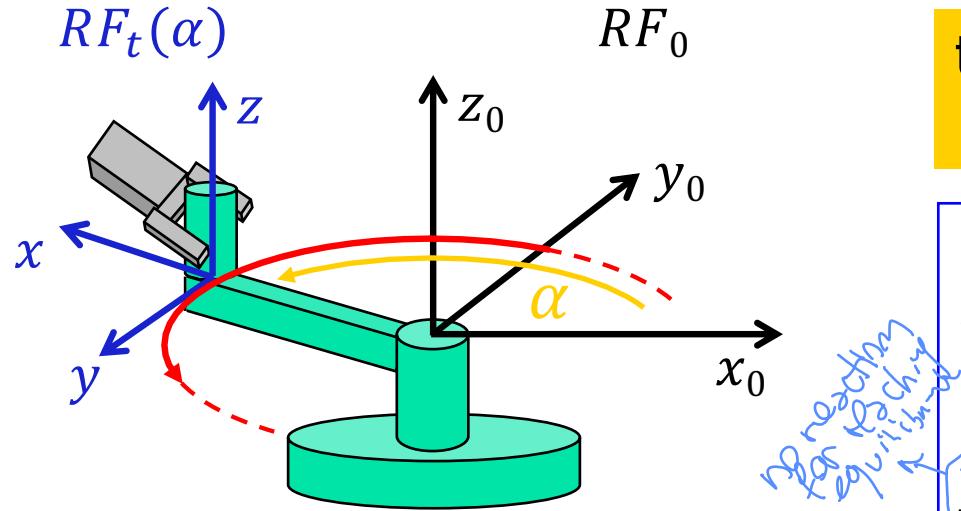
reaction forces/torques
do **not** perform work on
feasible motions

$$(f^T \quad \mu^T) \begin{pmatrix} v \\ \omega \end{pmatrix} = 0$$

In this simple case we
only components
 v_x , ω_y
basis
of feasible
motion



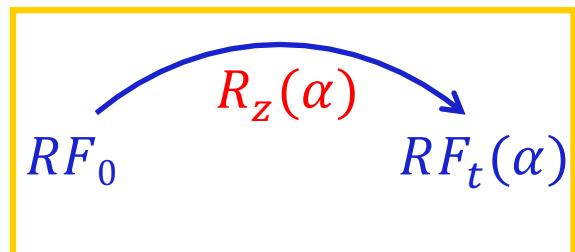
Task frame and constraints - example 2



task: turning a crank
(free handle)

natural constraints

$$\begin{aligned}v_x &= v_z = 0 \\ \omega_x &= \omega_y = 0 \\ f_y &= \mu_z = 0\end{aligned}$$

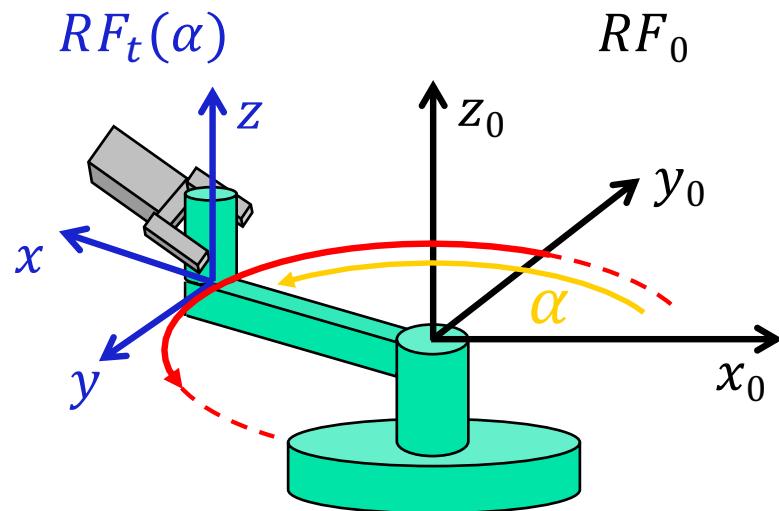


artificial constraints

$$\begin{aligned}f_x &= f_{x,des} (= 0), f_z = f_{z,des} (= 0) \\ \mu_x &= \mu_{x,des} (= 0), \mu_y = \mu_{y,des} (= 0) \\ v_y &= v_{y,des} \text{ (the tangent speed of rotation)} \\ \omega_z &= \omega_{z,des} \text{ (= 0 if handle should not spin)}\end{aligned}$$



Selection of directions – example 2



parametrization of feasible motions

$$\begin{pmatrix} {}^0v \\ {}^0\omega \end{pmatrix} = \begin{pmatrix} R^T(\alpha) & 0 \\ 0 & R^T(\alpha) \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v_y \\ \omega_z \end{pmatrix}$$

always and more when we wish using rotations
with respect to RF_0

$$T^T(\alpha)Y(\alpha) = 0$$

parametrization of feasible reactions

$$\begin{pmatrix} {}^0f \\ {}^0\mu \end{pmatrix} = \begin{pmatrix} R^T(\alpha) & 0 \\ 0 & R^T(\alpha) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} f_x \\ f_z \\ \mu_x \\ \mu_y \end{pmatrix} = Y(\alpha) \begin{pmatrix} f_x \\ f_z \\ \mu_x \\ \mu_y \end{pmatrix}$$

L change over time

$$\sqrt{y} = 1 \text{ m/s}$$

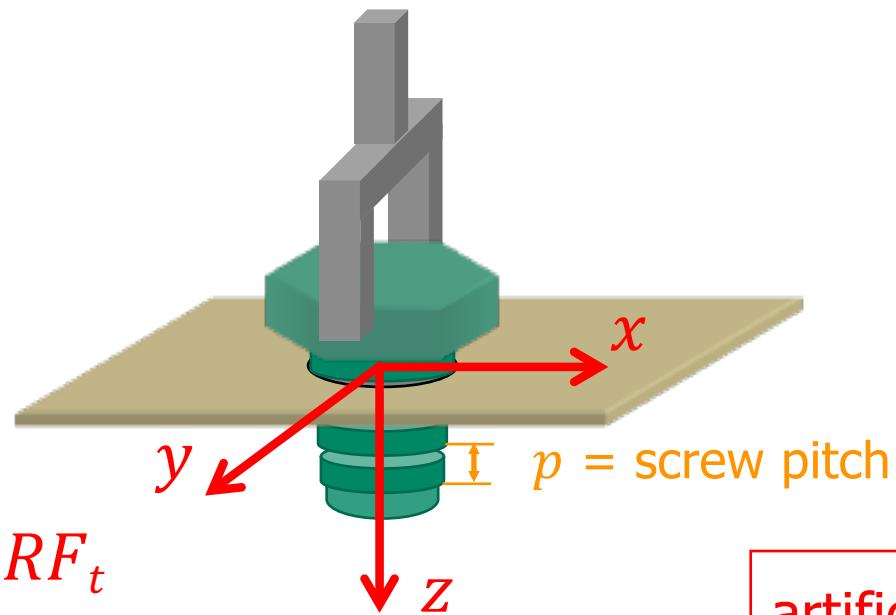
$$x(t) = 1/2 t$$

$$f(0) = 0$$

L = length of the crank



Task frame and constraints - example 3



the screw proceeds **along** and **around** the **z-axis**, but **not** in an **independent** way! (1 dof)

accordingly, f_z and μ_z **cannot** be **independent**

task: insert a screw in a bolt

natural constraints (partial...)

$$v_x = v_y = 0$$

$$\omega_x = \omega_y = 0$$

artificial constraints (abundant...)

$$f_x = f_{x,des} = 0, f_y = f_{y,des} = 0$$

$$\mu_x = \mu_{x,des} = 0, \mu_y = \mu_{y,des} = 0$$

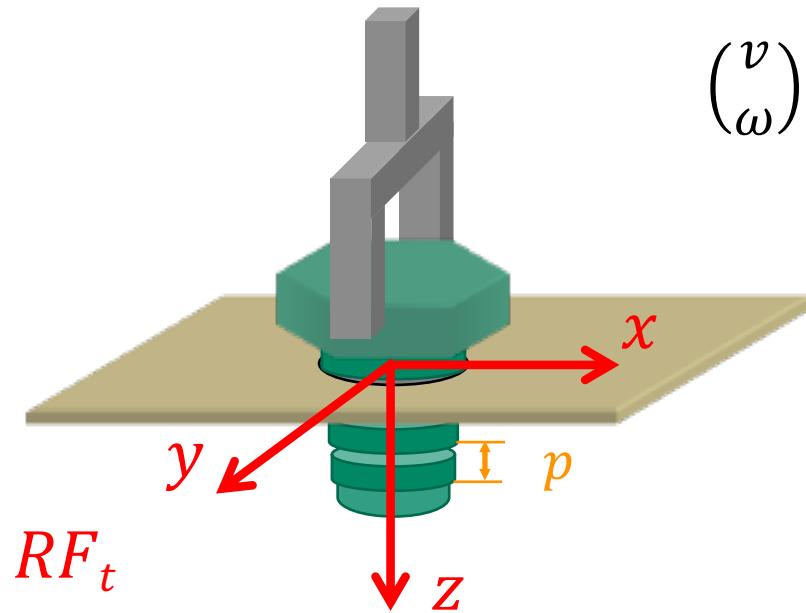
$$v_z = v_{z,des}, \omega_z = \omega_{z,des} = (2\pi/p)v_{z,des}$$

$$f_z = f_{z,des}, \mu_z = \mu_{z,des} \text{ (one function of the other!)}$$

wrench (force/torque) direction should be **orthogonal** to motion twist!



Selection of directions – example 3



the columns of \mathbf{T} and \mathbf{Y}
do not necessarily coincide
with selected columns
of the 6×6 identity matrix
 \Rightarrow generalized (screw)
directions

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & \frac{2\pi}{p} \end{pmatrix}^T v_z = T v_z \quad (k = 1)$$

or $\omega_z = 2\pi \frac{v_z}{p}$

\mathbf{Y} : such that $T^T \mathbf{Y} = 0$



$$f_z = -\frac{2\pi}{p} \mu_z$$

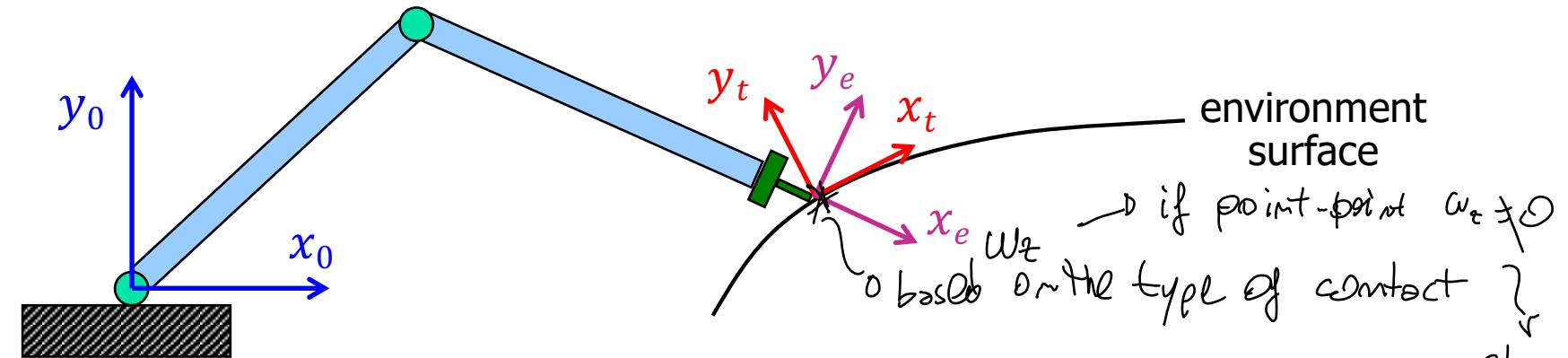
$(6 - k = 5)$

$$\begin{pmatrix} f \\ \mu \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2\pi/p \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ \mu_x \\ \mu_y \\ \mu_z \end{pmatrix} = \mathbf{Y} \begin{pmatrix} f_x \\ f_y \\ \mu_x \\ \mu_y \\ \mu_z \end{pmatrix}$$



Frames of interest – example 4

planar motion of a 2R robot ($n = 2$) in pointwise contact with a surface (task dimension $m = 2$)

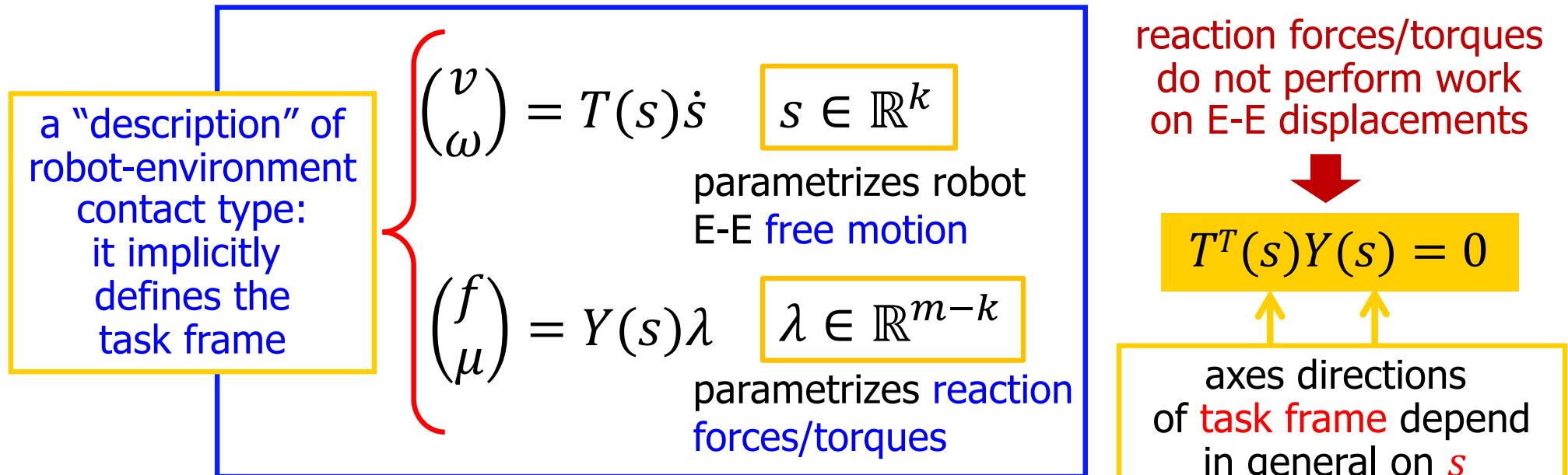


- task frame RF_t used for an independent definition of the hybrid reference values (here: ${}^t v_{x,des}$ [$k = 1$] and ${}^t f_{y,des}$ [$m - k = 1$]) and for computing the errors that drive the feedback control law
- sensor frame RF_e (here: RF_2) where the force ${}^e f = ({}^e f_x, {}^e f_y)$ is measured
- base frame RF_0 in which the end-effector velocity is expressed (here: ${}^0 v = ({}^0 v_x, {}^0 v_y)$ of O_2), computed using robot Jacobian and joint velocities

all quantities (and errors!) should be expressed ("rotated") in the same reference frame \Rightarrow the task frame!



General parametrization of hybrid tasks



in general, it is $m = 6$
(as in most of the previous examples) +

robot dynamics

$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = u + J^T(q) \begin{pmatrix} f \\ \mu \end{pmatrix}$$

robot kinematics

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = J(q)\dot{q}$$

in free motion T is identity
 Y vanishes
in no motion T vanishes, Y identity



Hybrid force/velocity control

- **control objective:** to impose desired task evolutions to the parameters s of **motion** and to the parameters λ of **force**

$$s \rightarrow s_d(t) \quad \lambda \rightarrow \lambda_d(t)$$

- the control law is designed in **two steps**

1. exact **linearization and decoupling** in the **task frame** by feedback

$$\begin{array}{c} \text{closed-loop} \\ \text{model} \end{array} \rightarrow \begin{pmatrix} \ddot{s} \\ \ddot{\lambda} \end{pmatrix} = \begin{pmatrix} a_s \\ a_\lambda \end{pmatrix}$$

2. (**linear**) design of a_s and a_λ so as to impose the desired dynamic behavior to the errors $e_s = s_d - s$ and $e_\lambda = \lambda_d - \lambda$

- **assumptions:** $n = m$ (= 6 usually), $J(q)$ out of singularity

Note: in “simple” cases, \dot{s} and $\dot{\lambda}$ drive single components of v or ω and of f or μ ; accordingly, T and Y are just columns of 0/1 selection matrices



Feedback linearization in task space

Joint robot state

$$J(q)\dot{q} = \begin{pmatrix} v \\ \omega \end{pmatrix} = T(s)\dot{s} \xrightarrow{\text{INTEGRATION}} J\ddot{q} + \dot{J}\dot{q} = T\ddot{s} + \dot{T}\dot{s}$$

$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = u + J^T(q) \begin{pmatrix} f \\ \mu \end{pmatrix} = u + J^T(q)Y(s)\lambda$$

$$(M(q)J^{-1}(q)T(s) : -J^T(q)Y(s)) \begin{pmatrix} \ddot{s} \\ \lambda \end{pmatrix} + M(q)J^{-1}(q)(\dot{T}(s)\dot{s} - \dot{J}(q)\dot{q}) + S(q, \dot{q})\dot{q} + g(q) = u$$

nonsingular $n \times n$ matrix
(under the assumptions made)

(Components to which I want to assign values)

$$u = (MJ^{-1}T : -J^TY) \begin{pmatrix} a_s \\ a_\lambda \end{pmatrix} + MJ^{-1}(\dot{T}\dot{s} - \dot{J}\dot{q}) + S\dot{q} + g$$

linearizing and decoupling control law

\ddot{s} has relative degree k

$$\begin{pmatrix} \ddot{s} \\ \lambda \end{pmatrix} = \begin{pmatrix} a_s \\ a_\lambda \end{pmatrix}$$

s has "relative degree" = 2
 λ has "relative degree" = 0



Stabilization with a_s and a_λ

as usual, it is sufficient to apply linear control techniques for the exponential stabilization of tracking errors (on each single, input-output decoupled channel)

$$a_s = \ddot{s}_d + K_D(\dot{s}_d - \dot{s}) + K_P(s_d - s)$$

$K_P, K_D > 0$
and diagonal

$$\ddot{e}_s + K_D \dot{e}_s + K_P e_s = 0 \rightarrow e_s = s_d - s \rightarrow 0$$

$K_I \geq 0$
diagonal

$$a_\lambda = \lambda_d + K_I \int (\lambda_d - \lambda) dt$$

$a_\lambda = \lambda_d$ would be enough,
but adding an integral
with the **force error**
gives more robustness
to (constant) disturbances

$$e_\lambda + K_I \int e_\lambda dt = 0 \rightarrow e_\lambda = \lambda_d - \lambda \rightarrow 0$$

we need "values" for s , \dot{s} and λ to be
extracted from actual measurements !



“Filtering” position and force measures

- s and \dot{s} are obtained from measures of q and \dot{q} , equating the descriptions of the end-effector pose and velocity “from the robot side” (direct and differential kinematics) and “from the environment side” (function of s, \dot{s})

example

$${}^0r = {}^0f(q) = \begin{pmatrix} L \cos s \\ L \sin s \\ 0 \end{pmatrix} \rightarrow s = \text{atan2}\{{}^0f_y(q), {}^0f_x(q)\}$$

$$J(q)\dot{q} = T(s)\dot{s} \rightarrow \dot{s} = T^\#(s)J(q)\dot{q}$$

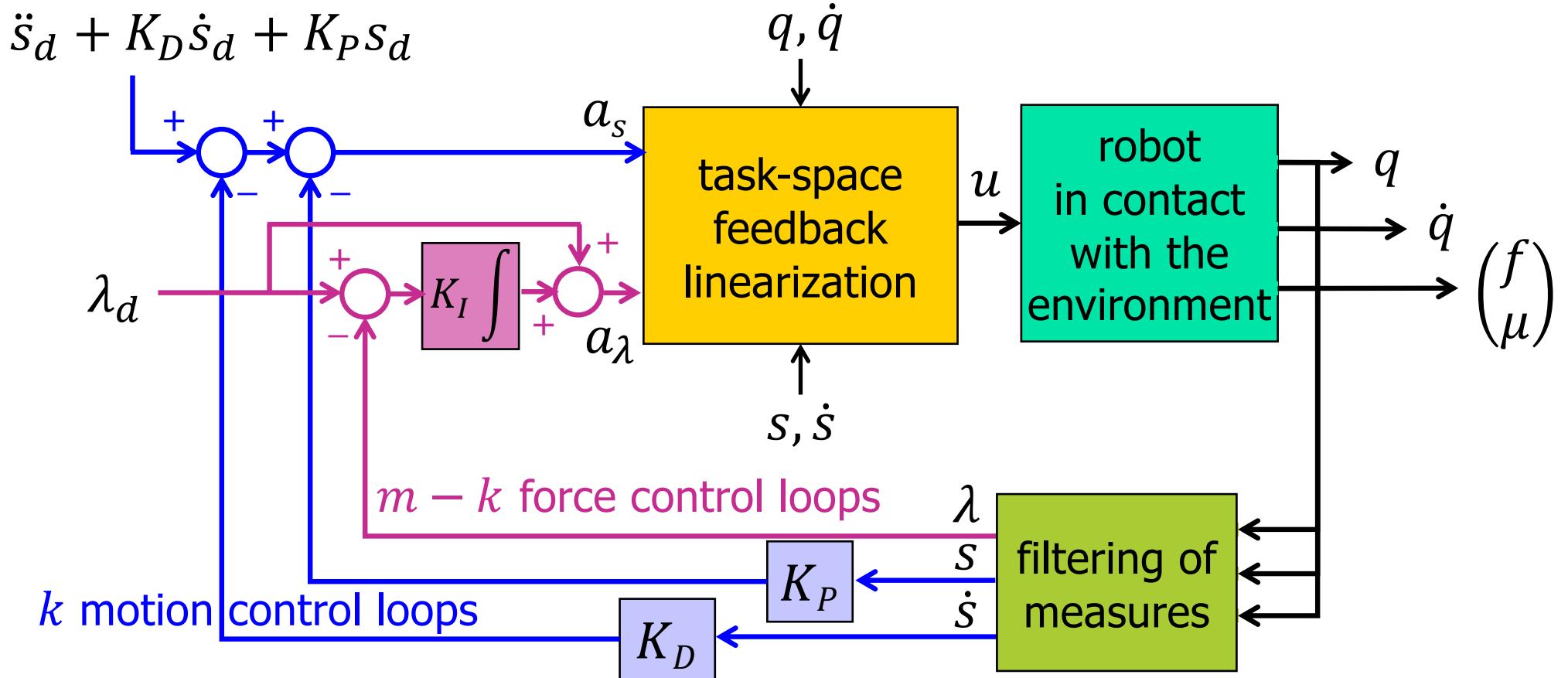
- λ is obtained from force/torque measures at end-effector

$$\begin{pmatrix} f \\ m \end{pmatrix} = Y(s)\lambda \rightarrow \lambda = Y^\#(s) \begin{pmatrix} f \\ m \end{pmatrix}$$

pseudoinverses
of “tall” matrices
having full
column rank, e.g.,
 $T^\# = (T^T T)^{-1} T^T$
(or weighted)



Block diagram of hybrid control



usually $m = 6$ (complete 3D space)

limit cases $k = m$: no force control loops, only motion (free motion)

$k = 0$: no motion control loops, only force ("frozen" robot end-effector)



Block diagram of hybrid control

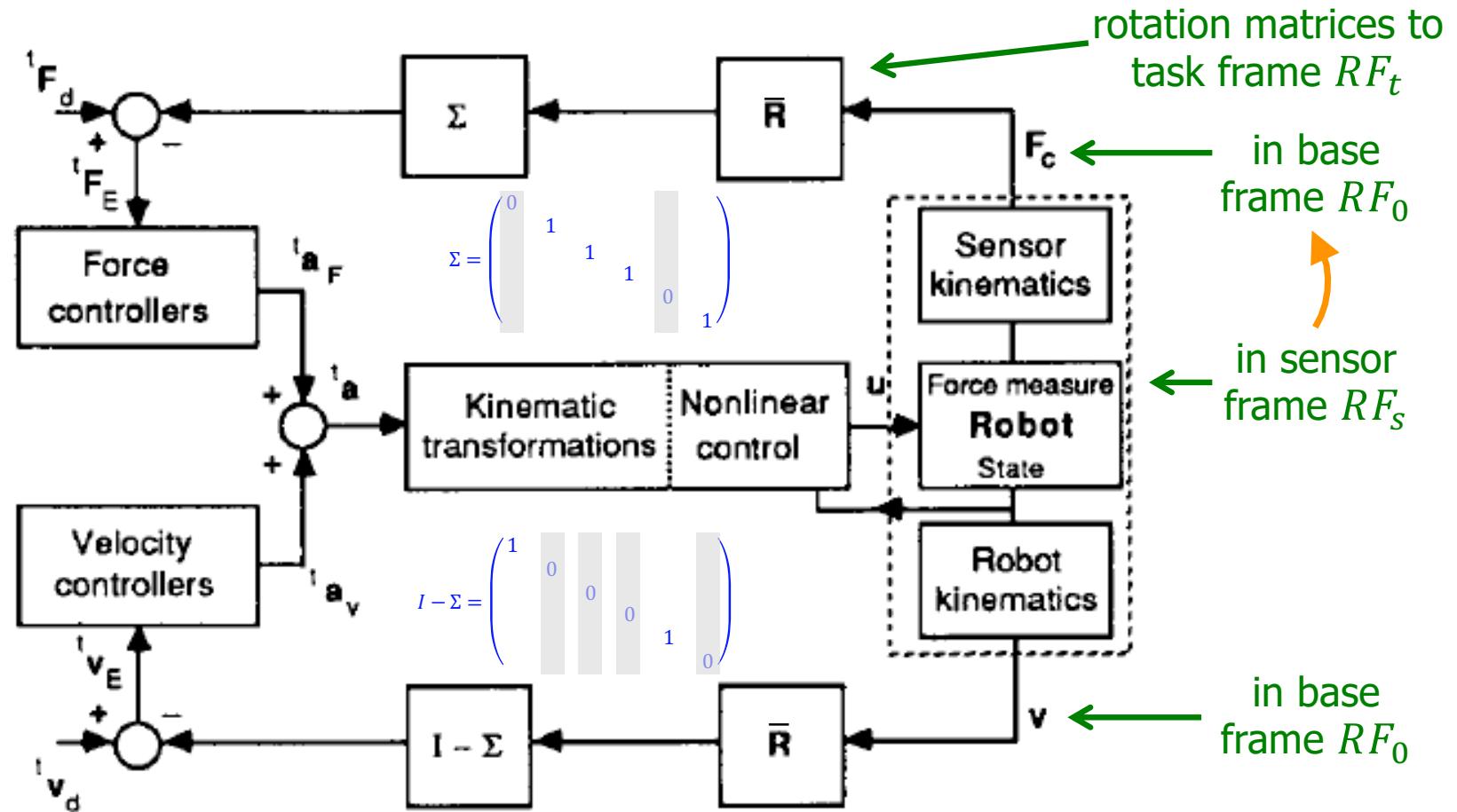
simpler case of 0/1 selection matrices

compact notation
in this slide

$$F = \begin{pmatrix} f \\ \mu \end{pmatrix}$$

$$V = \begin{pmatrix} v \\ \omega \end{pmatrix}$$

$$\bar{R} = \begin{pmatrix} R & 0 \\ 0 & R \end{pmatrix}$$



λ and \dot{s} are just single components of f (or μ) and v (or ω)

Y and T are replaced by 0/1 selection matrices: Σ and $I - \Sigma$



Force control via an impedance model

- in a force-controlled direction of the hybrid task space, when the **contact stiffness is limited** (i.e., far from infinite, as assumed in the ideal case), one may use **impedance model ideas** to explicitly **control the contact force**
 - let x be the position of the robot along such a direction, x_d the (constant) contact point, $k_s > 0$ the contact (viz., sensor) stiffness, and $f_d > 0$ the desired contact force
- the impedance model is chosen then as

$$m_m \ddot{x} + d_m \dot{x} + k_s(x - x_d) = f_d$$

where the **force sensor** measures $f_s = k_s(x - x_d)$, and only $m_m > 0$ and $d_m > 0$ are free model parameters

- after feedback linearization ($\ddot{x} = a_x$), the command a_x is designed as
$$a_x = (1/m_m)[(f_d - f_s) - d_m \dot{x}] \leftarrow^{\text{in free space}} f_s = 0$$
which is a **P-regulator** of the desired force, **with velocity damping**
- the **same** control law works also before the contact ($f_s = 0$), guaranteeing a steady-state speed $\dot{x}_{ss} = f_d/d_m > 0$ in the **approaching phase**



First experiments with hybrid control

First Experiments with Hybrid Force/Velocity Control

Università di Roma "La Sapienza"
DIS, LabRob
February 1991



video

First Experiments with Hybrid Force/Velocity Control

(part II)

Università di Roma "La Sapienza"
DIS, LabRob
February 1991



video

MIMO-CRF robot
(DIS, Laboratorio di Robotica, 1991)



Sources of inconsistency in force and velocity measurements

1. presence of **friction** at the contact → *wl must apply more force than free motion*
 - a reaction force component appears that opposes motion in a “free” motion direction (in case of Coulomb friction, the tangent force intensity depends also on the applied normal force ...)
2. **compliance** in the robot structure and/or at the contact
 - a (small) displacement may be present also along directions that are nominally “constrained” by the environment
3. uncertainty on **environment geometry** at the contact
 - can be reduced/eliminated by real-time **estimation processes** driven by external sensors (e.g., vision –but also force!) *so if we can believe on our measurements*

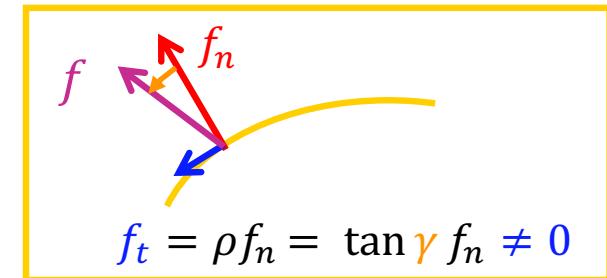


Estimation of an unknown surface

how difficult is to **estimate** the unknown profile of the environment surface, using information from velocity and force measurements at the contact?

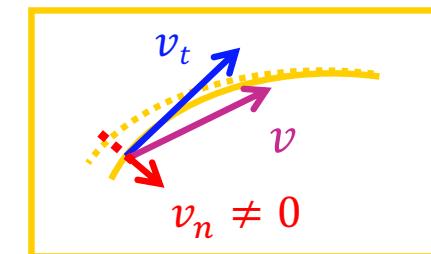
1. normal = nominal direction of measured **force**

... in the presence of contact motion with friction, the **measured** force f is slightly rotated from the actual normal by an (unknown) angle γ



2. tangent = nominal direction of measured **velocity**

... compliance in the robot structure (joints) and/or at the contact may lead to a **computed** velocity v having a small component along the actual normal to the surface



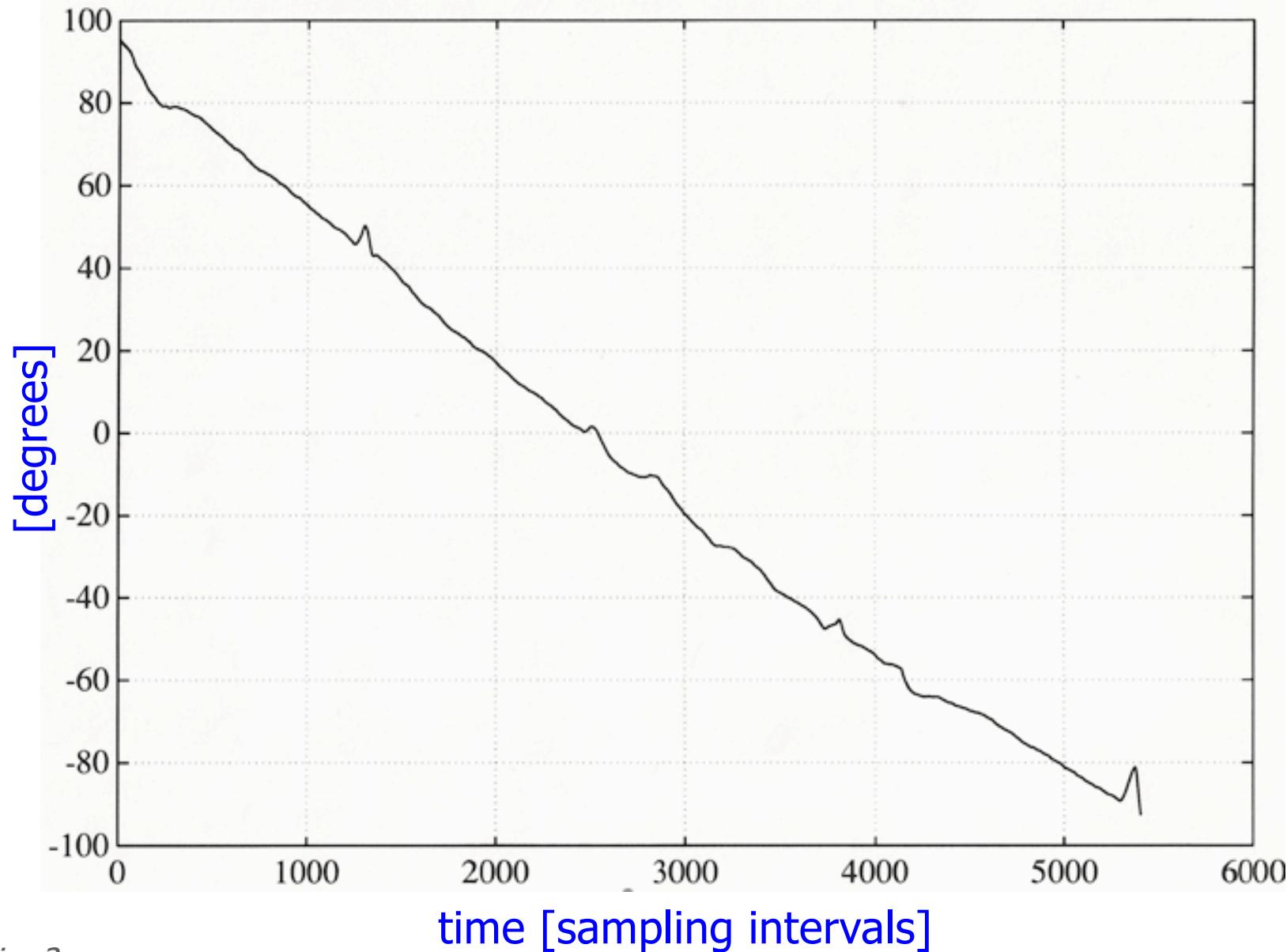
3. mixed method (sensor fusion) with RLS

- a. tangent direction is estimated by a **recursive least squares** method from position measurements
- b. friction angle is estimated by a **recursive least squares** method, using the current estimate of the tangent direction and from force measurements

of position interpolate them
to approach an unknown surface or to recover contact (in case of loss), the robot uses simple exploratory moves



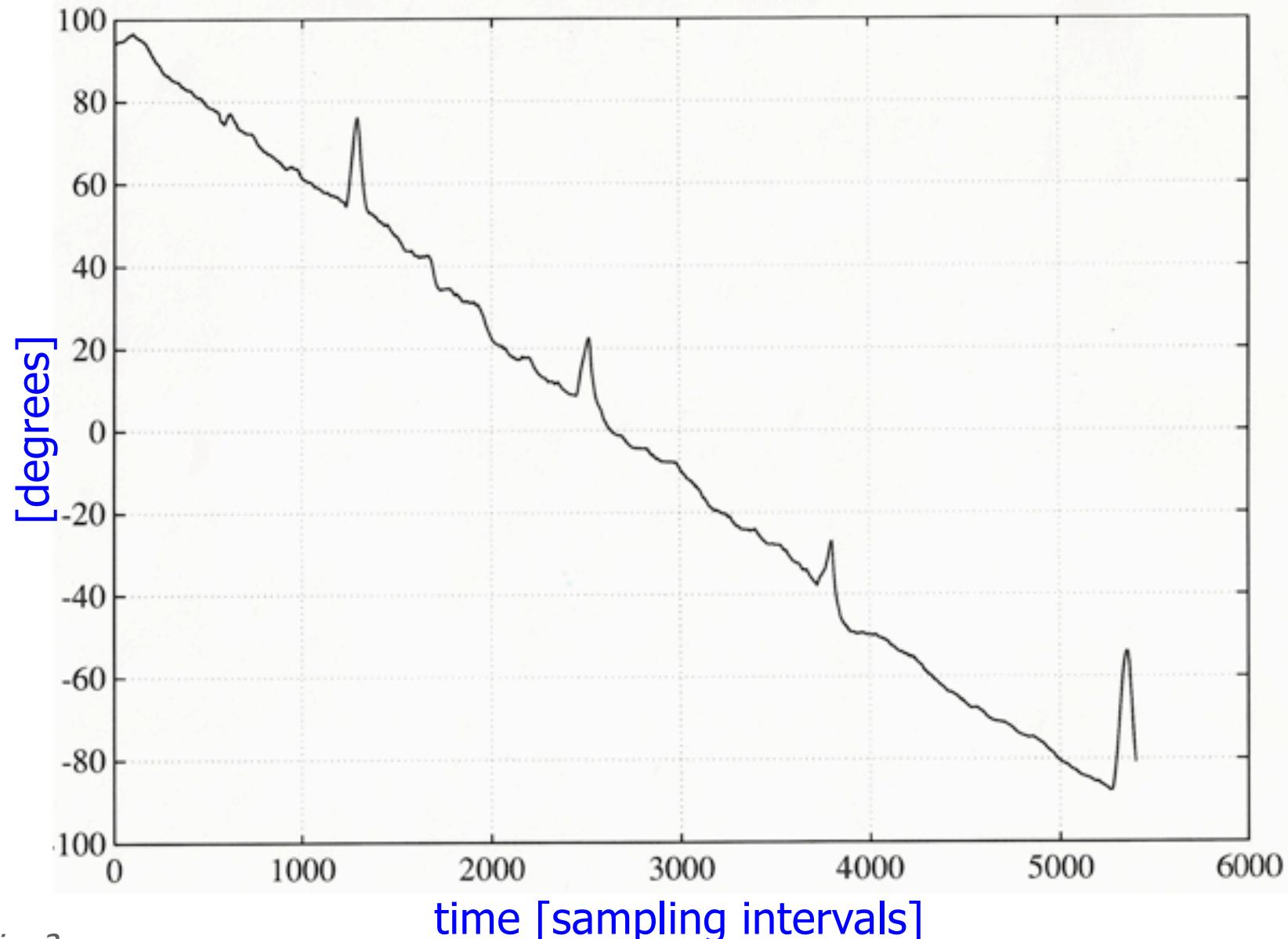
Position-based estimation of the tangent (for a circular surface traced at constant speed)





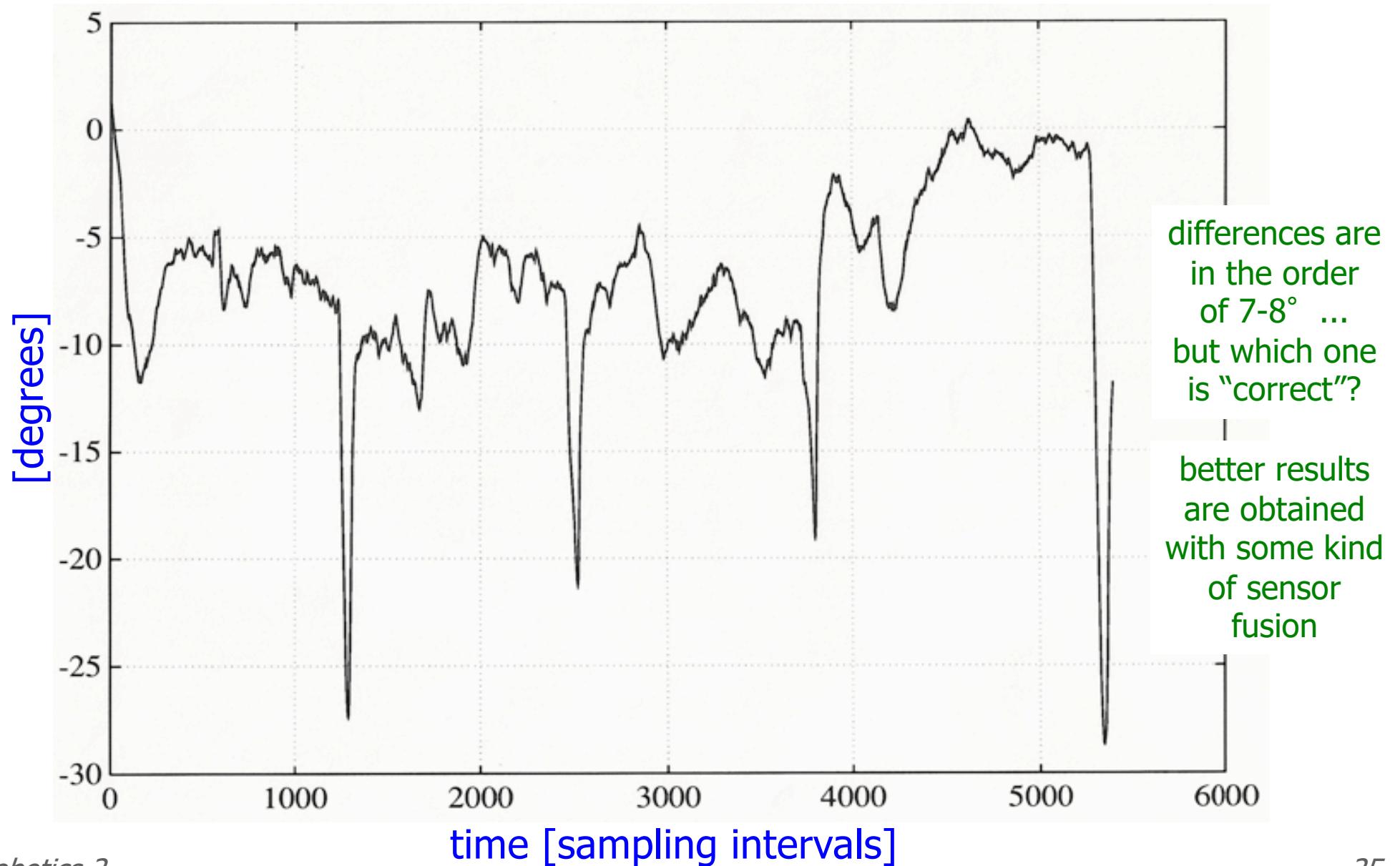
Force-based estimation of the tangent

(for the same circular surface traced at constant speed)





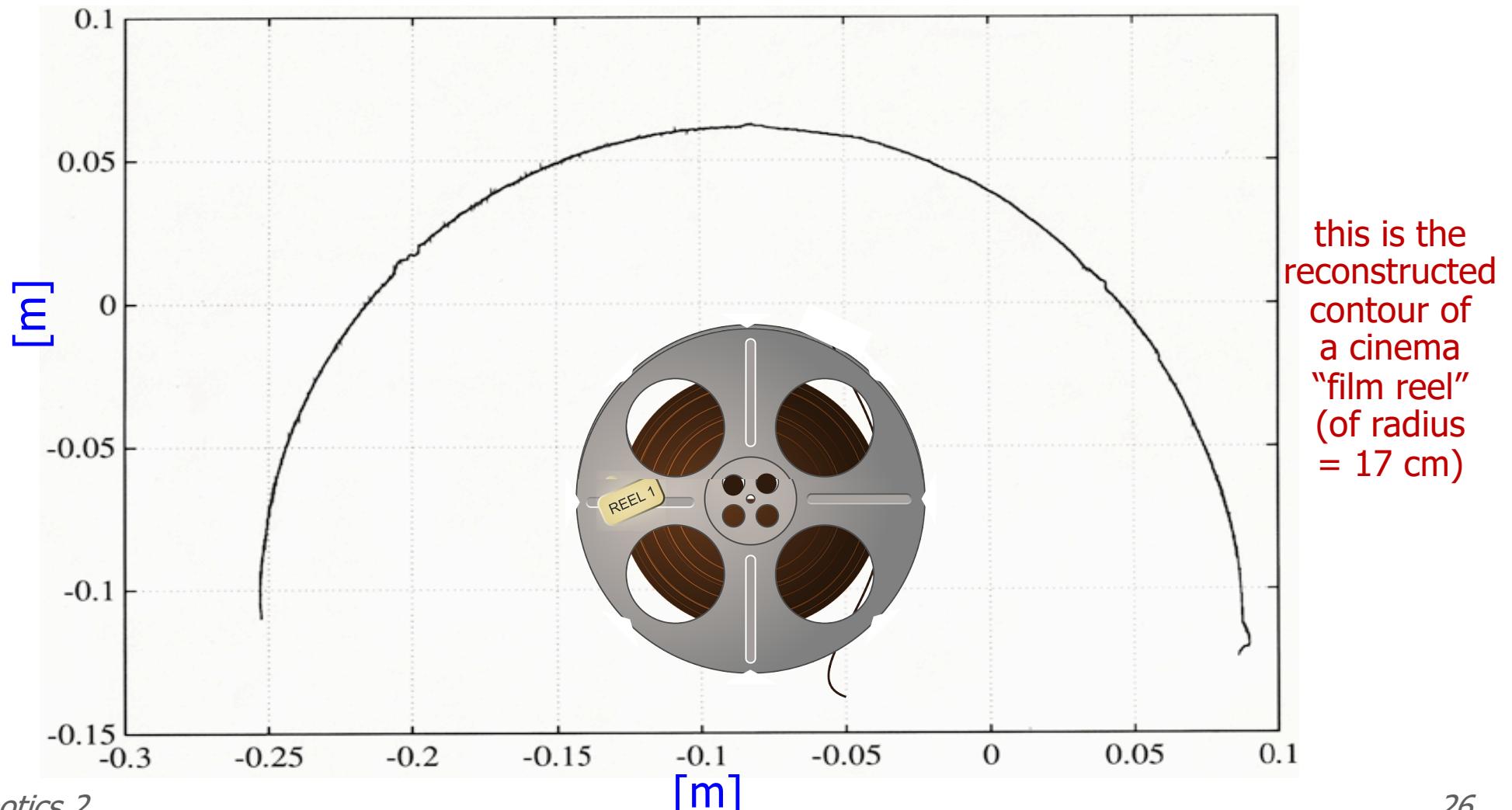
Difference between estimated tangents





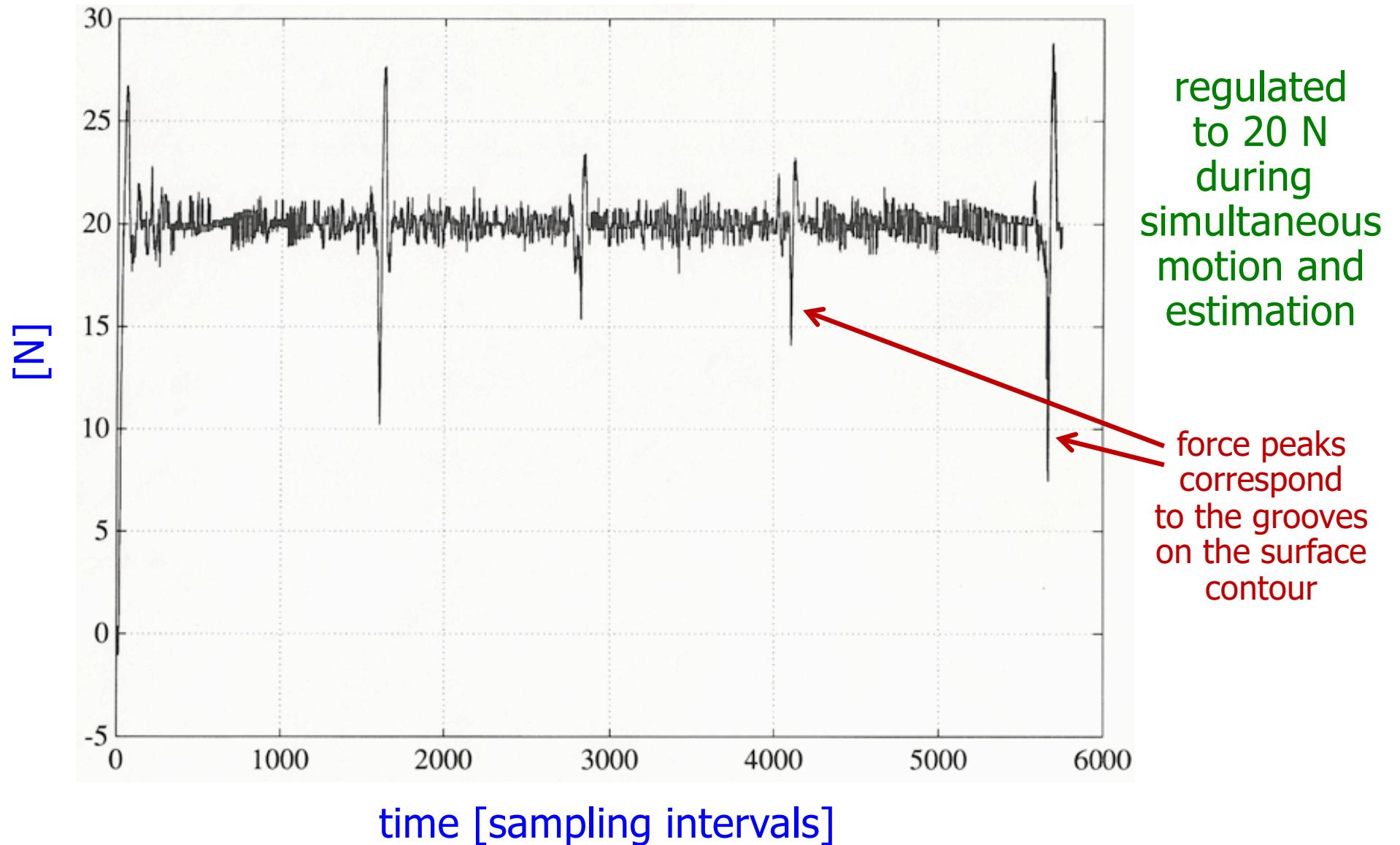
Reconstructed surface profile

estimation by a RLS (Recursive Least Squares) method: we continuously update the coefficients of two quadratic polynomials that fit locally the unknown contour, using data fusion from both force and position/velocity measurements



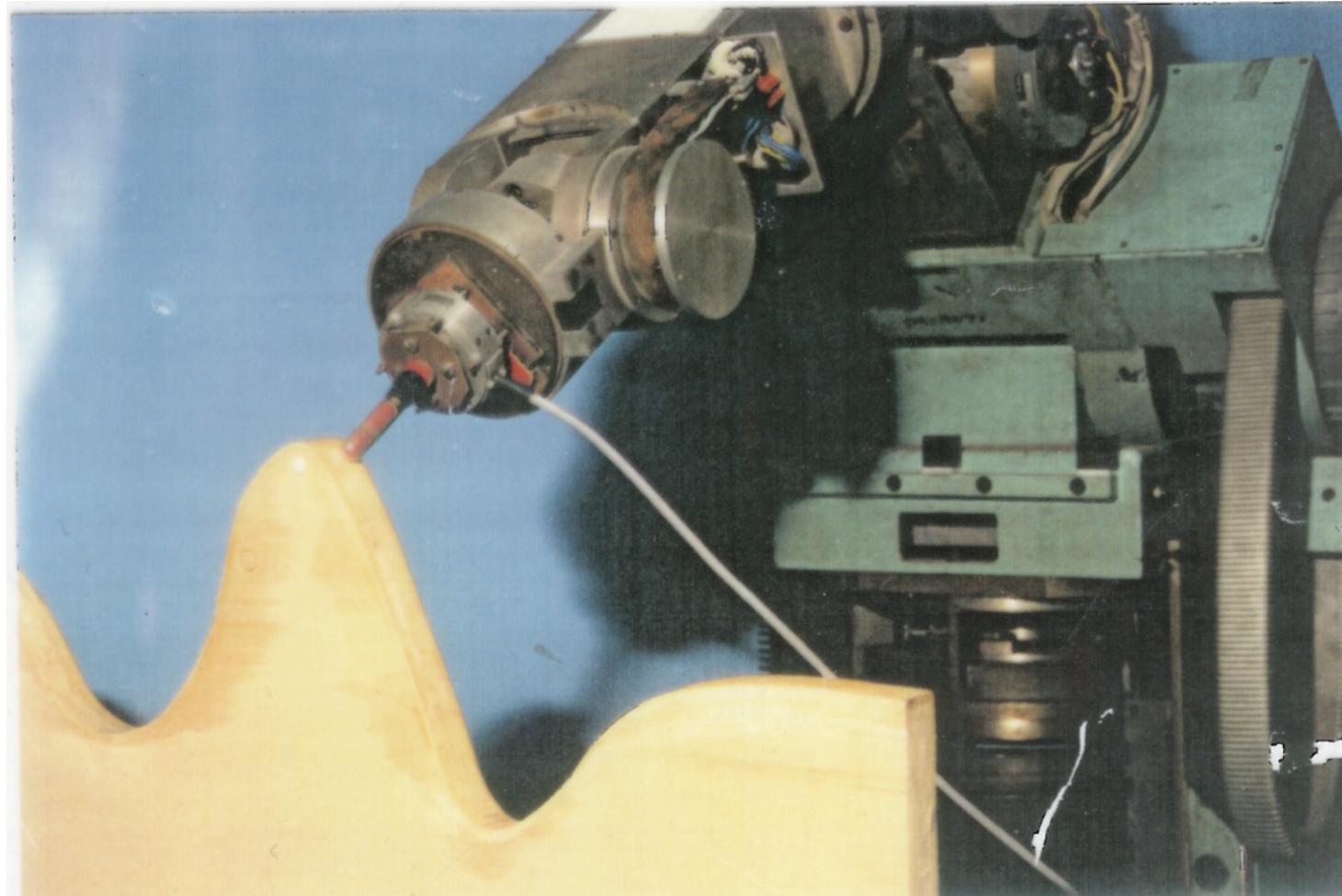


Normal force





Contour estimation and hybrid control performed simultaneously



MIMO-CRF robot (DIS, Laboratorio di Robotica, 1992)



Contour estimation and hybrid control

Hybrid Force/Velocity Control and Identification of Surfaces

**Università di Roma "La Sapienza"
DIS, LabRob
September 1992**



video

Robotized deburring of car windshields



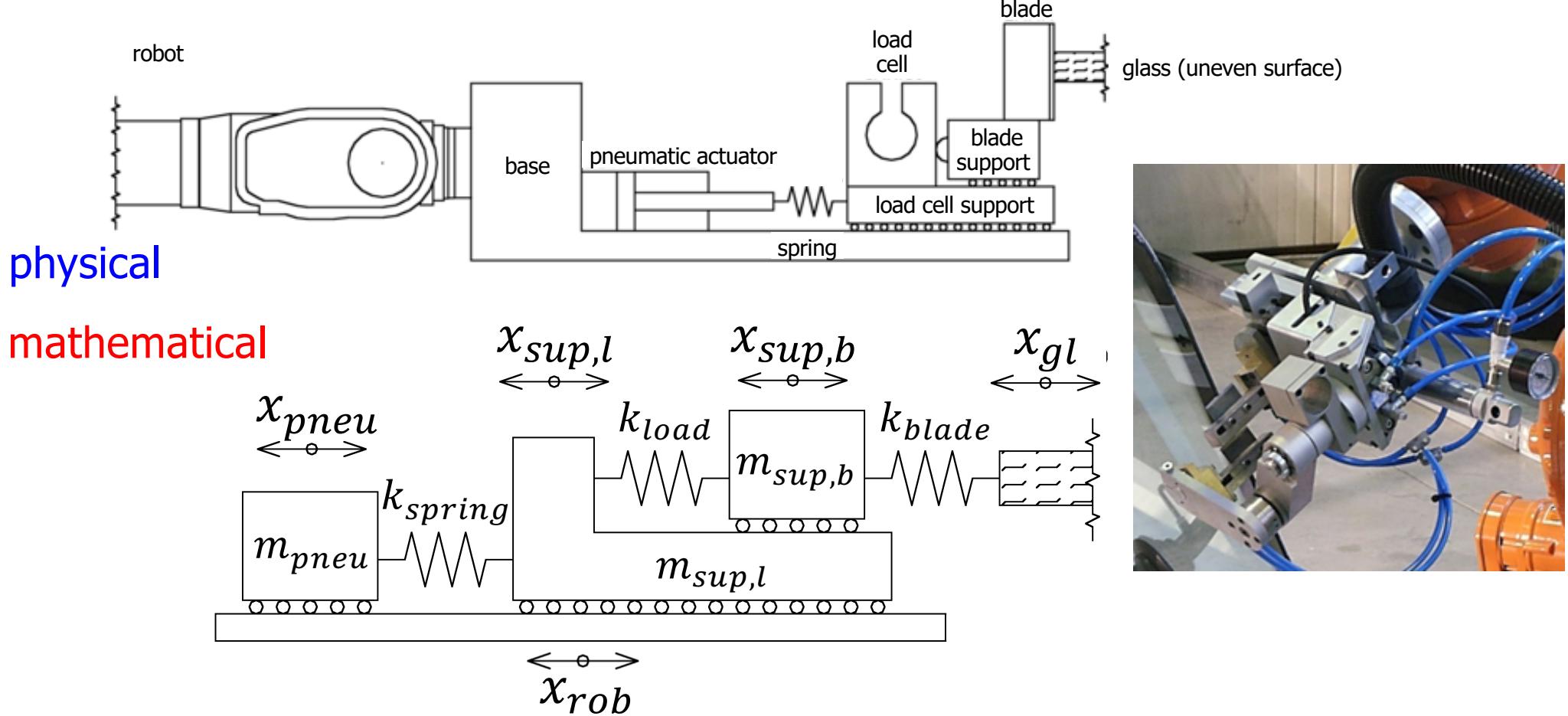
- car windshield with **sharp edges** and fabrication tolerances, with **excess of material** (PVB = Polyvinyl butyral for gluing glass layers) on the contour
- robot end-effector follows the pre-programmed path, despite the small errors w.r.t. the nominal windshield profile, thanks to the **compliance** of the deburring tool
- contact force between tool blades and workpiece can be independently controlled by a **pneumatic actuator** in the tool

the robotic deburring tool contains in particular

- **two blades** for cutting the exceeding plastic material (PVB), the first one actuated, the second passively pushed against the surface by a spring
- a **load cell** for measuring the 1D applied normal force at the contact
- on-board **control system**, exchanging data with the ABB robot controller



Model of the deburring work tool



for a stability analysis (based on linear models and root locus techniques) of force control in a single direction and in presence of multiple masses/springs, see again Eppinger & Seering, IEEE CSM, 1987 (material in the course web site)

Reducing gain = problem in free space (poor accuracy)
good response under force (?)
↳ no movements



Summary through video segments



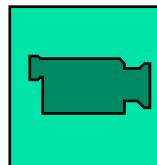
compliance control
(active Cartesian stiffness
control **without** F/T sensor)



impedance control
(with F/T sensor)



force control
(realized as external loop
providing the reference to
an internal position loop
–see Appendix)



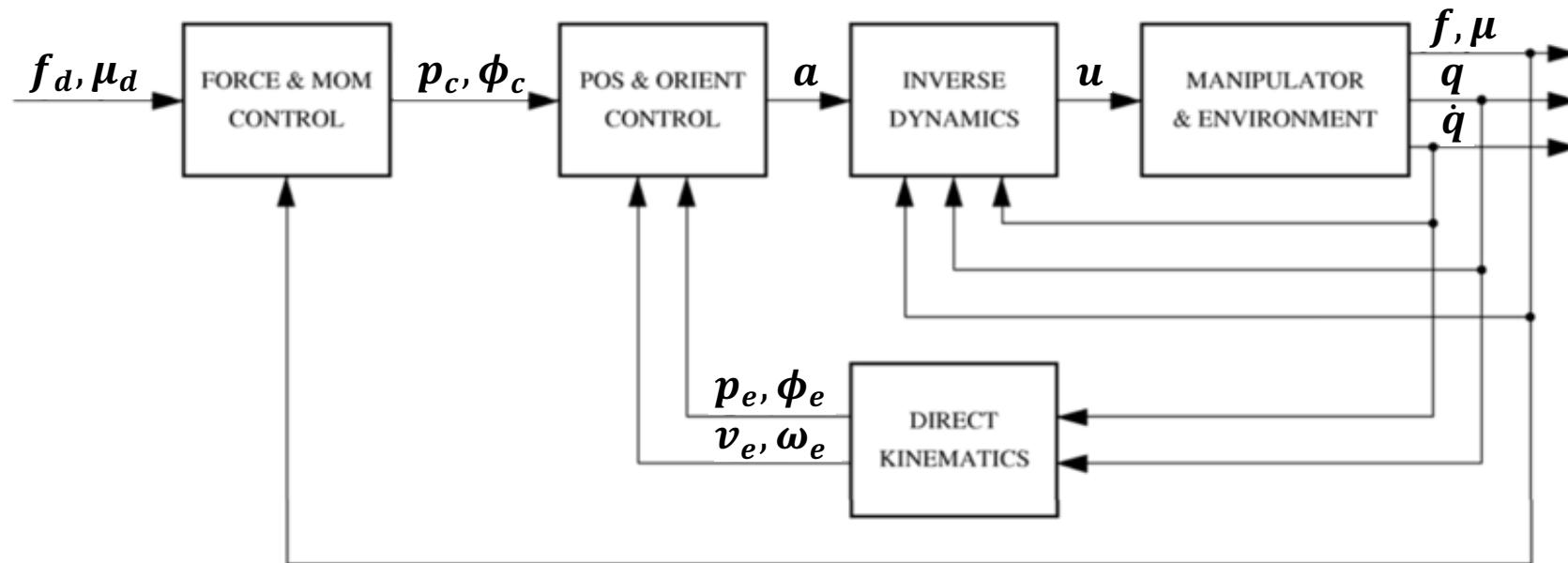
hybrid force/position control

COMAU Smart robot
c/o Università di Napoli, 1994
(full video on course web site)



Appendix

- force control can also be realized as an external loop providing reference values to an internal motion loop (see video in slide #32)
- inner-outer (or cascaded) control scheme
 - angular position quantities (E-E orientation, errors, commands) can be expressed in different ways (Euler angles ϕ , rotation matrices R , ...)





Robotics 2

Visual servoing

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Visual servoing

- objective
 - use information acquired by **vision sensors** (cameras) for **feedback control** of the pose/motion of a robot (or of parts of it)
- + data acquisition ~ human eye, with very large information content in the acquired images
- difficult to extract essential data, nonlinear perspective transformations, high sensitivity to ambient conditions (lightening), noise



Some applications

automatic navigation of robotic systems (agriculture, automotive)

video



video



surveillance

video



bio-motion synchronization (surgical robotics)

video





Image processing

- **real-time** extraction of characteristic parameters useful for robot motion control
 - **features:** points, area, geometric center, higher-order moments, ...
- low-level processing
 - binarization (threshold on grey levels), equalization (histograms), edge detection, ...
- segmentation
 - based on regions (possibly in binary images)
 - based on contours
- interpretation
 - association of characteristic parameters (e.g., texture)
 - problem of **correspondence** of points/characteristics of objects in different images (stereo or on image flows)

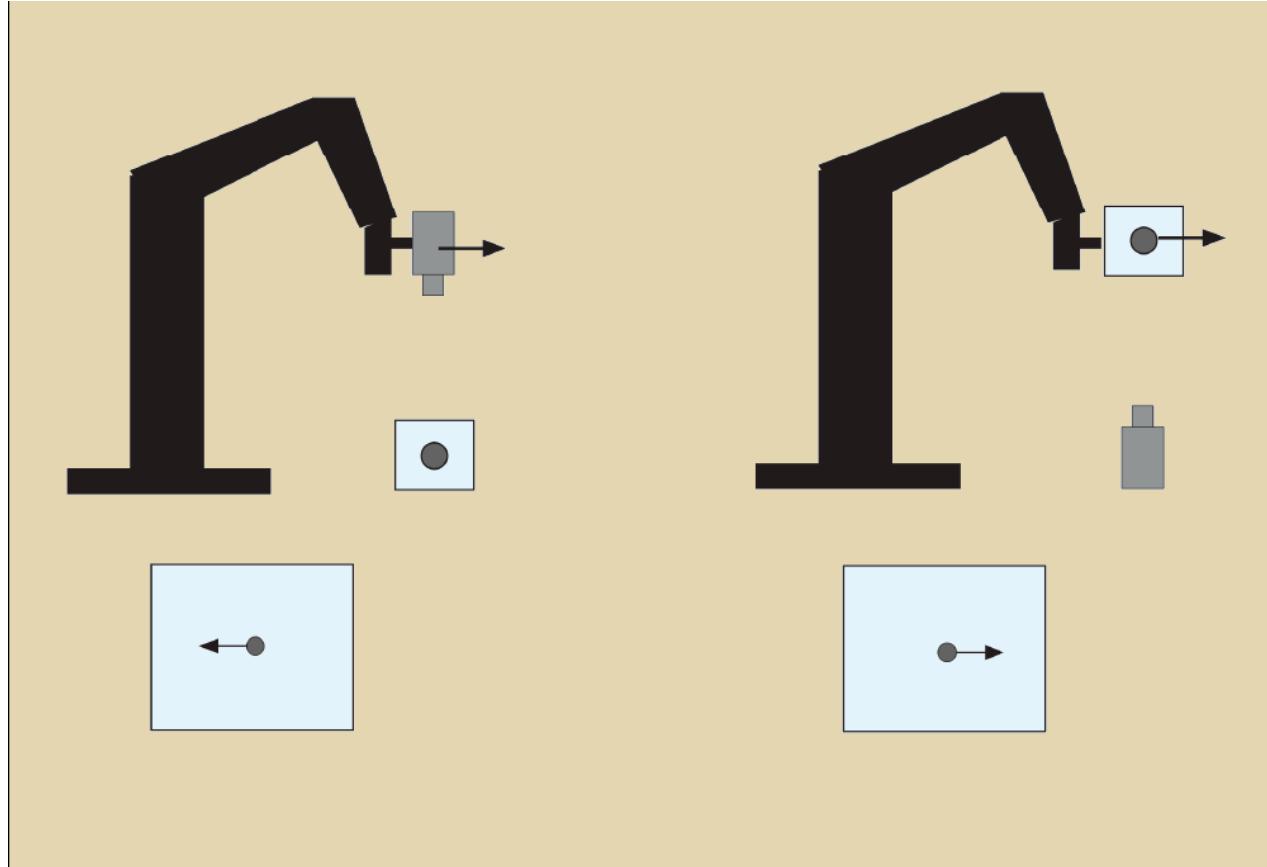


Configuration of a vision system

- one, two, or more cameras
 - grey-level or color
- 3D/stereo vision
 - obtained even with a single (moving) camera, with the object taken from different (known) points of view
- camera positioning
 - fixed (eye-to-hand)
 - mounted on the manipulator (eye-in-hand)
- robotized vision heads
 - motorized (e.g., stereo camera on humanoid head or pan-tilt camera on Magellan mobile robot)



Camera positioning



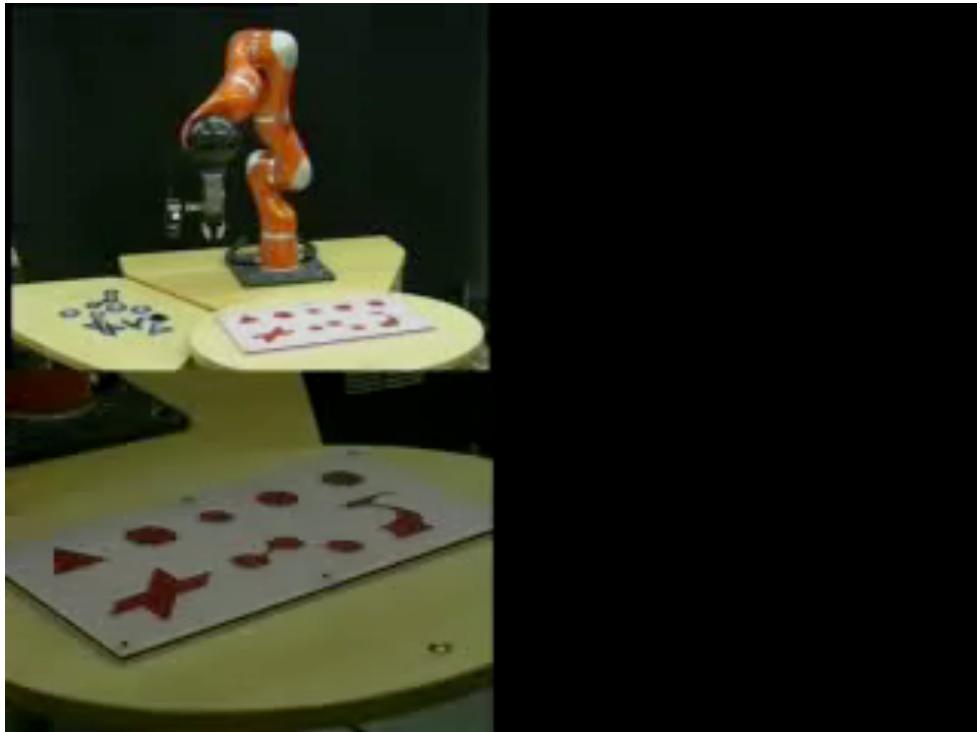
eye-in-hand

eye-to-hand



Vision for assembly

video



video

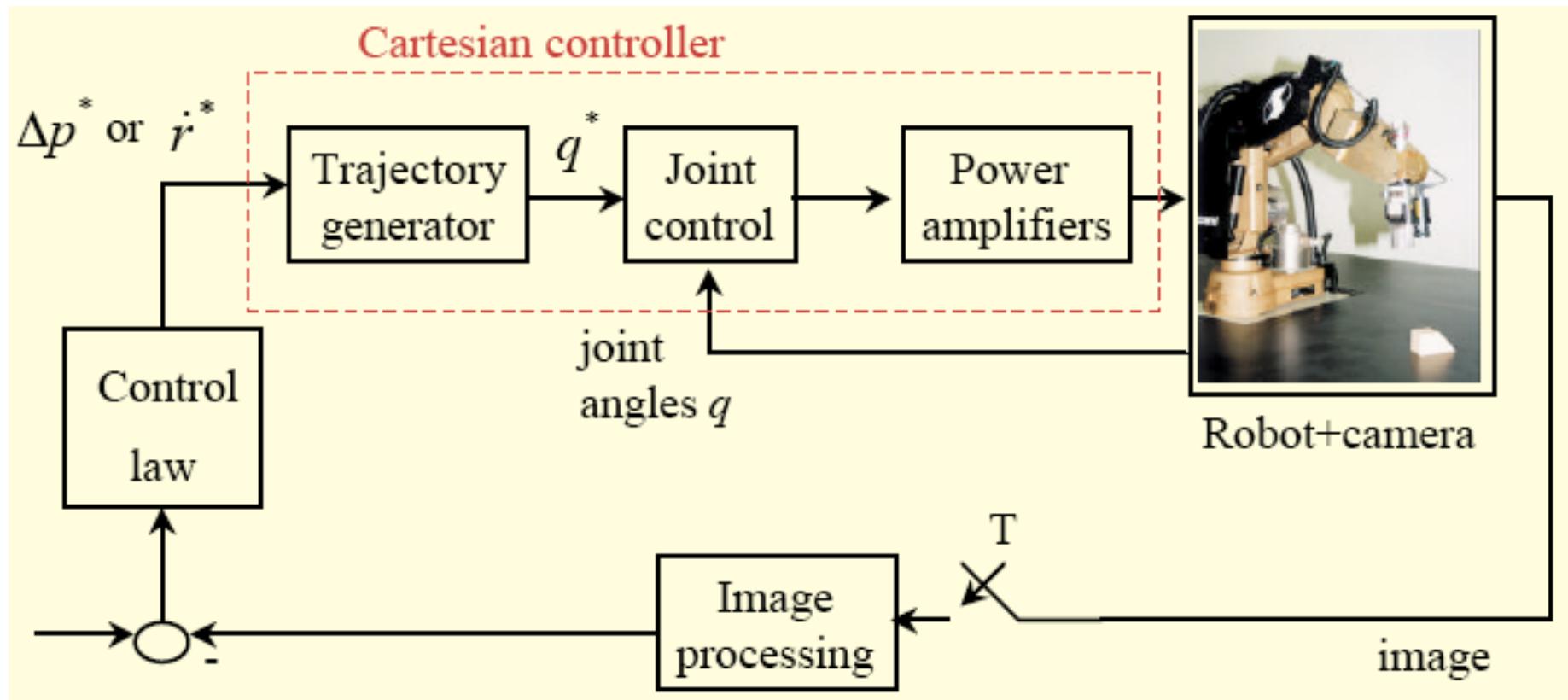


PAPAS-DLR system
(eye-in-hand, **hybrid force-vision**)

robust w.r.t. motion of
target objects



Indirect/external visual servoing

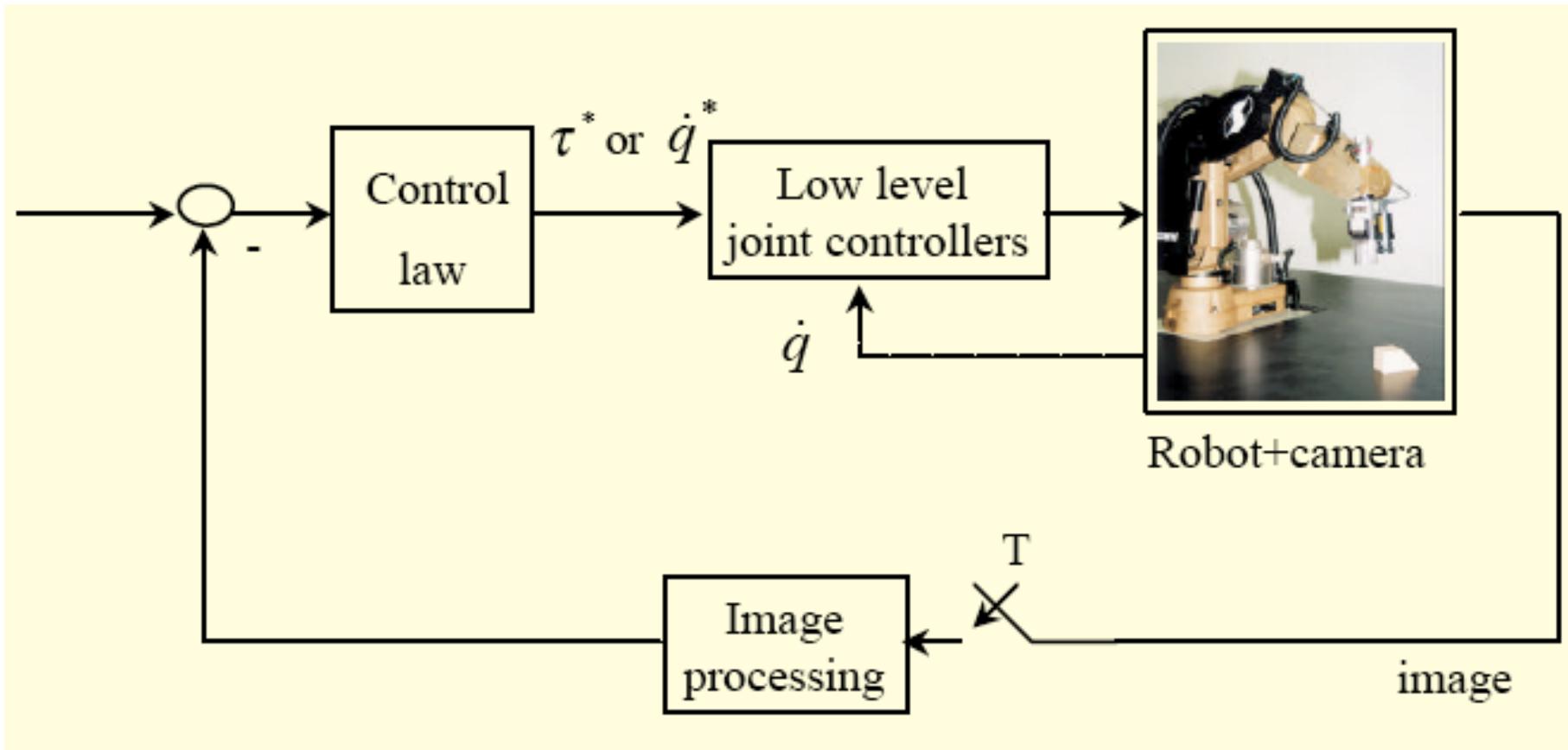


vision system **provides set-point references** to a **Cartesian motion controller**

- “easy” control law (same as without vision)
- appropriate for relatively slow situations (control sampling $f = 1/T < 50\text{Hz}$)



Direct/internal visual servoing



replace Cartesian controller with one based on vision that **directly computes joint reference commands**

- control law is more complex (involves robot kinematics/dynamics)
- preferred for fast situations (control sampling $f = 1/T > 50\text{Hz}$)



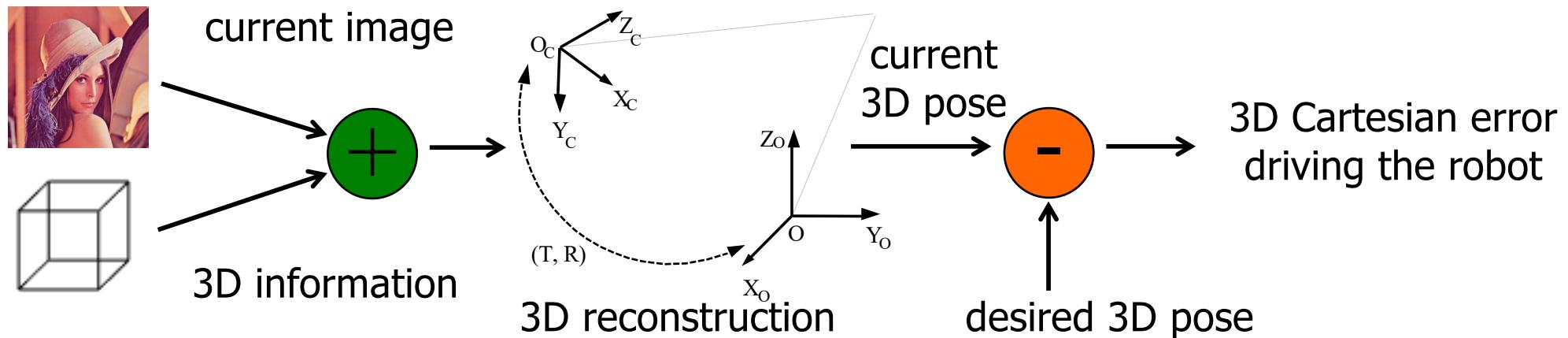
Classification of visual servoing schemes

- position-based visual servoing (**PBVS**)
 - information extracted from images (**features**) is used to reconstruct the **current 3D pose** (pose/orientation) of an object
 - combined with the knowledge of a **desired 3D pose**, we generate a **Cartesian** pose error signal that drives the robot to the goal
- image-based visual servoing (**IBVS**)
 - error is computed directly on the values of the features extracted on the **2D image plane**, **without** going through a 3D reconstruction
 - the robot should move so as to bring the current image features (what it “sees” with the camera) to their desired values
- some mixed schemes are possible (e.g., **2½D methods**)

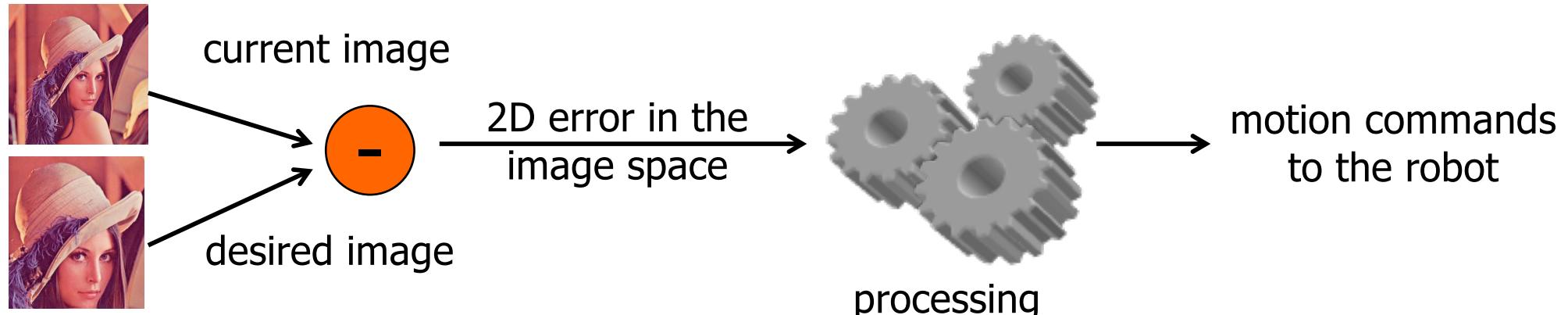


Comparison between the two schemes

- position-based visual servoing (**PBVS**)

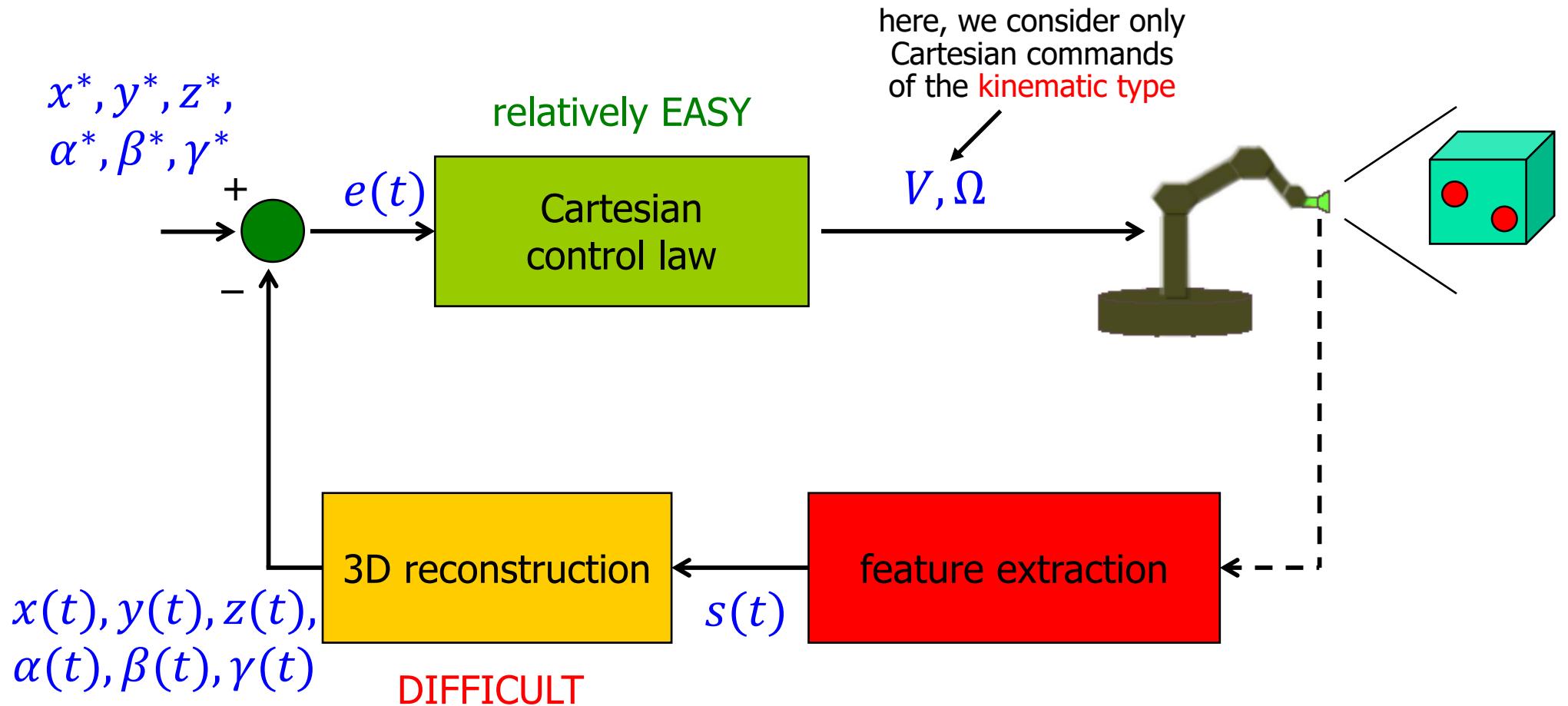


- image-based visual servoing (**IBVS**)





PBVS architecture

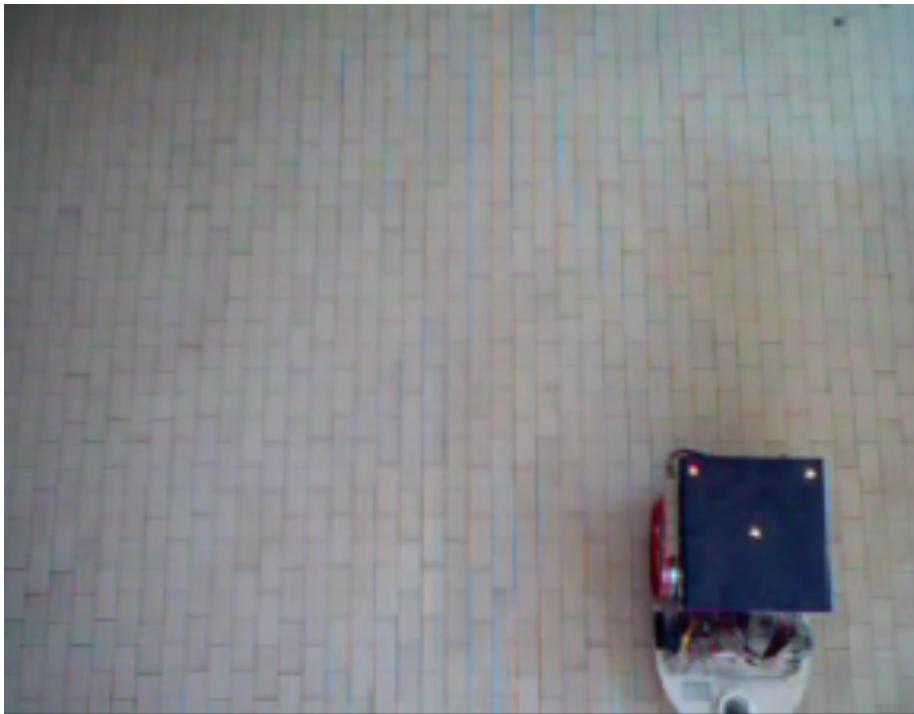


highly sensitive to camera calibration parameters



Examples of PBVS

video



eye-to-“robot” (SuperMario)

position/orientation of the camera
and scene geometry

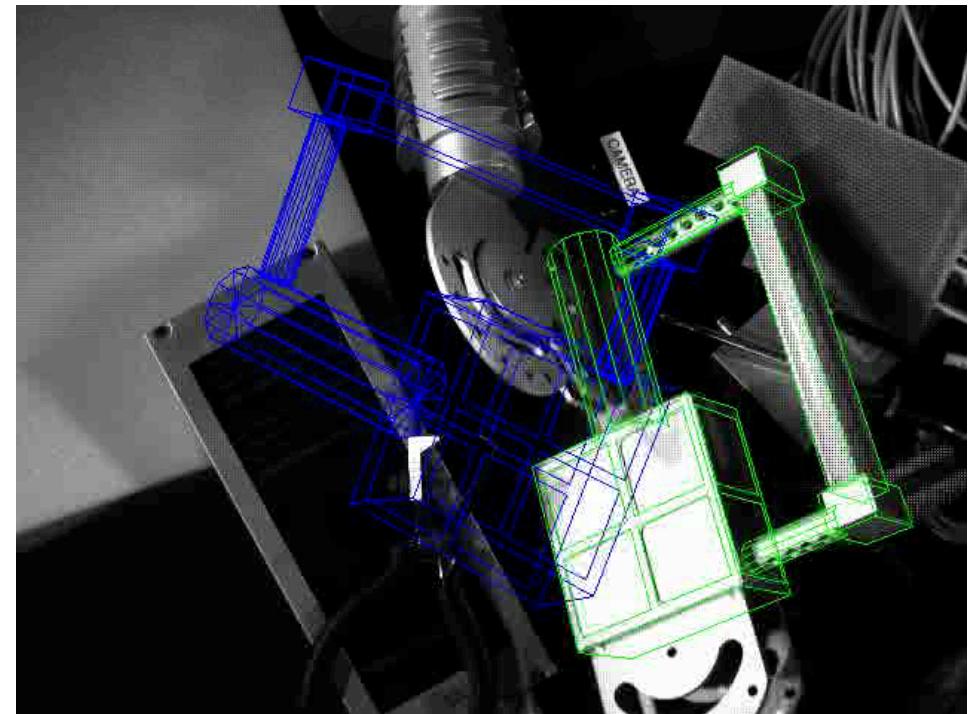


eye-in-hand (Puma robot)

position and orientation of the robot
(with mobile or fixed base)

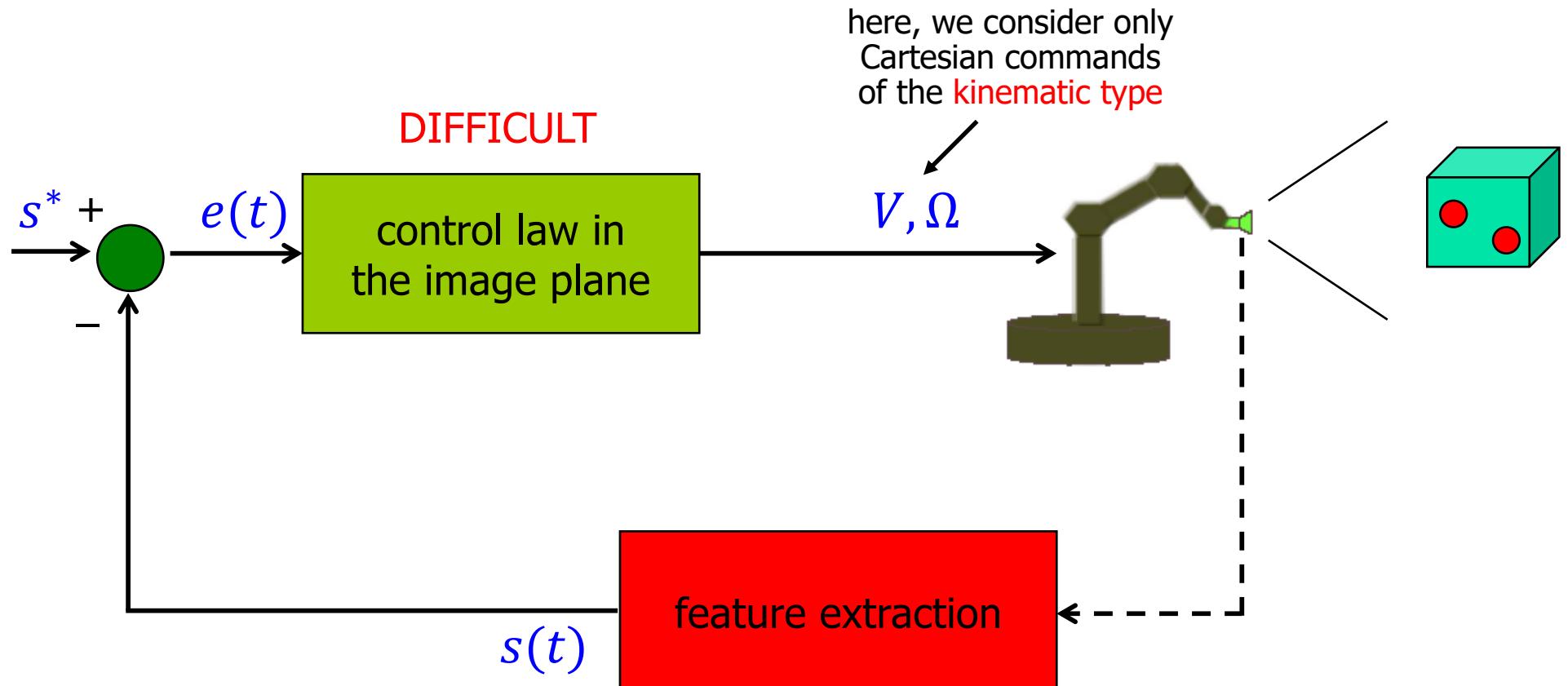
known a priori!

video





IBVS architecture

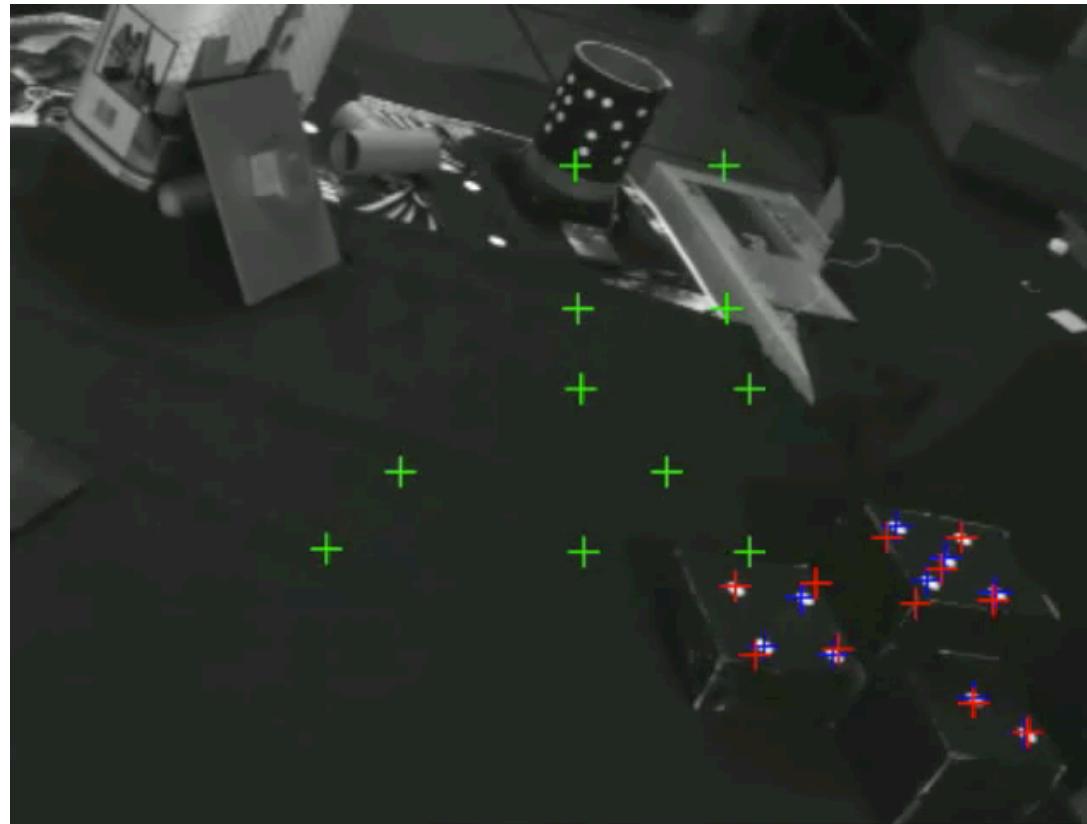


almost insensitive to intrinsic/extrinsic camera calibration parameters



An example of IBVS

here, the features are **points** (selected from the given set, or in suitable combinations)



video

desired feature positions
current feature positions



the error in the image plane (task space!) drives/controls the motion of the robot



PBVS vs IBVS

PBVS = position-based
visual servoing

video



IBVS = image-based
visual servoing

video



F. Chaumette, INRIA Rennes

reconstructing the instantaneous
(relative) 3D pose of the object

using (four) point features
extracted from the 2D image

...and intermediate 2½-D visual servoing...

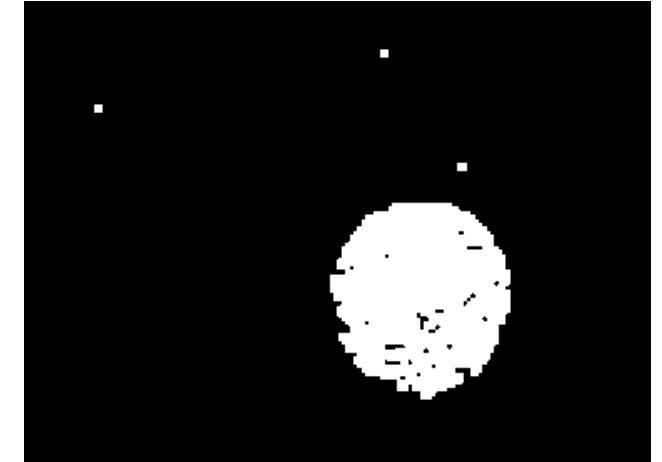
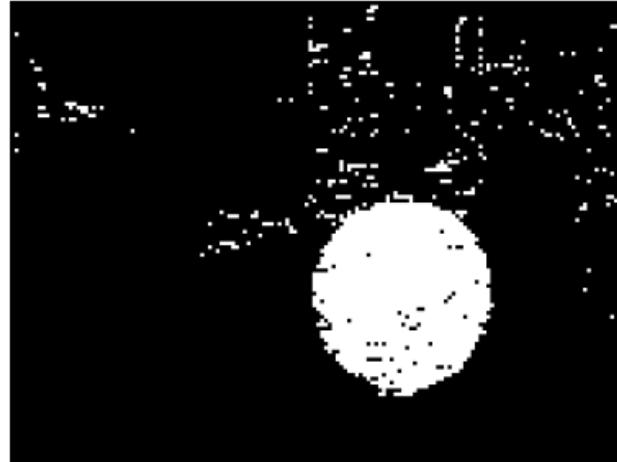
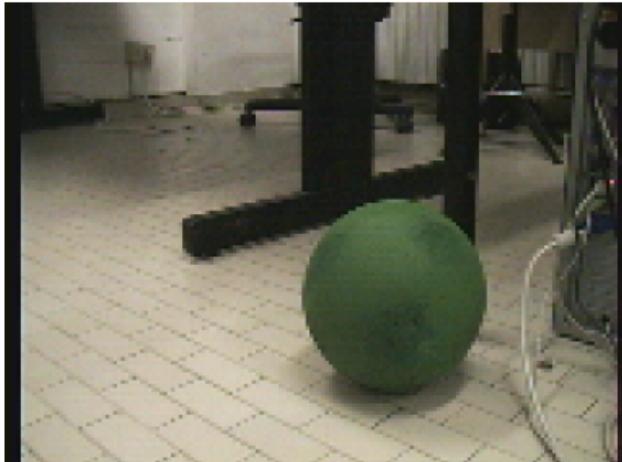


Steps in an IBVS scheme

- **image acquisition**
 - frame rate, delay, noise, ...
- **feature extraction**
 - with image processing techniques (it could be a difficult and time consuming step!)
- **comparison with “desired” feature values**
 - definition of an **error** signal in the **image plane**
- **generation of motion of the camera/robot**
 - perspective transformations
 - differential kinematics of the manipulator
 - control laws of **kinematic** (most often) or **dynamic** type (e.g., PD + gravity cancelation --- **see reference textbook**)



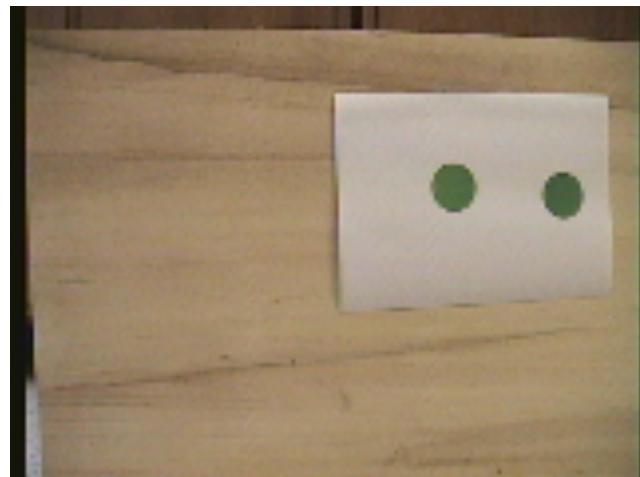
Image processing techniques



binarization in **RGB** space



erosion and dilation



binarization in **HSV** space



dilation

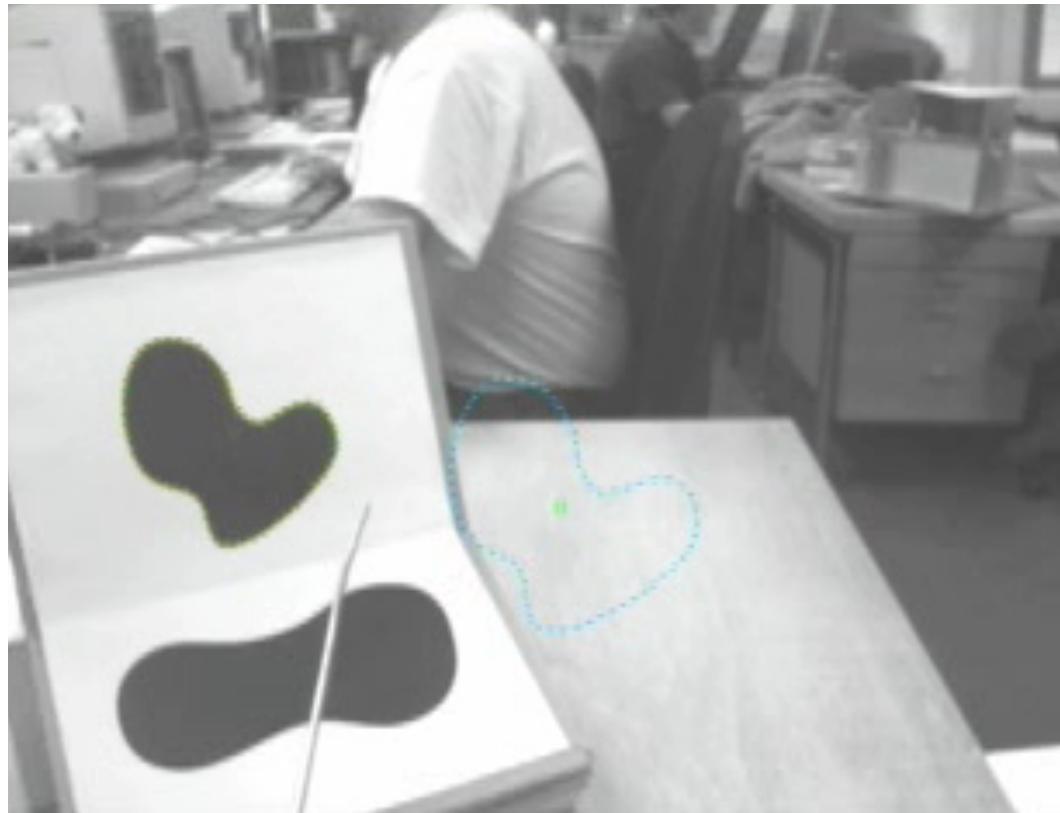


What is a feature?

- **image feature**: any interesting **characteristic** or **geometric structure** extracted from the image/scene
 - points
 - lines
 - ellipses (or any other 2D contour)
- **feature parameter(s)**: any **numerical quantity** associated to the selected feature in the image plane
 - coordinates of a point
 - angular coefficient and offset of a line
 - center and radius of a circle
 - area and center of a 2D contour
 - generalized **moments** of an object in the image
 - can also be defined so as to be scale- and rotation-invariant



Example of IBVS using moments



video

- avoids the problem of “finding **correspondences**” between points
- however, it is not easy to control the motion of all **6 dofs of the camera** when using only moments as features

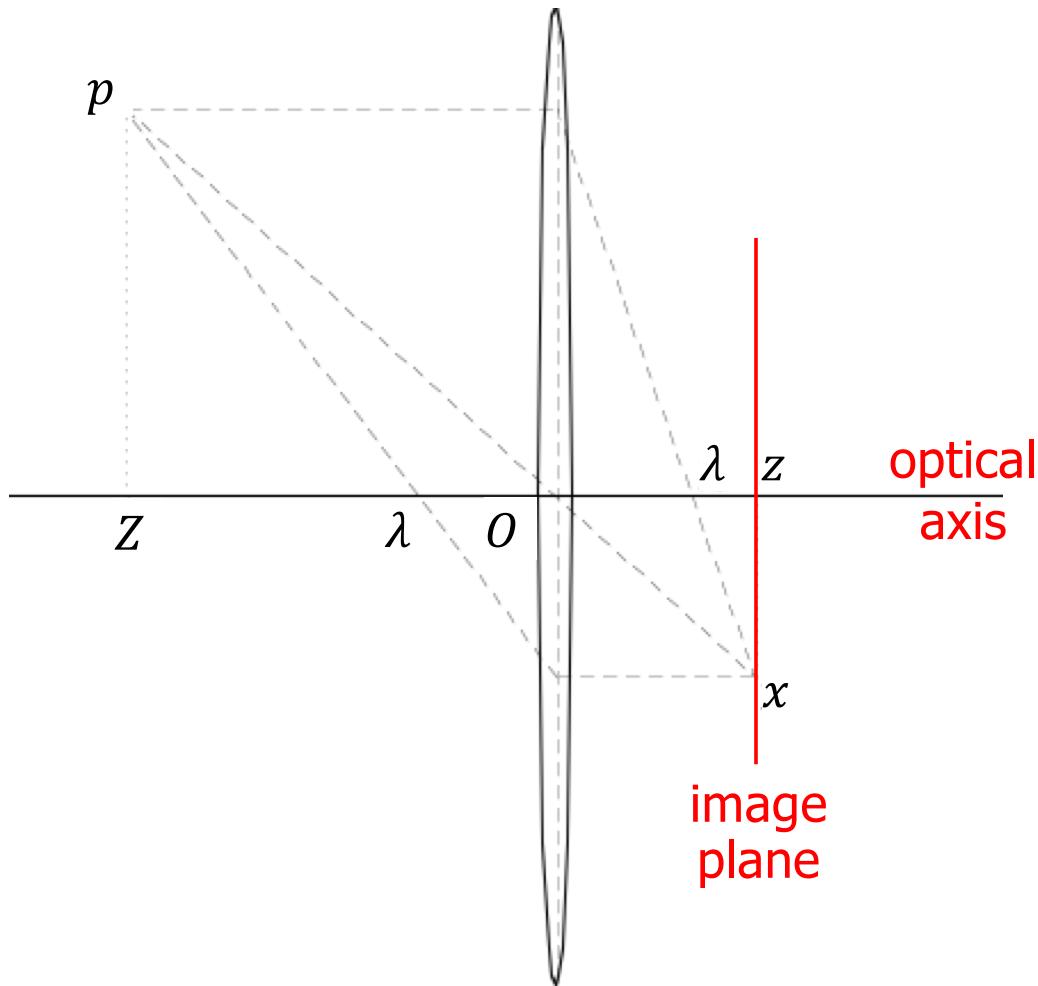


Camera model

- set of **lenses** to focus the incoming light
 - (converging) thin lenses
 - pinhole approximation
 - catadioptric lens or systems (combined with mirrors)
- matrix of light sensitive elements (pixels in **image plane**), possibly selective to **RGB** colors ~ human eye
- **frame grabber** “takes shots”: an electronic device that captures individual, digital still frames as output from an analog video signal or a digital video stream
 - board + software on PC
 - **frame rate** = output frequency of new digital frames
 - it is useful to randomly access a subset (area) of pixels



Thin lens camera model



- geometric/projection optics
- rays parallel to the optical axis are deflected through a point at distance λ (**focal length**)
- rays passing through the optical center O are **not** deflected

fundamental equation
of a thin lens

$$\frac{1}{Z} + \frac{1}{z} = \frac{1}{\lambda}$$

} dioptic
(optical)
power

proof? left as exercise ...



Pinhole camera model

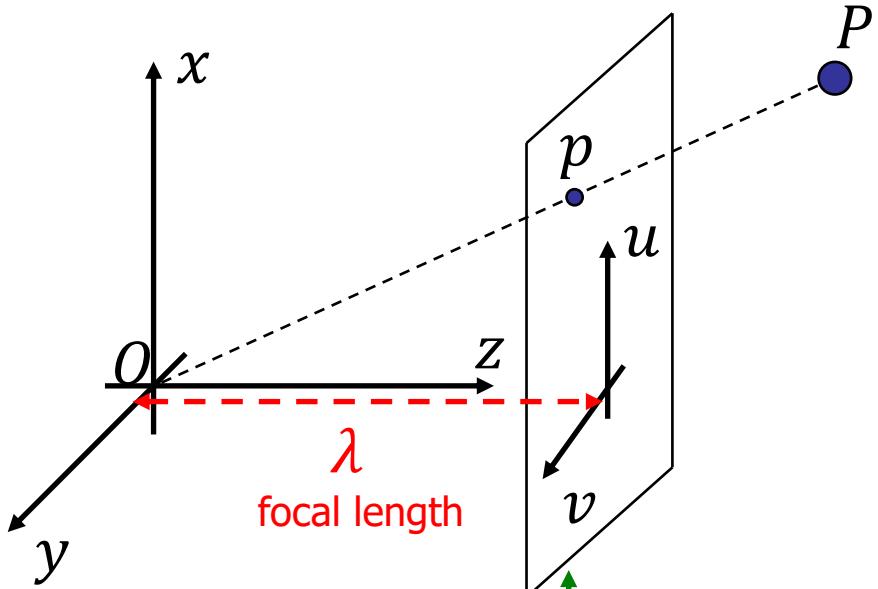


image plane with an array
of light sensitive elements
(actually located symmetrically
to the optical center O , but this
model avoids a change of signs...)

- when thin lens wideness is neglected
- all rays pass through optical center
- from the relations on similar triangles one gets the **perspective equations**

$$u = \lambda \frac{X}{Z} \quad v = \lambda \frac{Y}{Z}$$

to obtain these from discrete pixel values, an **offset** and a **scaling factor** should be included in each direction

sometimes **normalized** values (u', v') are used
(dividing by the focal length)

with

$$P = (X, Y, Z)$$

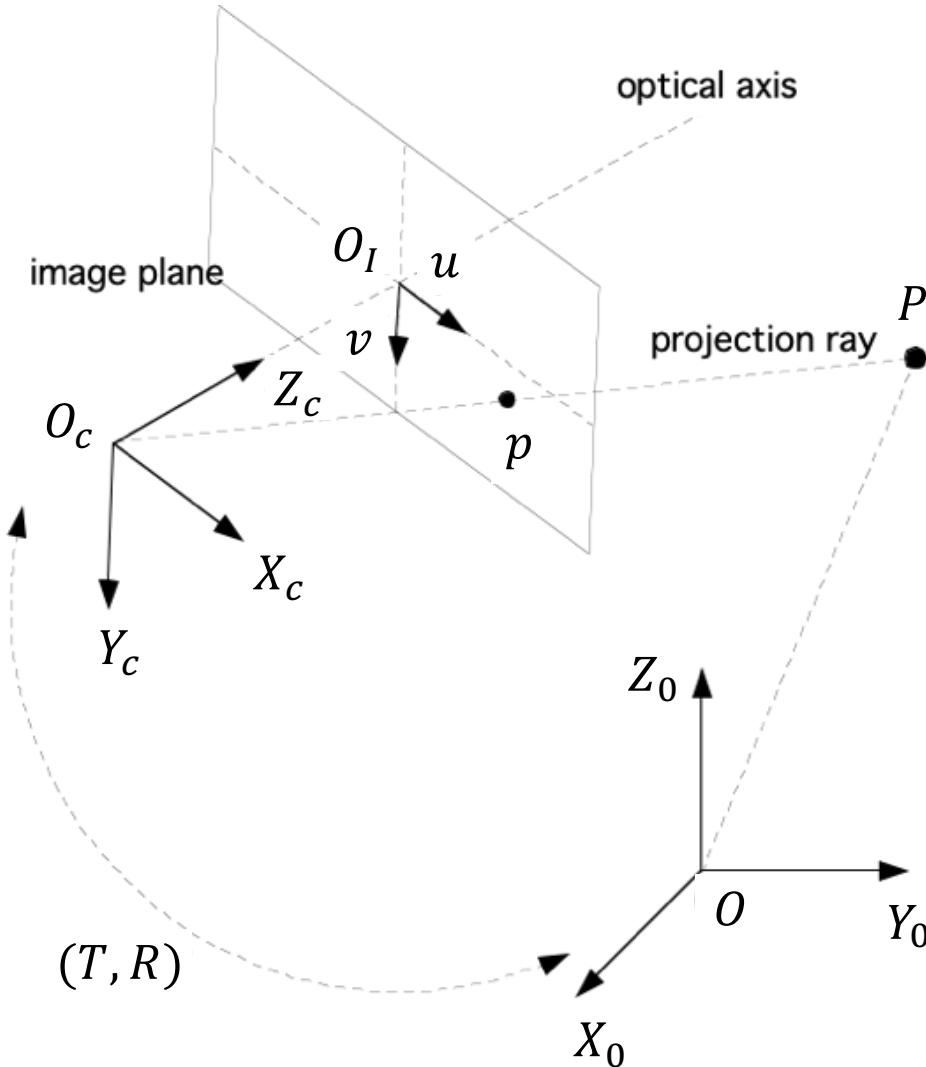
Cartesian point
(in camera frame)

$$p = (u, v, \lambda)$$

representative point
on the image plane



Reference frames of interest



- **absolute reference frame**
 $\mathcal{F}_0: \{O, \vec{X}_0, \vec{Y}_0, \vec{Z}_0\}$
- **camera reference frame**
 $\mathcal{F}_c: \{O_c, \vec{X}_c, \vec{Y}_c, \vec{Z}_c\}$
- **image plane reference frame**
 $\mathcal{F}_I: \{O_I, \vec{u}, \vec{v}\}$
- **position/orientation of \mathcal{F}_c w.r.t. \mathcal{F}_0**
 (T, R)
 (translation, rotation)



Interaction matrix

- given a set of feature(s) parameters $f = [f_1 \quad \cdots \quad f_k]^T \in \mathbb{R}^k$
- we look for the **(kinematic) differential relation** between **motion imposed to the camera** and **motion of features** on the image plane

$$\dot{f} = J(\cdot) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

- $(V, \Omega) \in \mathbb{R}^6$ is the camera linear/angular velocity, a vector expressed in \mathcal{F}_c
- $J(\cdot)$ is a $k \times 6$ matrix, known as **interaction matrix**
- in the following, we consider mainly **point features**
 - other instances (areas, lines, ...) can be treated as extensions of the presented analysis



Computing the interaction matrix point feature, pinhole model

- from the perspective equations $u = \lambda \frac{X}{Z}$, $v = \lambda \frac{Y}{Z}$, we have

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{\lambda}{z} & 0 & -\frac{u}{z} \\ 0 & \frac{\lambda}{z} & -\frac{v}{z} \end{bmatrix} \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = J_1(u, v, \lambda) \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix}$$

- the velocity $(\dot{X}, \dot{Y}, \dot{Z})$ of a point P in frame \mathcal{F}_c is **actually due to** the roto-translation (V, Ω) of the camera (P is assumed fixed in \mathcal{F}_0)
- kinematic relation between $(\dot{X}, \dot{Y}, \dot{Z})$ and (V, Ω)

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = -V - \Omega \times \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Note: ALL quantities are expressed in the camera frame \mathcal{F}_c
without explicit indication of subscripts



Computing the interaction matrix (cont)

point feature, pinhole model

- the last equation can be expressed in matrix form

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & -Z & Y \\ 0 & -1 & 0 & Z & 0 & -X \\ 0 & 0 & -1 & -Y & X & 0 \end{bmatrix} \begin{bmatrix} V \\ \Omega \end{bmatrix} = J_2(X, Y, Z) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

- combining things, the **interaction matrix** for a **point feature** is

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = J_1 J_2 \begin{bmatrix} V \\ \Omega \end{bmatrix} = \begin{bmatrix} -\frac{\lambda}{Z} & 0 & \frac{u}{Z} & \frac{uv}{\lambda} & -\left(\lambda + \frac{u^2}{\lambda}\right) & v \\ 0 & -\frac{\lambda}{Z} & \frac{v}{Z} & \lambda + \frac{v^2}{\lambda} & -\frac{uv}{\lambda} & -u \end{bmatrix} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

$$= J_p(u, v, Z) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$



p = point (feature)

here, λ is assumed to be known



Comments

- the **interaction matrix** in the map

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = J_p(u, v, Z) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

- depends on the actual value of the **feature** and on its **depth Z**
- since $\dim \ker J_p = 4$, there exist ∞^4 motions of the camera that are unobservable (for this feature) on the image
 - for instance, a translation along the projection ray
- when **more point features** are considered, the associated interaction matrices are stacked one on top of the other
 - p point features: the interaction matrix is $k \times 6$, with $k = 2p$



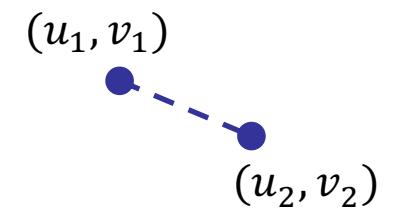
Other examples of interaction matrices

- distance between two point features

$$d = \sqrt{(u_1 - u_2)^2 + (v_1 - v_2)^2}$$

$$\dot{d} = \frac{1}{d} [u_1 - u_2 \quad v_1 - v_2 \quad u_2 - u_1 \quad v_2 - v_1] \begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \dot{u}_2 \\ \dot{v}_2 \end{bmatrix}$$

$$= J_p(u_1, u_2, v_1, v_2) \begin{bmatrix} J_{p1}(u_1, v_1, Z_1) \\ J_{p2}(u_2, v_2, Z_2) \end{bmatrix} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$



- image moments

$$m_{ij} = \iint_{\mathcal{R}(t)} x^i y^j dx dy$$

$\mathcal{R}(t)$ region of the image plane
occupied by the object

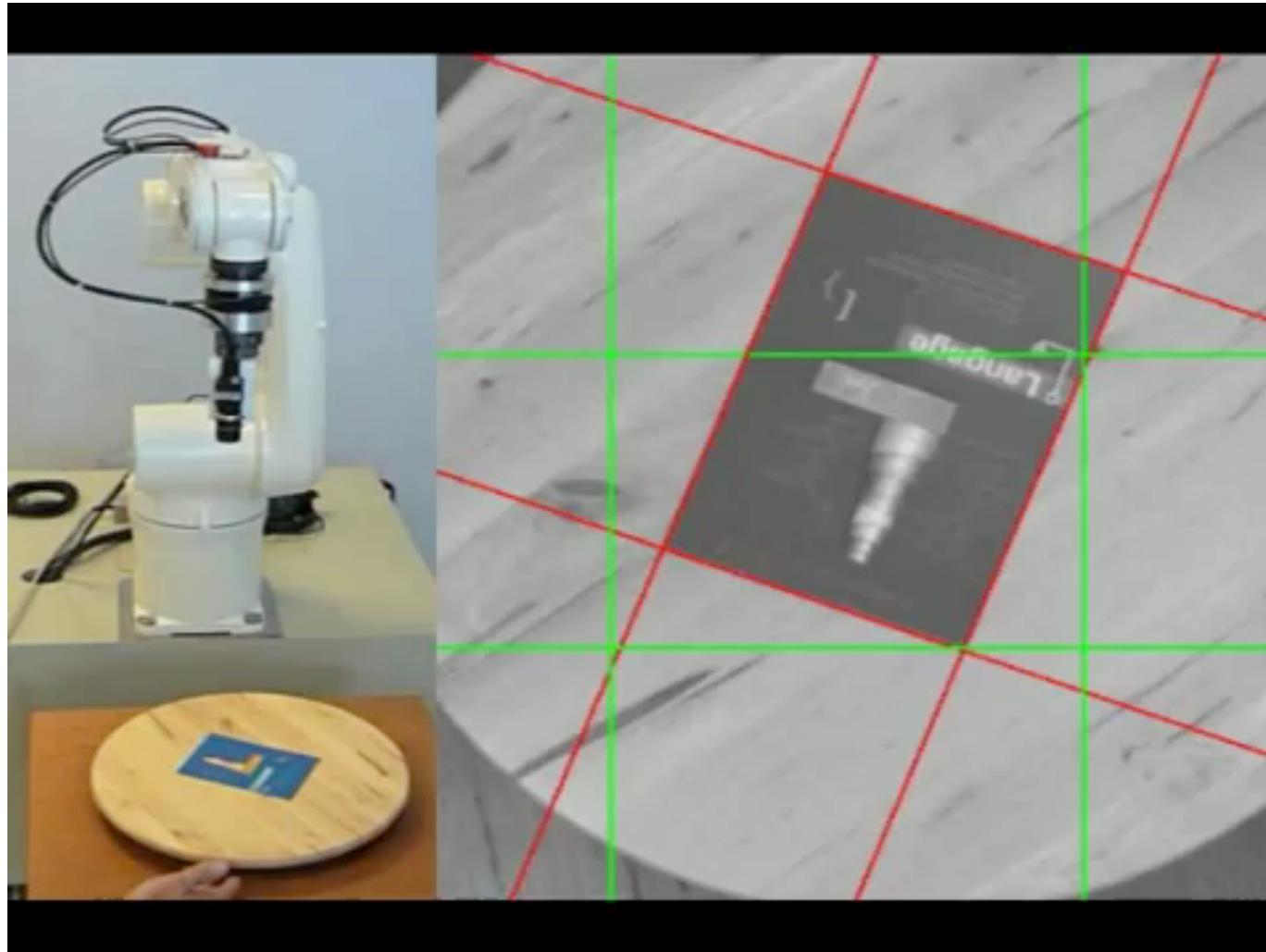
- useful for representing also qualitative geometric information ([area](#), [center](#), [orientation of an approximating ellipse](#), ...)
- by using Green formulas and the interaction matrix of a generic point feature

$$\rightarrow \dot{m}_{ij} = L_{ij} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

(see F. Chaumette: "Image moments:
A general and useful set of features for visual servoing",
IEEE Trans. on Robotics, August 2004)



IBVS with straight lines as features



F. Chaumette, INRIA Rennes



Robot differential kinematics

- **eye-in-hand** case: camera is mounted on the **end-effector** of a manipulator arm (with fixed base or on a wheeled mobile base)
 - the motion (V, Ω) to be imposed to the camera coincides with the desired **end-effector linear/angular velocity** and is realized by choosing the manipulator **joint velocity** (or, more in general, the feasible **velocity commands** of a **mobile** manipulator)

$$\begin{bmatrix} V \\ \Omega \end{bmatrix} = J_m(q)\dot{q} = J_m(q)u$$

velocity
control
input

↑
Geometric Jacobian
of a manipulator

↑
... or the NMM Jacobian
of a mobile manipulator

for consistency with the previous interaction matrix, these Jacobians must be expressed in the camera frame \mathcal{F}_c

with $q \in \mathbb{R}^n$ being the robot configuration variables

- in general, an hand-eye calibration is needed for this Jacobian



Image Jacobian

- combining the step involved in the passage from the **velocity of point features** on the image plane to the **joint velocity/feasible velocity commands of the robot**, we finally obtain

$$\dot{f} = J_p(f, Z)J_m(q)u = J(f, Z, q)u$$

- matrix $J(f, Z, q)$ is called the **Image Jacobian** and plays the same role of a classical robot Jacobian
- we can thus apply one of the many standard kinematic (or even dynamics-based) control techniques for robot motion
- the (controlled) **output** is given by the vector of features in the image plane (the **task space!**)
- the **error** driving the control law is defined in this task space



Kinematic control for IBVS

- defining the error vector as $e = f_d - f$, the general choice

$$u = J^{\#}(\dot{f}_d + ke) + (I - J^{\#}J)u_0$$

minimum norm solution

term in $\ker J$ does not “move” the features

will **exponentially stabilize** the task error to zero (up to singularities, limit joint range/field of view, features exiting the image plane, ...)

- in the redundant case, vector u_0 (projected in the image Jacobian null space) can be used for optimizing behavior/criteria
- the error feedback signal **depends only on feature values**
- there is still a dependence of J on the **depths** Z of the features
 - use the constant and “known” values at the final **desired pose**
 - or, **estimate on line** their current values using a dynamic observer

$$J(f, Z^*, q)$$



Example with NMM

- mobile base (unicycle) + 3R manipulator
- eye-in-hand configuration
- 5 commands
 - linear and angular velocity of mobile base
 - velocities of the three manipulator joints
- task specified by 2 point features → 4 output variables



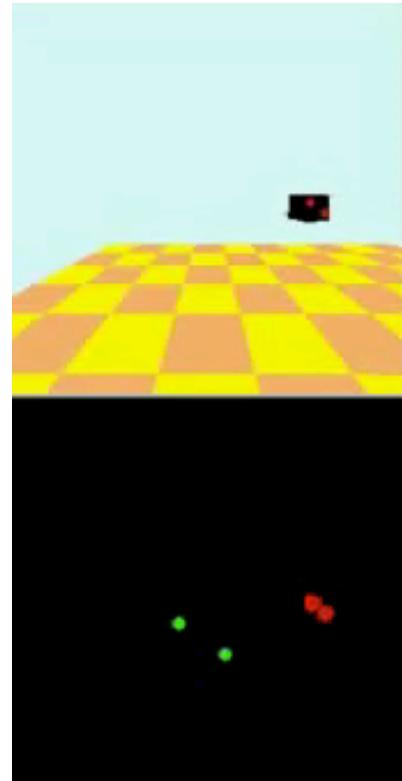
$$f = [f_{u1} \quad f_{v1} \quad f_{u2} \quad f_{v2}]^T$$

- $5 - 4 = 1$ degree of redundancy (w.r.t. the task)

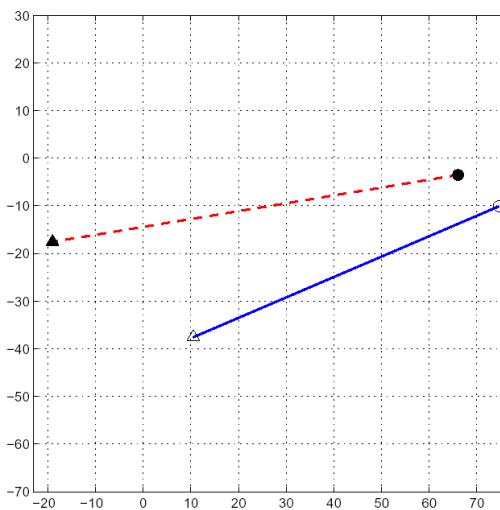


Simulation

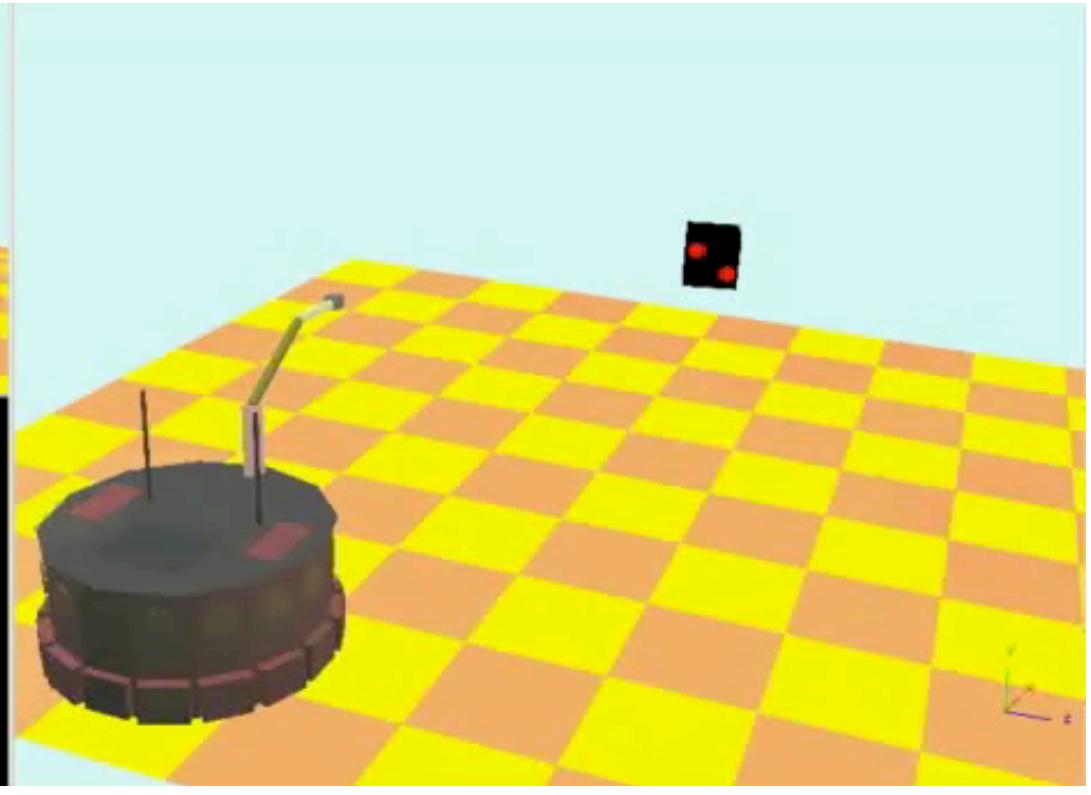
view from
camera



processed
image



behavior of
the 2 features



video

- simulation in Webots
- current value of Z is supposed to be known
- diagonal task gain matrix k (decoupling!)
- “linear paths” of features motion on the image plane



IBVS control with Task Sequencing

- approach: **regulate only one (some) feature** at the time, while keeping “**fixed**” the others by **unobservable motions** in the image plane

- Image Jacobians of single point features (e.g., $p = 2$)

$$\dot{f}_1 = J_1 u, \quad \dot{f}_2 = J_2 u$$

- the first feature f_1 is regulated during a first phase by using

$$u = J_1^\# k_1 e_1 + (I - J_1^\# J_1) u_0$$

- feature f_2 is then regulated by a command in the null-space of the first task

$$u = (I - J_1^\# J_1) J_2^T k_2 e_2$$

- in the first phase there are **two (more) degrees of redundancy** w.r.t. the “**complete**” case, which can be used, e.g., for improving robot alignment to the target

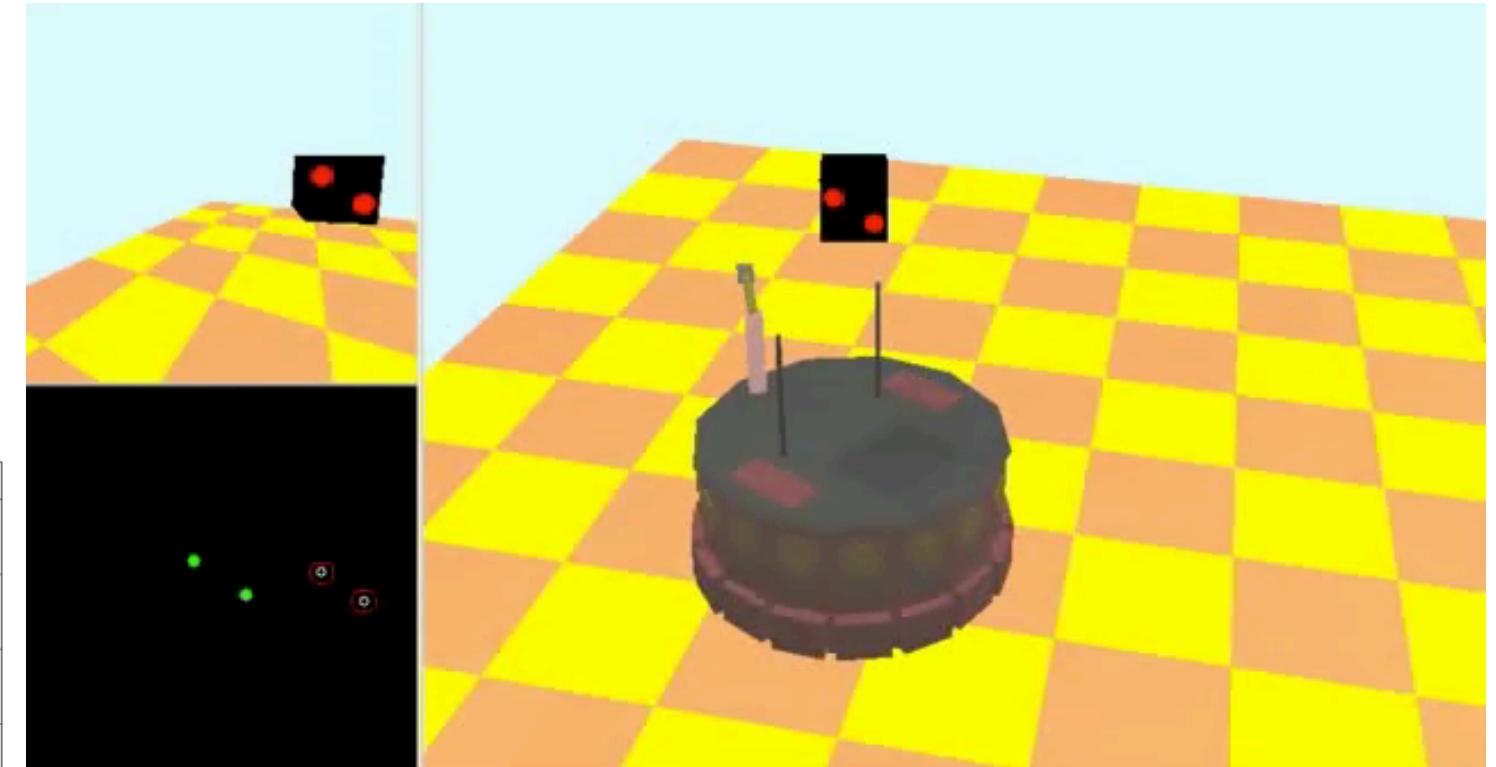
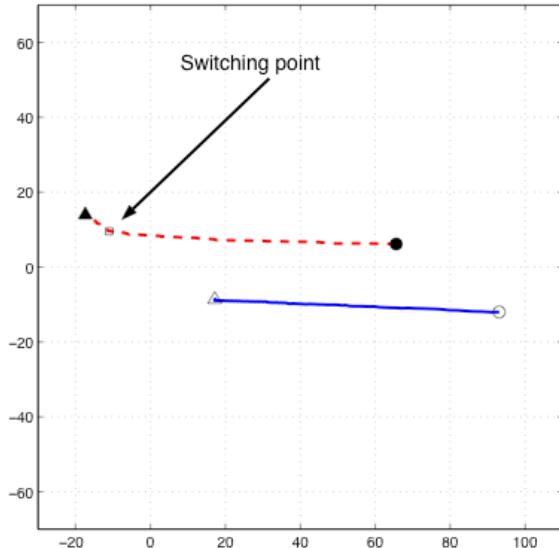
- if the complete case is **not** redundant: singularities are typically met without Task Sequencing, but possibly prevented with TS ...



Simulation with TS scheme

mobile base (unicycle) + polar 2R arm: Image Jacobian is square (4×4)

point feature 1
(regulated in
first phase)
point feature 2
(in second phase)



video

behavior of
the 2 features

- simulation in Webots
- current value of Z is supposed to be known



Experiments

Magellan (unicycle) + pan-tilt camera (same mobility of a polar 2R robot)



Video attachment to ICRA'08 paper

Visual Servoing with Exploitation of Redundancy:
An Experimental Study

A. De Luca

M. Ferri

G. Oriolo

P. Robuffo Giordano

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"

video

- comparison between **Task Priority (TP)** and **Task Sequencing (TS)** schemes
- both can handle singularities (of Image Jacobian) that are possibly encountered
- camera frame rate = 7 Hz



In many practical cases...

- the uncertainty in a number of relevant data may be large
 - focal length λ (**intrinsic** camera parameters)
 - **hand-eye calibration** (**extrinsic** camera parameters)
 - **depth** Z of point features
 -
- one can only compute an **approximation** of the Image Jacobian (both in its **interaction matrix** part, as well as in the **robot Jacobian** part)
- in the closed loop, error dynamics on features becomes
$$\dot{e} = -J \hat{J}^{\#} K e$$
 - **ideal** case: $J J^{\#} = I$ **real** case: $J \hat{J}^{\#} \neq I$
- it is possible to show that a **sufficient condition** for **local** convergence of the error to zero is

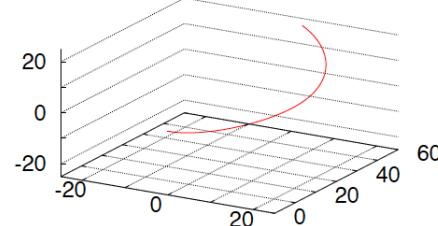
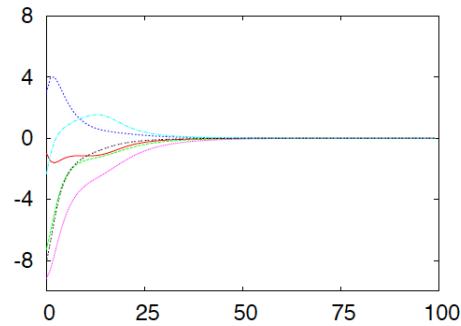
$$J \hat{J}^{\#} > 0$$



Approximate Image Jacobian

- use a constant Image Jacobian $\hat{J}(Z^*)$ that is computed at the desired target s^* (with a known, fixed depth Z^*)

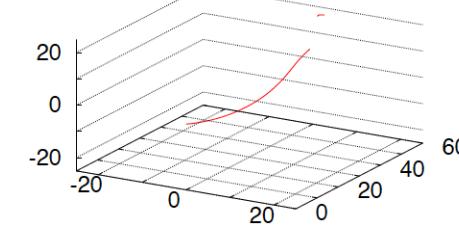
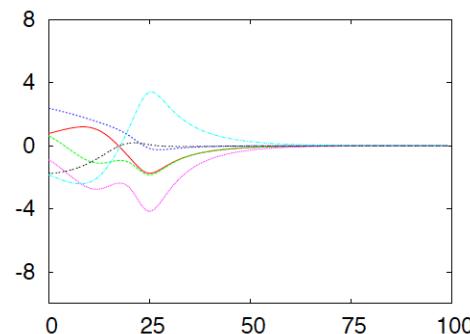
$$\dot{q} = J^\#(Z)Ke$$



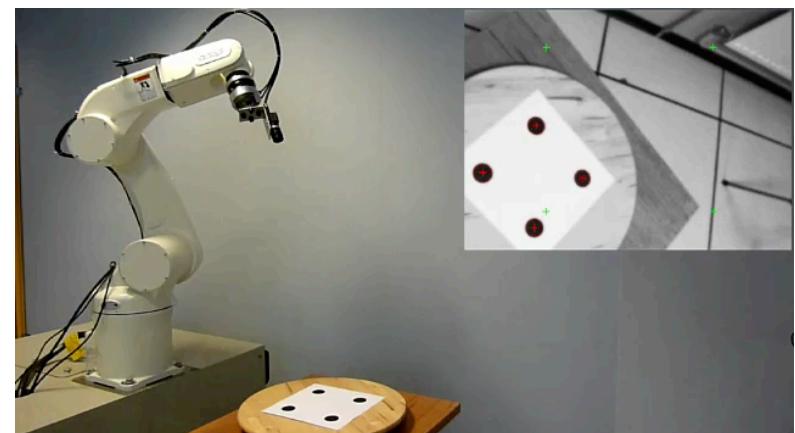
video



$$\dot{q} = \hat{J}^\#(Z^*)Ke$$



video



F. Chaumette, INRIA Rennes



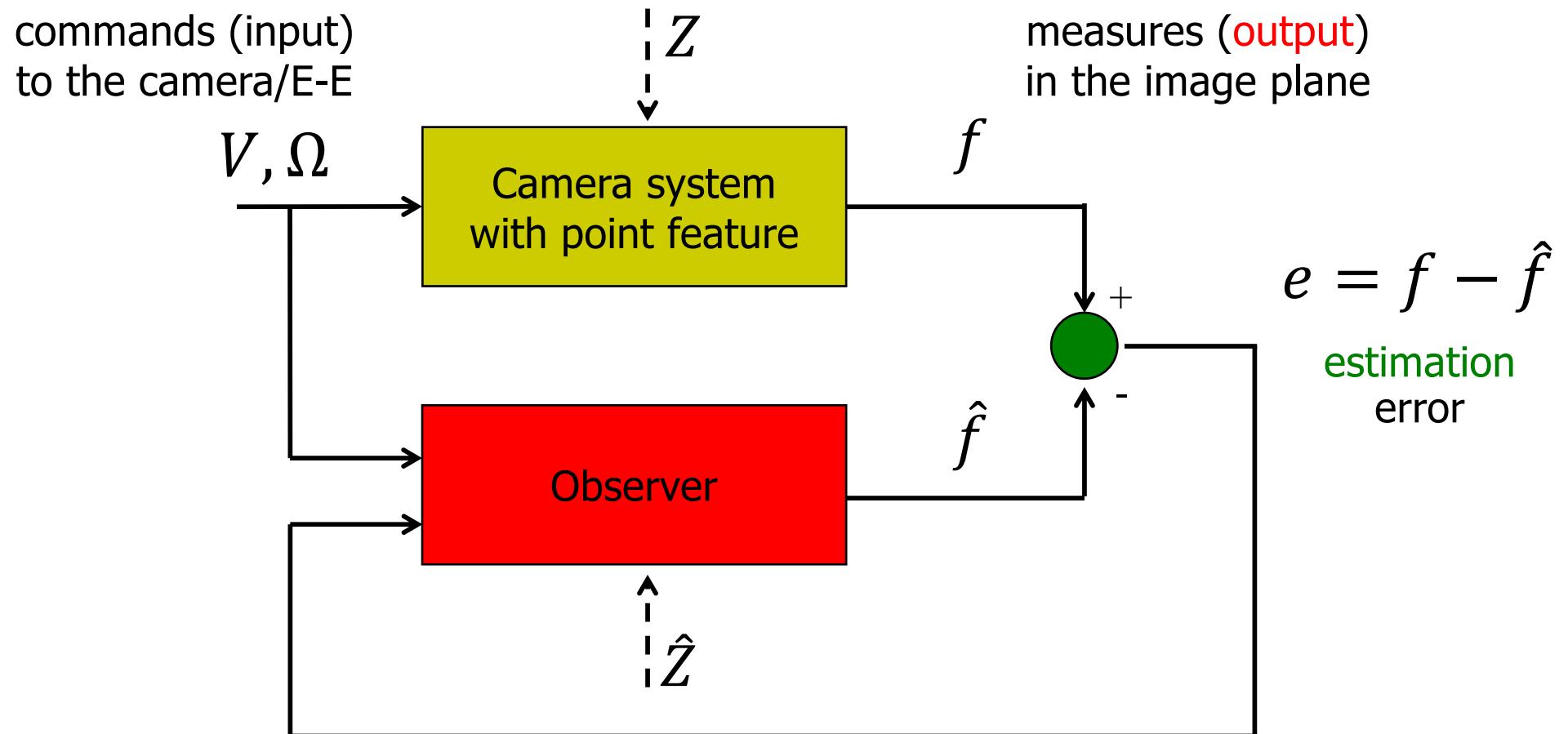
An observer of the depth Z

- it is possible to **estimate on line** the current value (possibly time-varying) of the depth Z , **for each** considered point feature, using a dynamic **observer**
- define $x = [u \ v \ 1/Z]^T$, $\hat{x} = [\hat{u} \ \hat{v} \ 1/\hat{Z}]^T$ as **current state** and **estimated state**, and $y = [u \ v]^T$ as **measured output**
- a (nonlinear) observer of x with input $u_c = [V \ \Omega]^T$
$$\dot{\hat{x}} = \alpha(\hat{x}, y)u_c + \beta(\hat{x}, y, u_c)$$
 guarantees $\lim_{t \rightarrow \infty} \|x(t) - \hat{x}(t)\| = 0$ **provided that**
 - **linear velocity** of the camera is **not** zero
 - the linear velocity vector **is not aligned** with the projection ray of the considered point feature

⇒ these are **persistent excitation** conditions (\sim observability conditions)



Block diagram of the observer





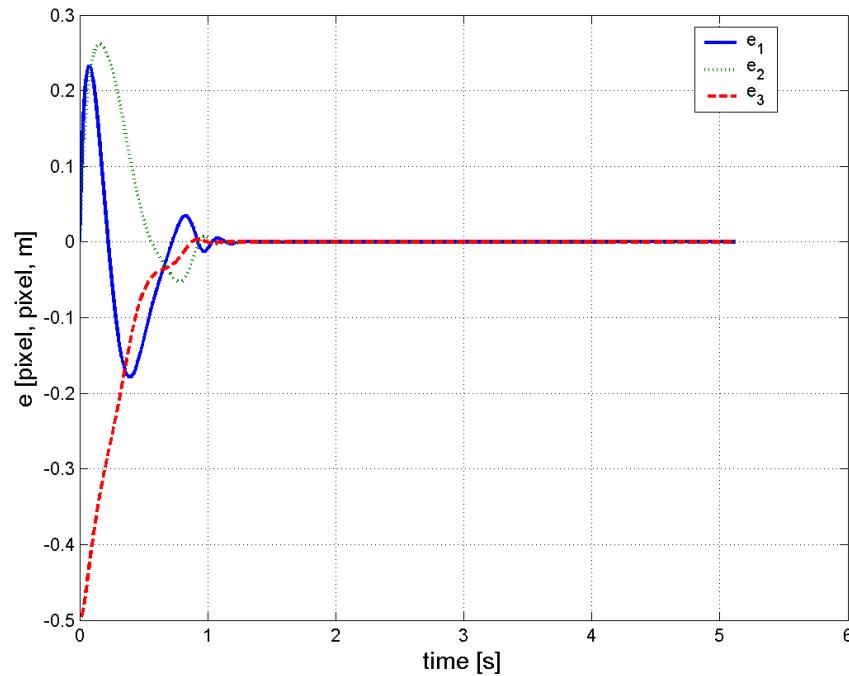
Results on the estimation of Z

real and estimated initial state

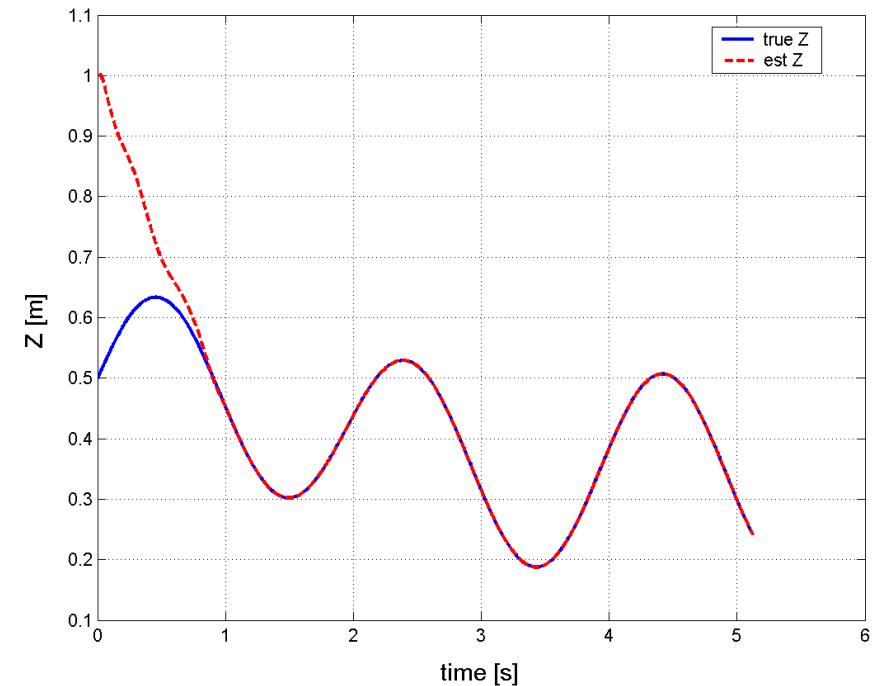
$$x(t_0) = [10 \quad -10 \quad 2]^T$$
$$\hat{x}(t_0) = [10 \quad -10 \quad 1]^T$$

$$v_x(t) = 0.1 \cos 2\pi t$$
$$v_z(t) = 0.5 \cos \pi t$$
$$\omega_x(t) = 0.6 \cos(\pi/2)t$$
$$\omega_z(t) = 1$$

open-loop
commands



estimation errors $e(t)$



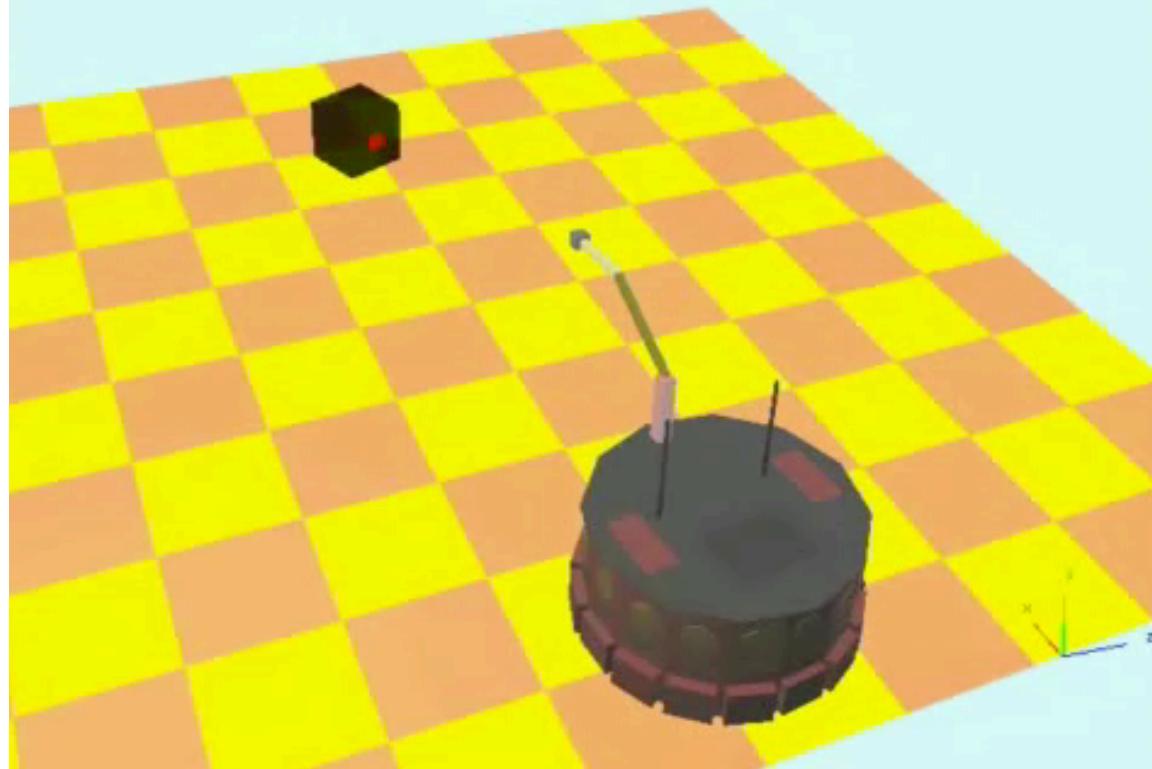
$Z(t)$ and $\hat{Z}(t)$



Simulation of the observer with stepwise displacements of the target

[video](#)

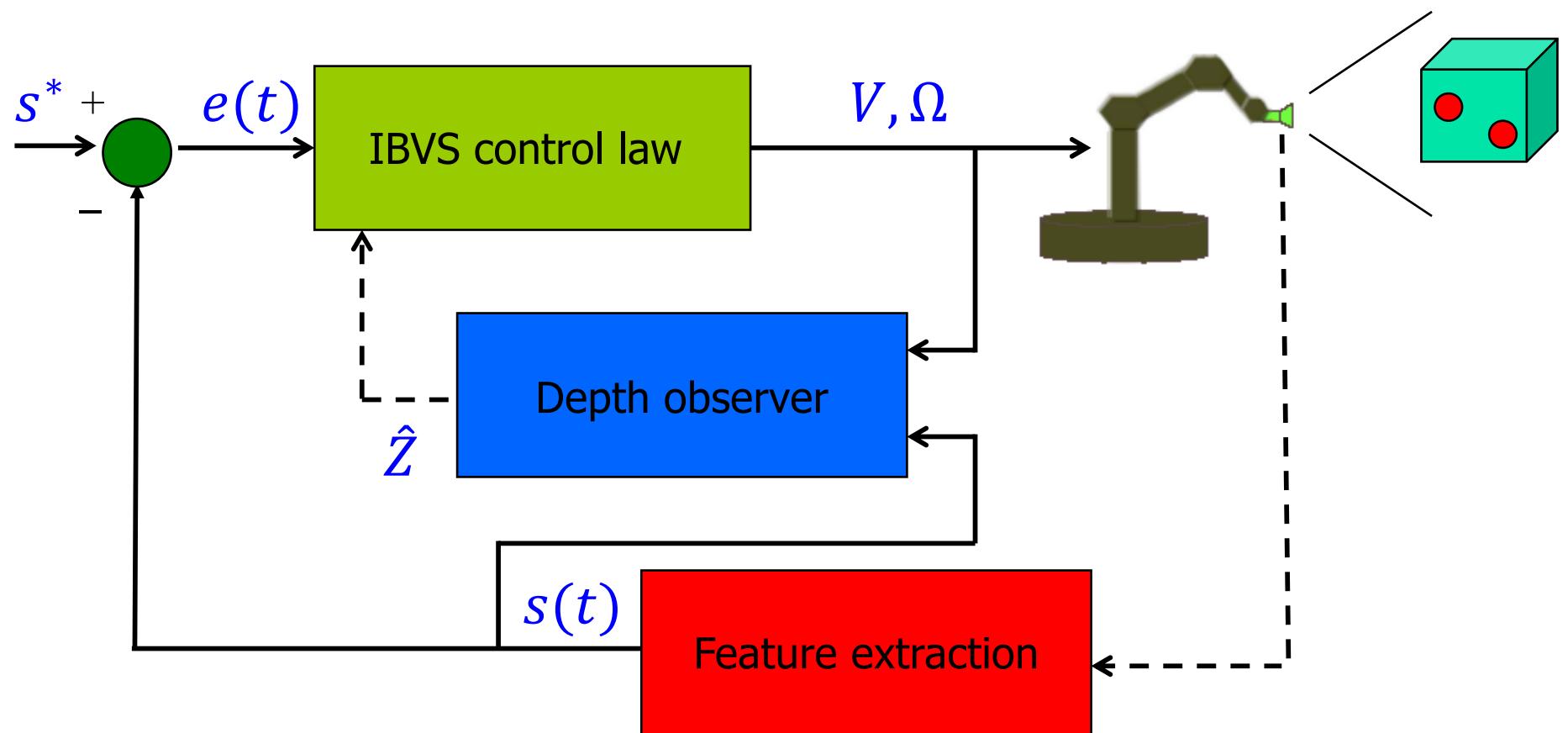
The crosshair represents the estimated value of the depth Z





IBVS control with observer

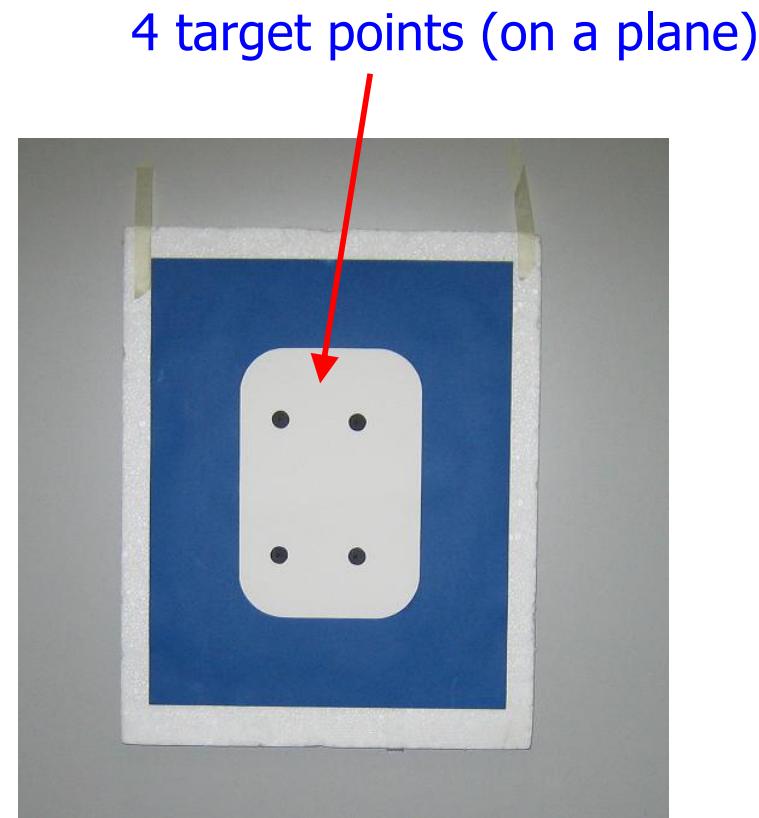
- the depth observer can be easily integrated on line with any IBVS control scheme





Experimental set-up

- visual servoing with fixed camera on a skid-steering mobile robot
- **very rough** eye-robot calibration
- **unknown** depths of target points (**estimated on line**)



a “virtual” 5th feature is also used
as the **average** of the four point features



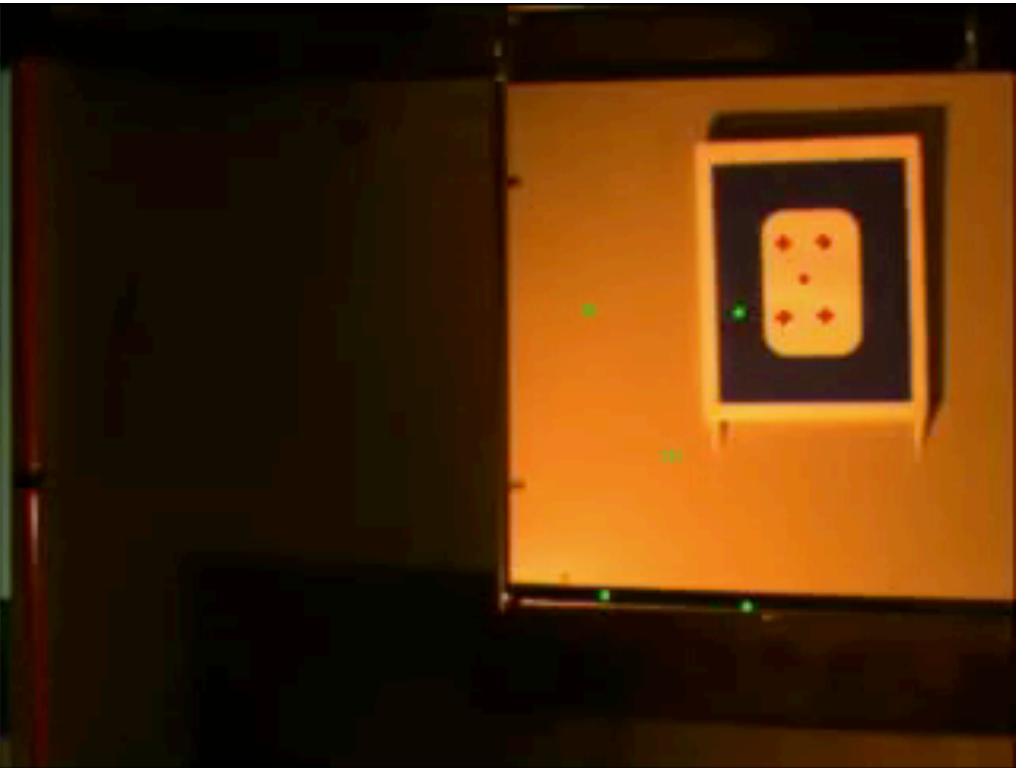
Experiments

- motion of features on the image plane is not perfect...
- the visual **regulation** task is however correctly **realized**

external view



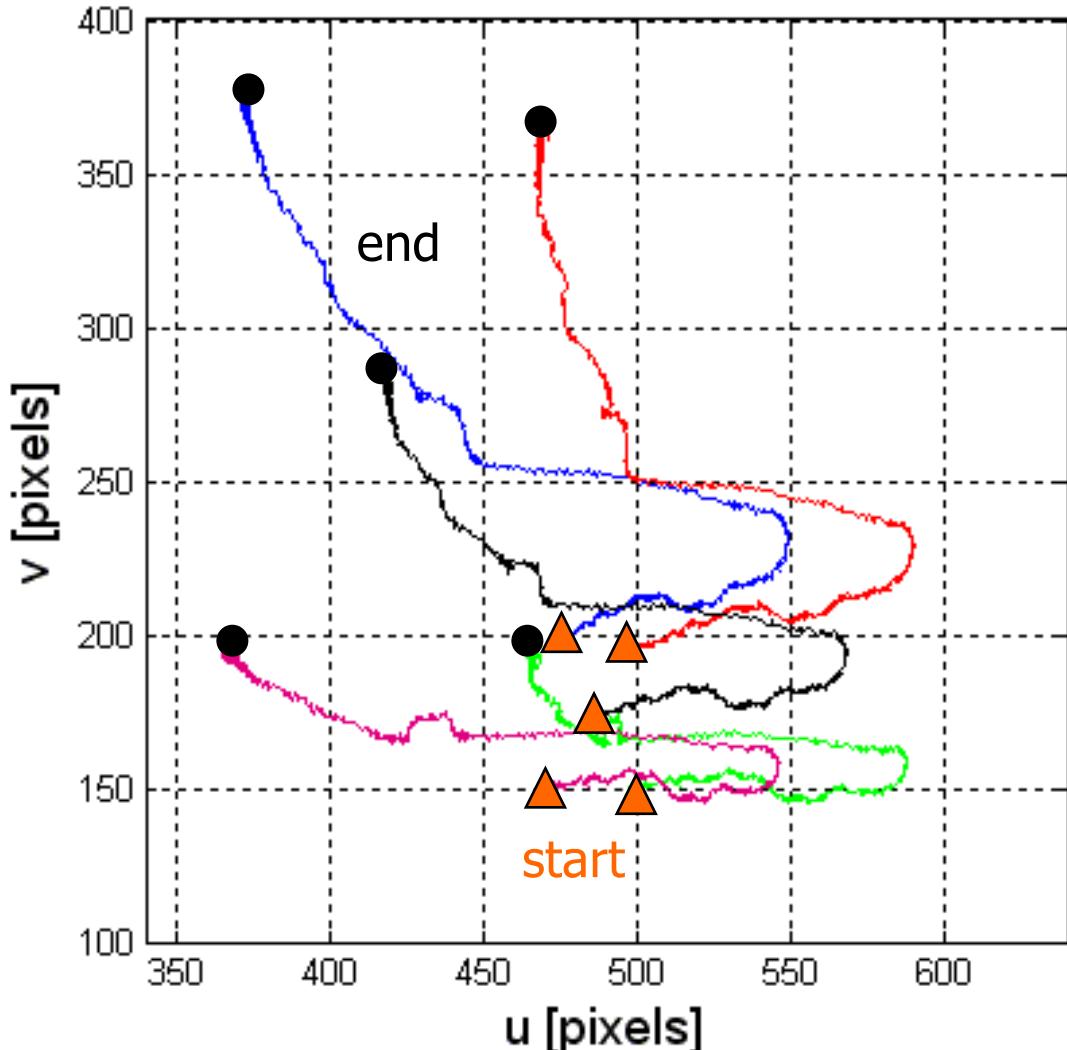
camera view



video (c/o Università di Siena)



Evolution of features



- motion of the 5 features (including the average point) in the image plane
- toward end, motion is \approx linear since the depth observers have already converged to the true values
- computed Image Jacobian is close to the actual one

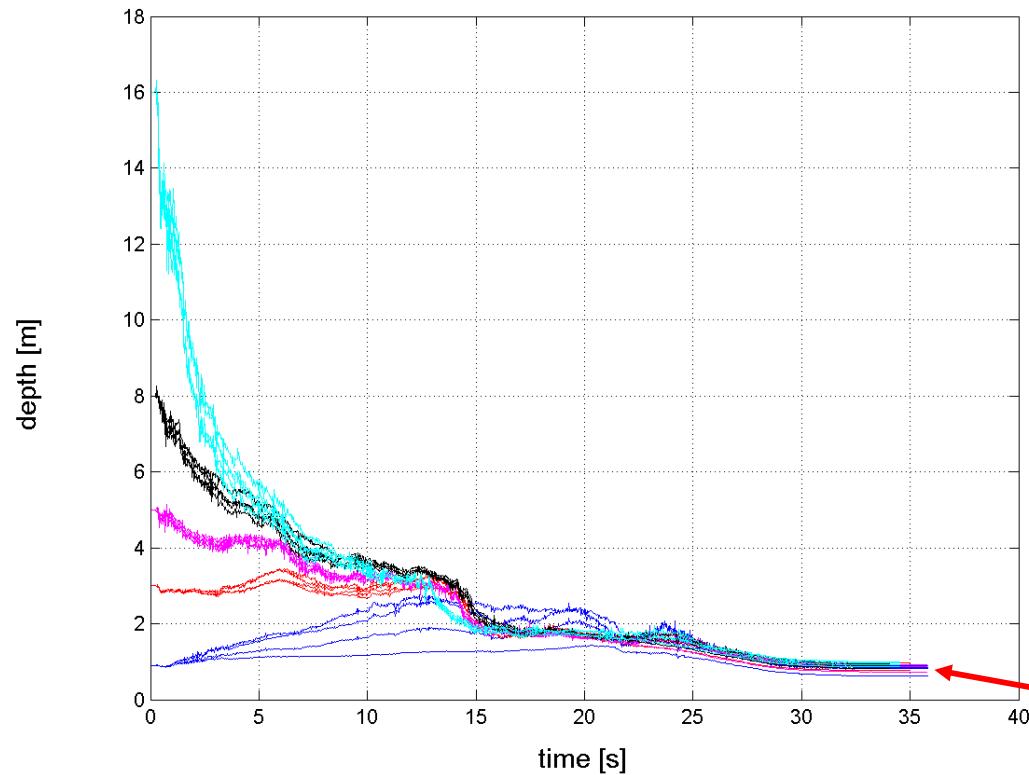
$$\hat{Z} \rightarrow Z \Rightarrow \hat{J} \rightarrow J$$

- “true” initial and final depths
- $$Z(t_0) \cong 4 \text{ m} \quad Z_d \cong 0.9 \text{ m}$$



Experiments with the observer

- the same task was executed with **five different initializations** for the depth observers, ranging between **0.9 m** (= true depth in the **final** desired pose) and **16 m** (much larger than in the true **initial** pose)



the evolutions of the **estimated** depths for the 4 point features

- initial values of depth estimates in the five tests

$$\left\{ \begin{array}{l} \hat{Z}_1(t_0) = 16 \text{ m} \\ \hat{Z}_2(t_0) = 8 \text{ m} \\ \hat{Z}_3(t_0) = 5 \text{ m} \\ \hat{Z}_4(t_0) = 3 \text{ m} \\ \hat{Z}_5(t_0) = 0.9 \text{ m} \end{array} \right.$$

- true depths in **initial** pose

$$Z(t_0) \cong 4 \text{ m}$$

- true depths in **final** pose

$$Z_d \cong 0.9 \text{ m}$$



Visual servoing with Kuka robot set-up

- Kuka KR5 sixx R650 manipulator (6 revolute joints, spherical wrist)
- Point Grey Flea^{©2} CCD camera, eye-in-hand (mounted on a support)
- Kuka KR C2sr low-level controller (RTAI Linux and Sensor Interface)
- image processing and visual servoing on PC (Intel Core2 @2.66GHz)
- high-level control data exchange every **12ms** (via UDP protocol)



@ DIAG Robotics Laboratory

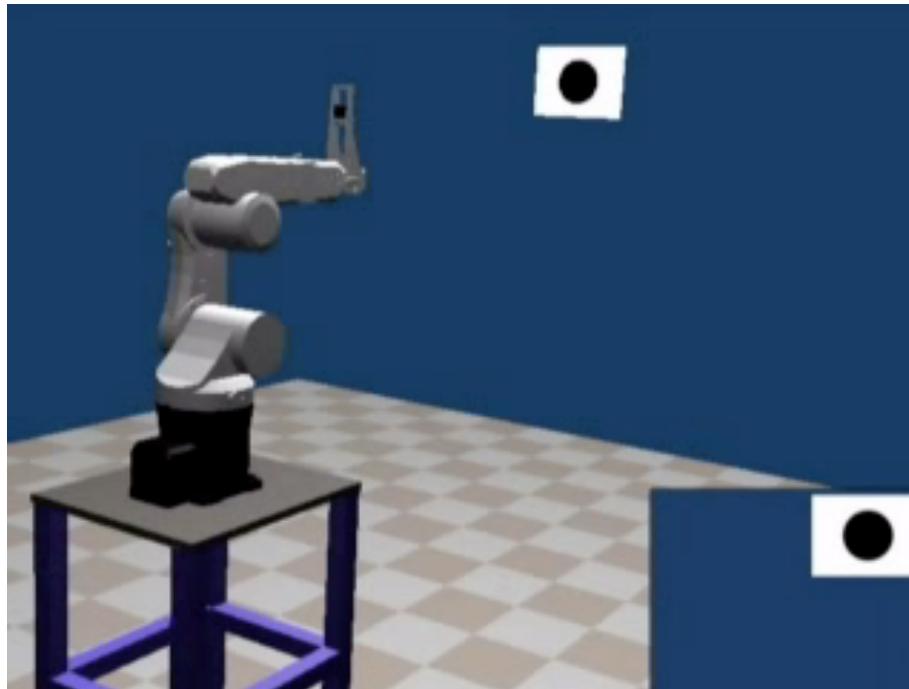


Visual servoing with Kuka robot

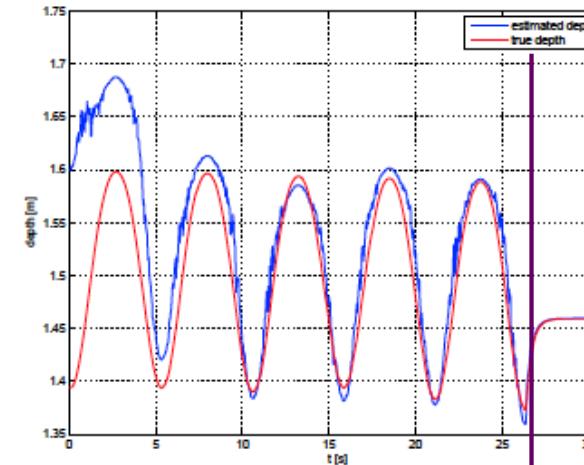
simulation and experiment with the observer

- depth estimation for a **fixed target** (single point feature)
- **simulation** using Webots first, then **experimental** validation
- **sinusoidal motion** of four robot joints so as to provide sufficient excitation

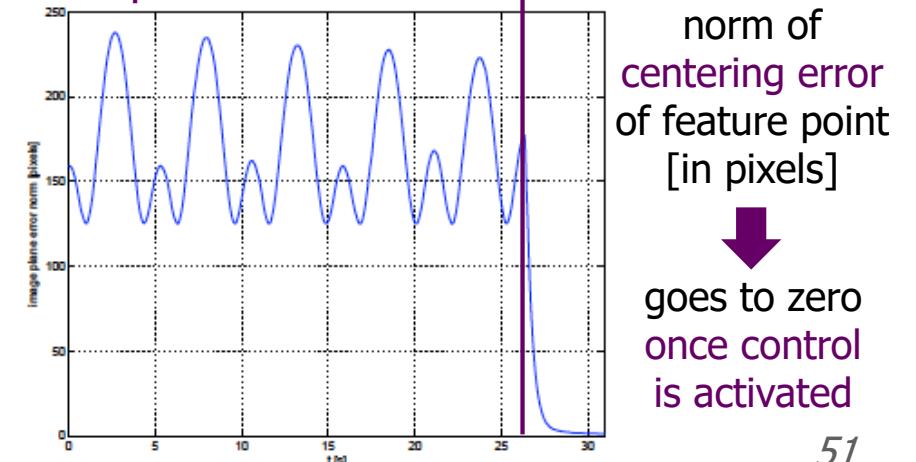
video



Webots simulation



Kuka experiment





Visual servoing with Kuka robot tracking experiment

- tracking a (slowly) time-varying target by visual servoing, including the depth observer (after convergence)



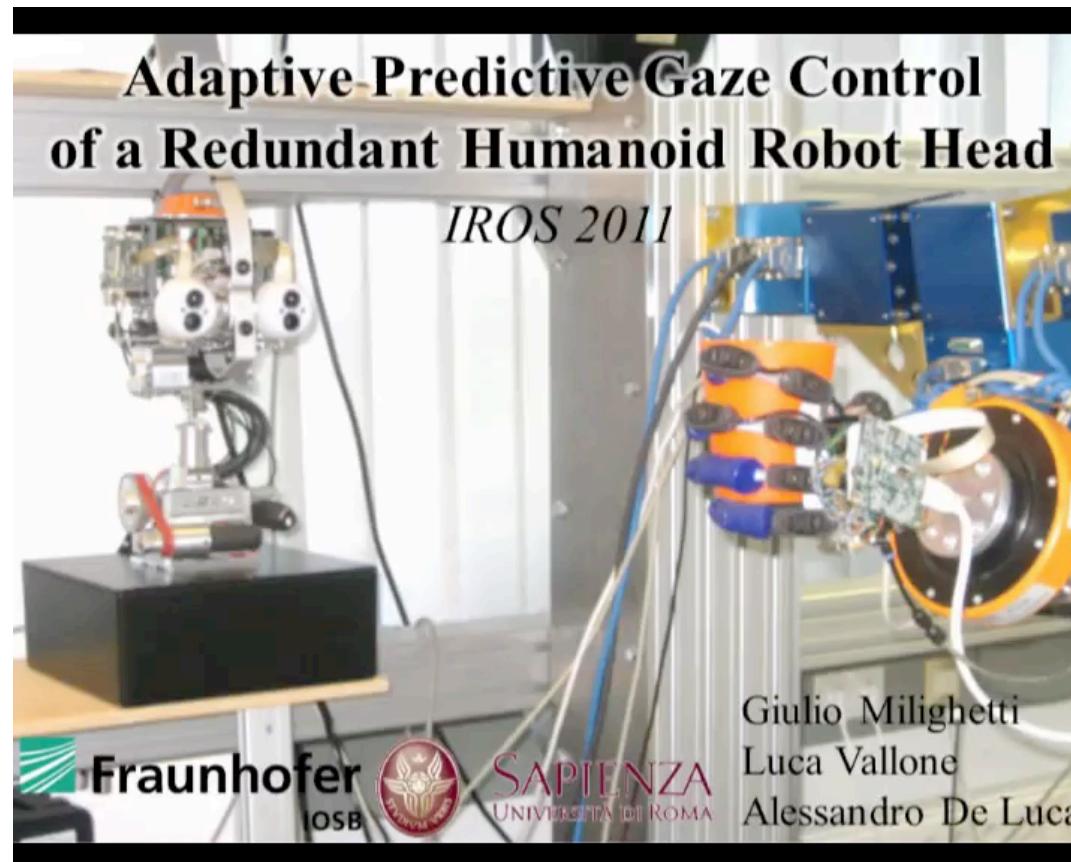
video



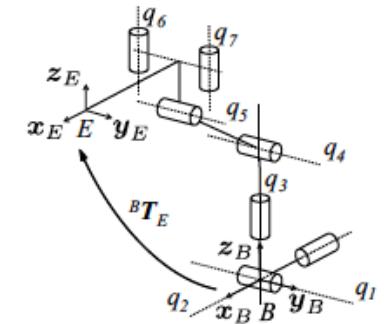
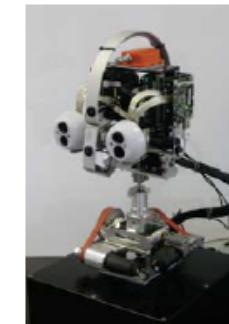
Gazing with humanoid head

stereo vision experiment

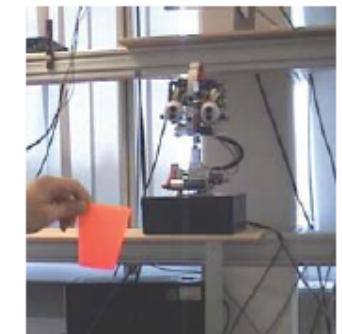
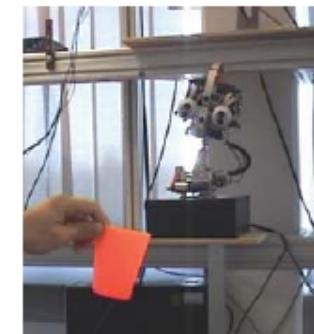
- gazing with a robotized head (7 joints, with $n = 6$ independently controlled) at a moving target (visual task dimension $m = 2$), using redundancy (to keep a better posture and avoid joint limits) and a predictive feedforward



video (c/o Fraunhofer IOSB, Karlsruhe)



$$\dot{\mathbf{q}} = \mathbf{J}_W^\dagger(\mathbf{q}) \left(\dot{\theta}_{fb} + \dot{\theta}_{ff} \right) - k_0 \left(\mathbf{I} - \mathbf{J}_W^\dagger(\mathbf{q})\mathbf{J}(\mathbf{q}) \right) \nabla_{\mathbf{q}} H_0(\mathbf{q})$$



final head posture
without and with self-motions



Robotics 2

Detection and isolation of robot actuator faults

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Fault diagnosis problems - 1

- in the diagnosis of faults possibly affecting a (nonlinear) dynamic system various problems can be formulated
- **Fault Detection**
 - recognize that the malfunctioning of the (controlled) system is due to the occurrence of a fault (or not proper behavior) affecting some physical or functional component of the system
- **Fault Isolation**
 - discriminate which particular fault f has occurred out of a (large) class of potential ones, by distinguishing it from any other fault and from the effects of disturbances possibly acting on the system
- **Fault Identification**
 - determine the time profile (and/or class type) of the isolated fault f
- **Fault Accommodation**
 - modify the control law so as to compensate for the effects of the detected and isolated fault (possibly also identified)



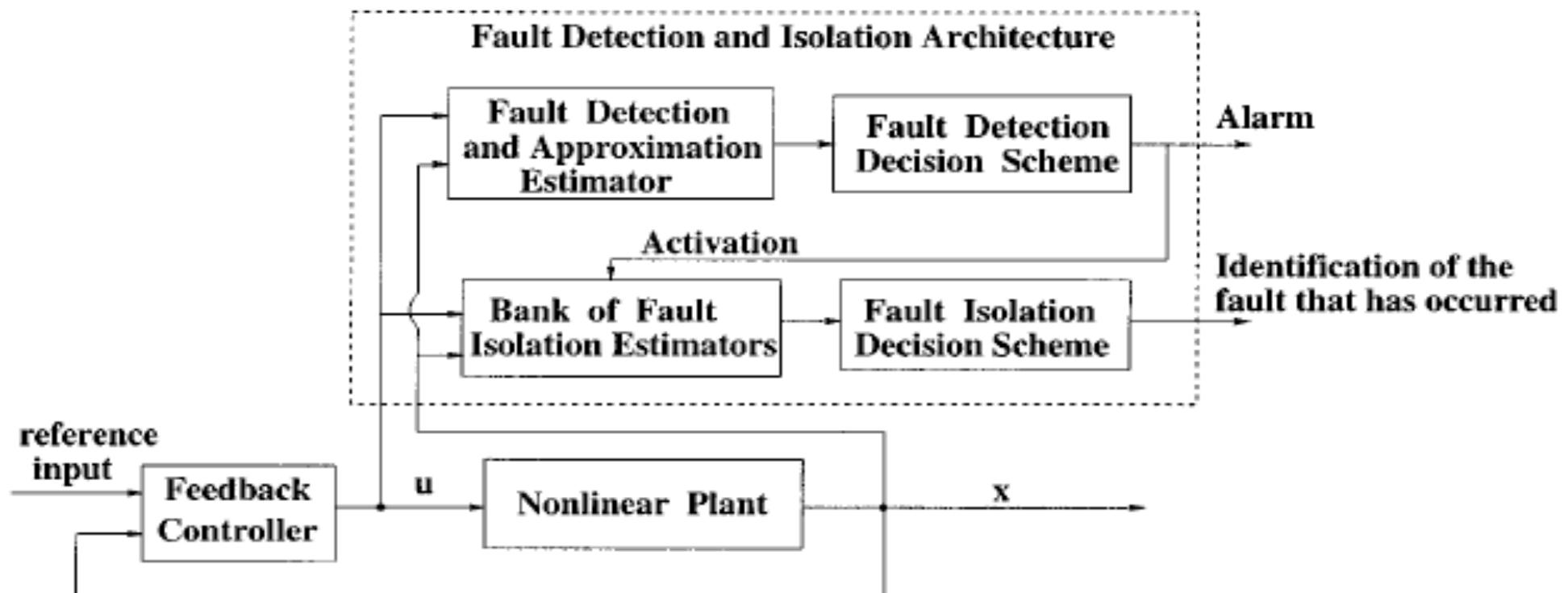
Fault diagnosis problems - 2

- FDI solution (simultaneous detection and isolation)
 - definition of an auxiliary dynamic system (**Residual Generator**) whose **output** will depend only on the presence of the fault f to be detected and isolated (and **not** on any other fault or disturbance) and will converge asymptotically to zero when $f \equiv 0$ (**stability**)
 - in case of many potential faults, each component r_i of the **vector r of residuals** will depend on one and only one associated fault f_i (possibly reproducing approximately its time behavior)
 - many of the FDI schemes are **model-based**: they use a nominal (fault- and disturbance-free) dynamic model of the system
- Fault Tolerant Control
 - **passive**: control scheme that is intrinsically robust to uncertainties and/or faults (typically having only moderate/limited effects)
 - **active**: control scheme involving a reconfiguration after FDI (with guaranteed performance for the faulted system)



Typical FDI architecture

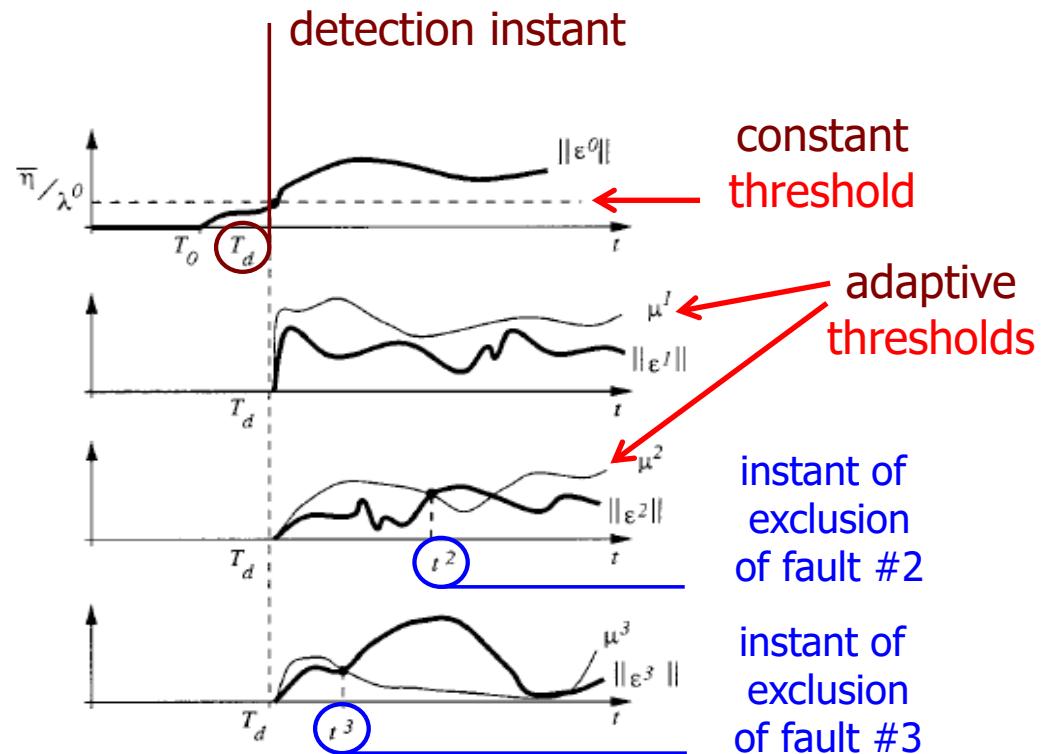
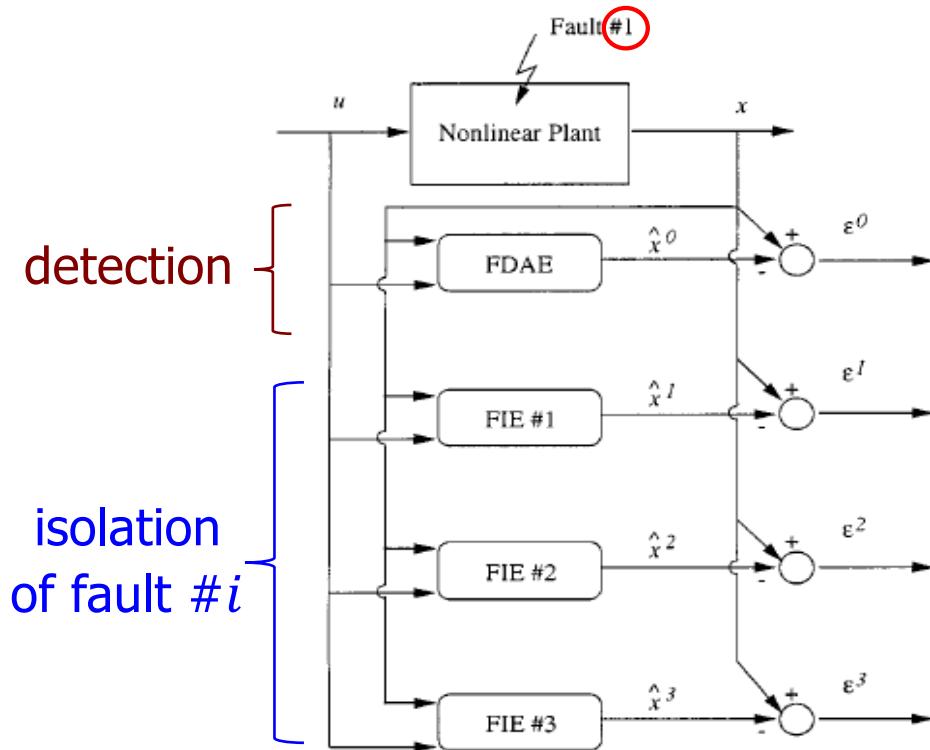
- bank of $n + 1$ (model-based) estimators
 - 1 for **detection** of a faulty condition
 - n for **isolation** of the specific (in general, **modeled**) fault





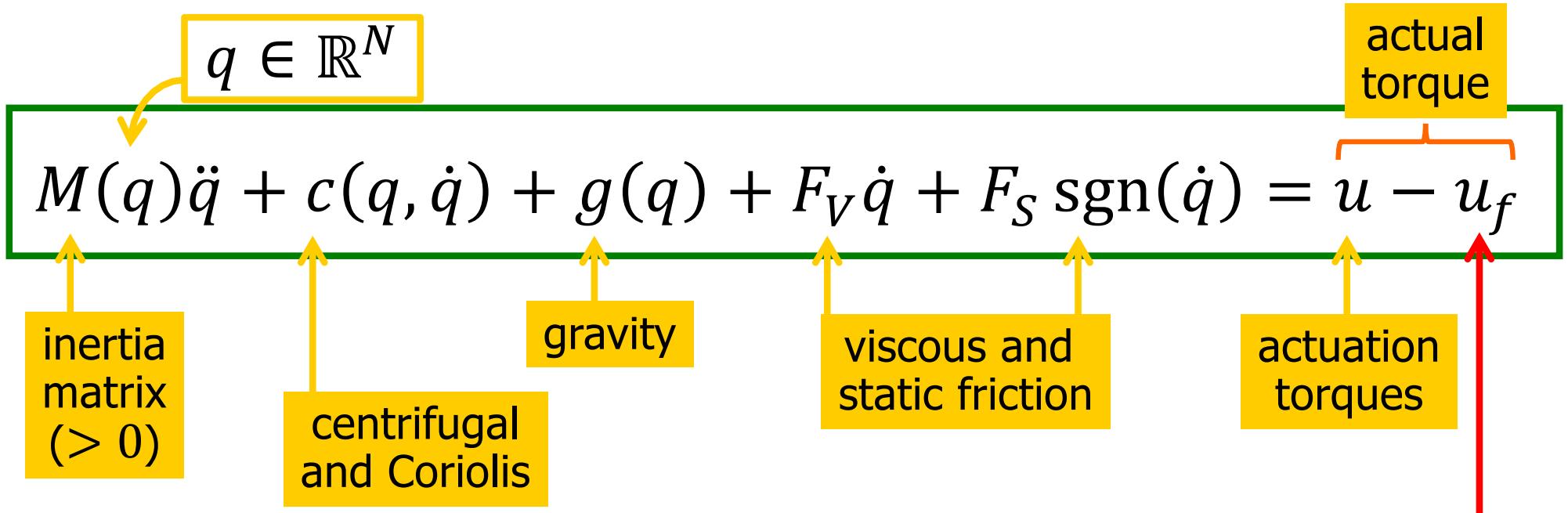
Some terminology

- fault types
 - instantaneous (abrupt), incipient (slow), intermittent, concurrent
- thresholds for detection/isolation (also adaptive)
 - delay times (w.r.t. the instant T_0 of fault start) vs. false alarms





Actuator faults in robots



vector of actuation faults (even concurrent on more axes)

- total fault $u_{f,i} = u_i$
- partial fault $u_{f,i} = \varepsilon u_i$ ($0 < \varepsilon < 1$)
- saturation $u_{f,i} = u_i - \operatorname{sgn}(u_i) u_{i,max}$
- bias $u_{f,i} = b_i$ Ex: ??
- block $u_{f,i} = \dots$
- ... any type!



Working assumptions

- signals and measurements available
 - the commanded input torque u , but obviously **not u_f** ...
 - a measure of the **full state** (q, \dot{q}) is available
 - can be relaxed: in practice, with an **estimate** of joint velocities
 - no further sensors are anyway necessary ("**sensorless**")
- the **robot dynamic model is known**
 - in the absence of faults, and neglecting disturbances
 - **no pre-specified model or type of faults** is needed
- **no dependence on/request of a specific input $u(t)$**
 - can be anything (open loop, linear or nonlinear feedback)
- **no dependence on/request of a specific motion $q_d(t)$**



Generalized momentum

$$p = M(q)\dot{q}$$

with associated dynamic equation

$$\dot{p} = u - u_f - \alpha(q, \dot{q})$$

decoupled components
relative to the single fault inputs

exploiting structure
of centrifugal and
Coriolis terms

$$\alpha_i = -\frac{1}{2}\dot{q}^T \frac{\partial M(q)}{\partial q_i} \dot{q} + g_i(q) + F_{V,i}\dot{q}_i + F_{S,i} \operatorname{sgn}(\dot{q}_i)$$

scalar expressions, for $i = 1, \dots, N$



FDI solution

- definition of a **vector of residuals**

$$r = K \left[\int (u - \alpha(q, \dot{q}) - r) dt - p \right]$$

$K > 0$
diagonal

- no need to compute joint accelerations nor to invert the robot inertia matrix $M(q)$
- with perfect model knowledge, the dynamics of r is

N **decoupled** filters,
with unitary gains and
time constants $\tau_i = 1/k_i$

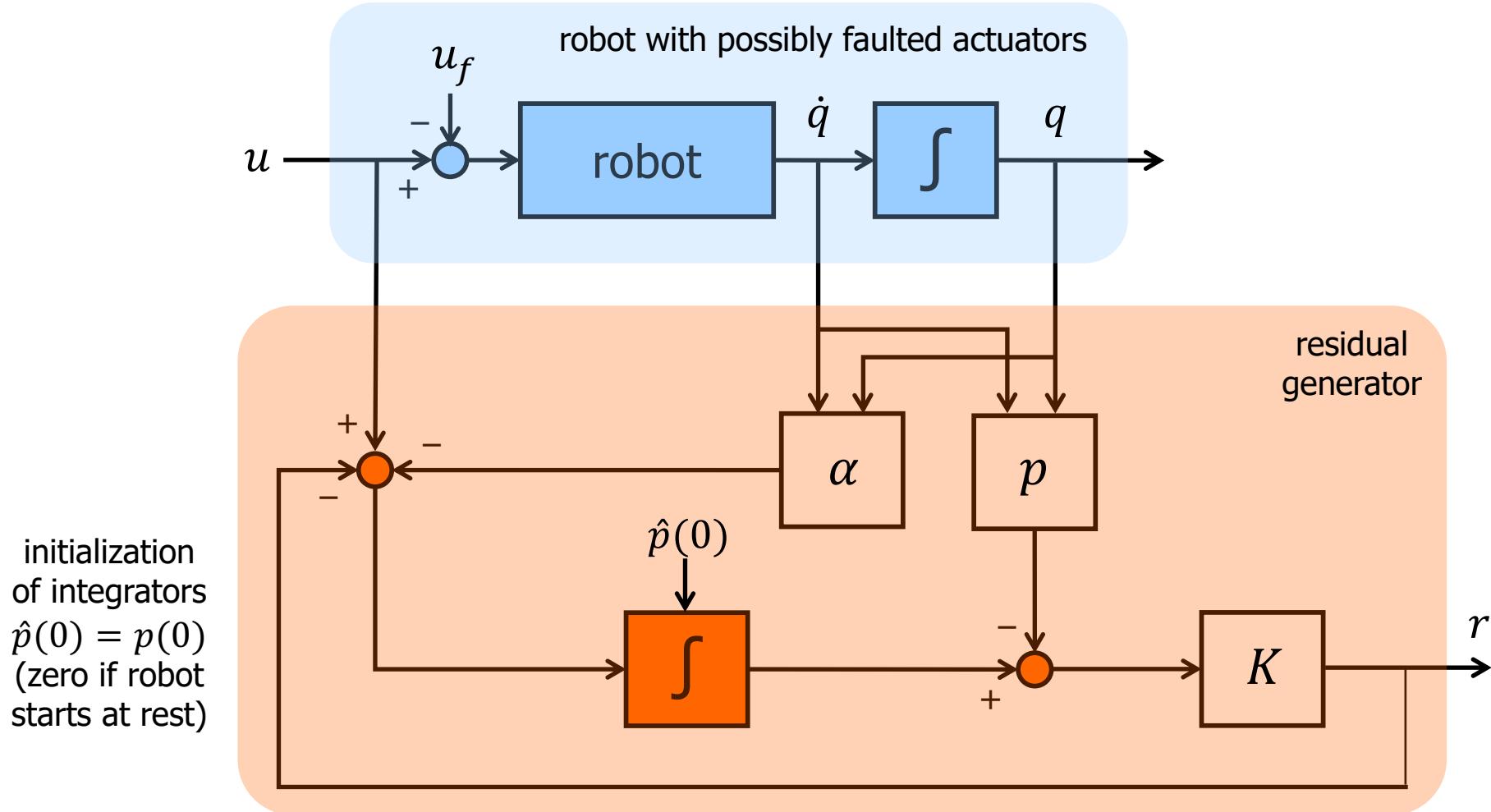
$$\dot{r} = -Kr + Ku_f$$

in the Laplace domain
$$\frac{r_i(s)}{u_{f,i}(s)} = \frac{k_i}{s + k_i} = \frac{1}{1 + \tau_i s}$$

for sufficiently large K , r reproduces the time behavior of u_f



Block diagram of the residual generator



$$r = K \left[\int (u - \alpha(q, \dot{q}) - r) dt - p \right]$$



Residual generator as “disturbance observer”

from the
block diagram...

$$\begin{aligned}\dot{\hat{p}} &= u - \alpha(q, \dot{q}) + K(p - \hat{p}) \\ r &= K(\hat{p} - p)\end{aligned}$$



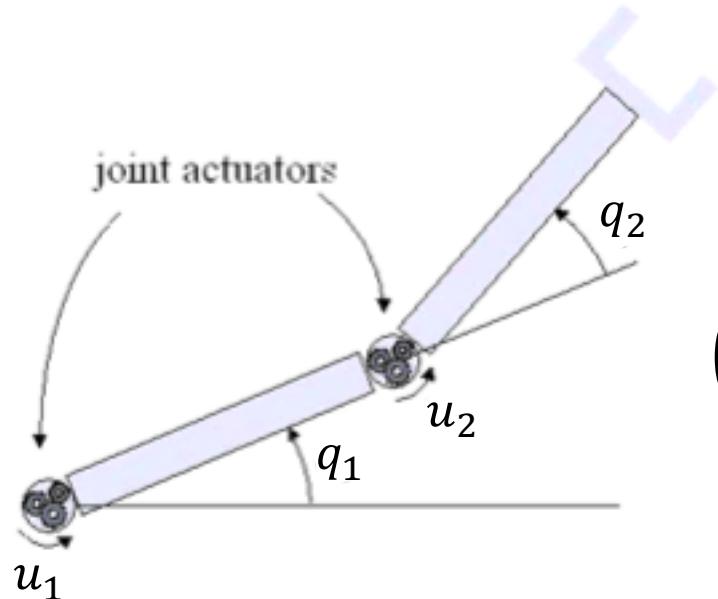
dynamic observer of the unknown actuation faults
($r \approx \rightarrow u_f$ = external disturbances)
with **linear** error dynamics (for constant u_f)

$$\begin{aligned}e_{obs} &= u_f - r \quad \rightarrow \quad \dot{e}_{obs} = \dot{u}_f - \dot{r} = \dot{u}_f - K(\dot{\hat{p}} - \dot{p}) \\ &= \dot{u}_f - K((u - \alpha - r) - (u - \alpha - u_f)) \\ &= \dot{u}_f - K(u_f - r) = \dot{u}_f - K e_{obs} \cong -K e_{obs}\end{aligned}$$



A worked-out example

- planar 2R robot under gravity



dynamic model (without friction)

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

$\overbrace{\quad\quad\quad}^{= S(q, \dot{q})\dot{q}}$

$$\begin{pmatrix} a_1 + 2a_2c_2 & a_3 + a_2c_2 \\ a_3 + a_2c_2 & a_3 \end{pmatrix} \begin{pmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{pmatrix} + \begin{pmatrix} -a_2(2\dot{q}_1 + \dot{q}_2)\dot{q}_2 s_2 \\ a_2\dot{q}_1^2 s_2 \end{pmatrix}$$

$$+ \begin{pmatrix} a_4c_1 + a_5c_{12} \\ a_5c_{12} \end{pmatrix} = \begin{pmatrix} u_1 - u_{f,1} \\ u_2 - u_{f,2} \end{pmatrix}$$

computation of the residual vector

$$r = K \left[\int (u - \alpha(q, \dot{q}) - r) dt - p \right]$$

$$p = M(q)\dot{q}$$

$$\alpha_1 = g_1(q) = a_4c_1 + a_5c_{12}$$

$$\alpha_2 = -\frac{1}{2} \dot{q}^T \frac{\partial M(q)}{\partial q_2} \dot{q} + g_2(q)$$

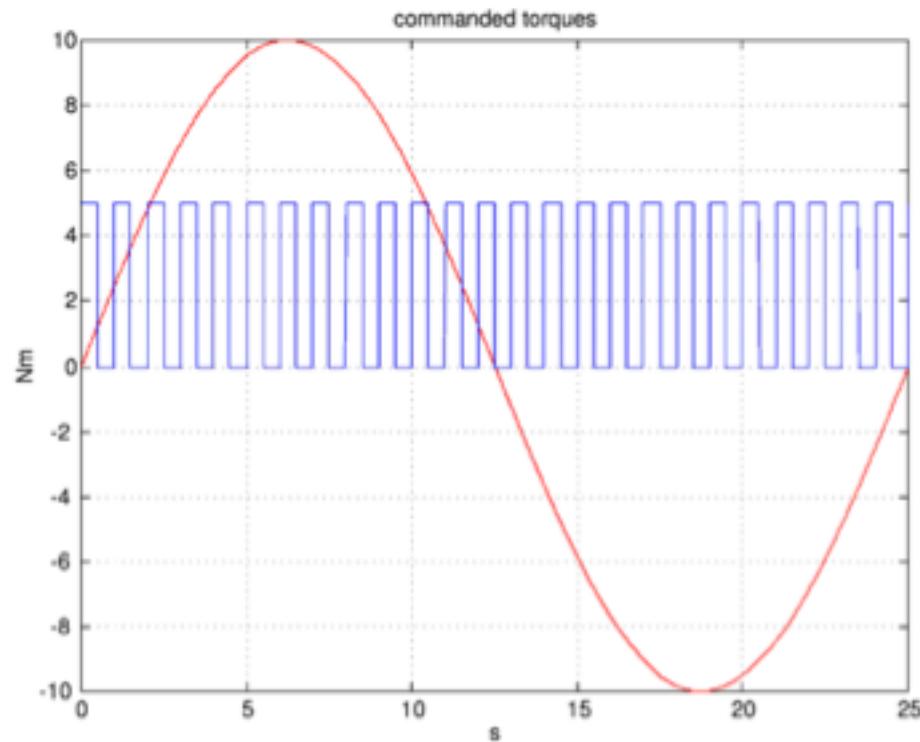
$$= a_2(\dot{q}_1 + \dot{q}_2)\dot{q}_1 s_2 + a_5c_{12}$$



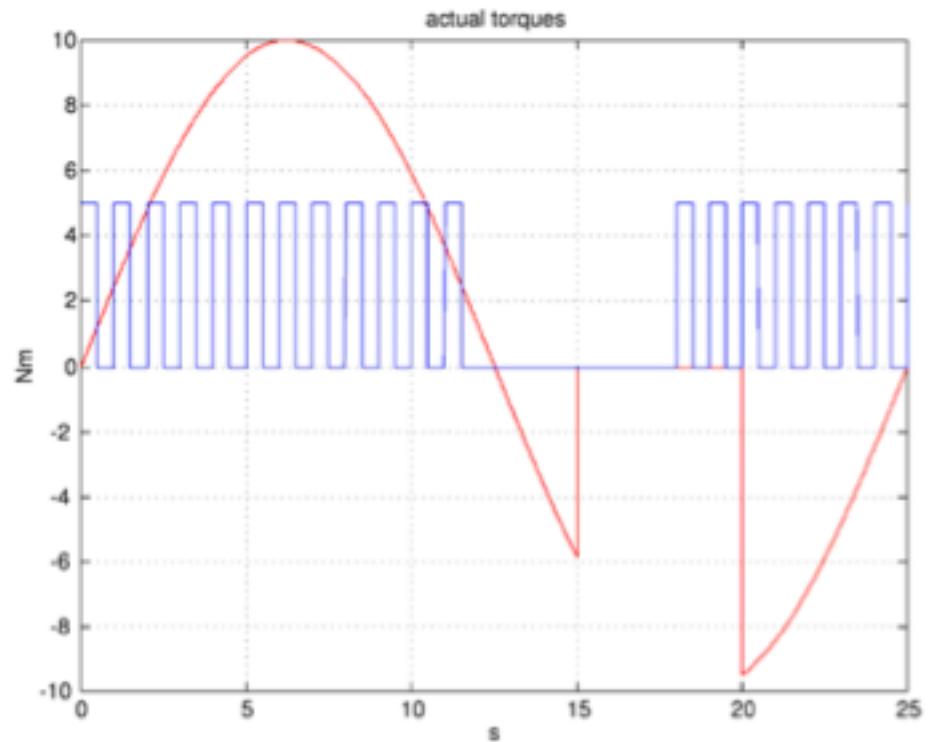
Faults on both actuators

(total, intermittent, concurrent)

commanded torques (in open loop)



actual (faulted) torques



= first joint (fault for $t \in [15 \div 20]$ sec)



= second joint (fault for $t \in [12 \div 18]$ sec)

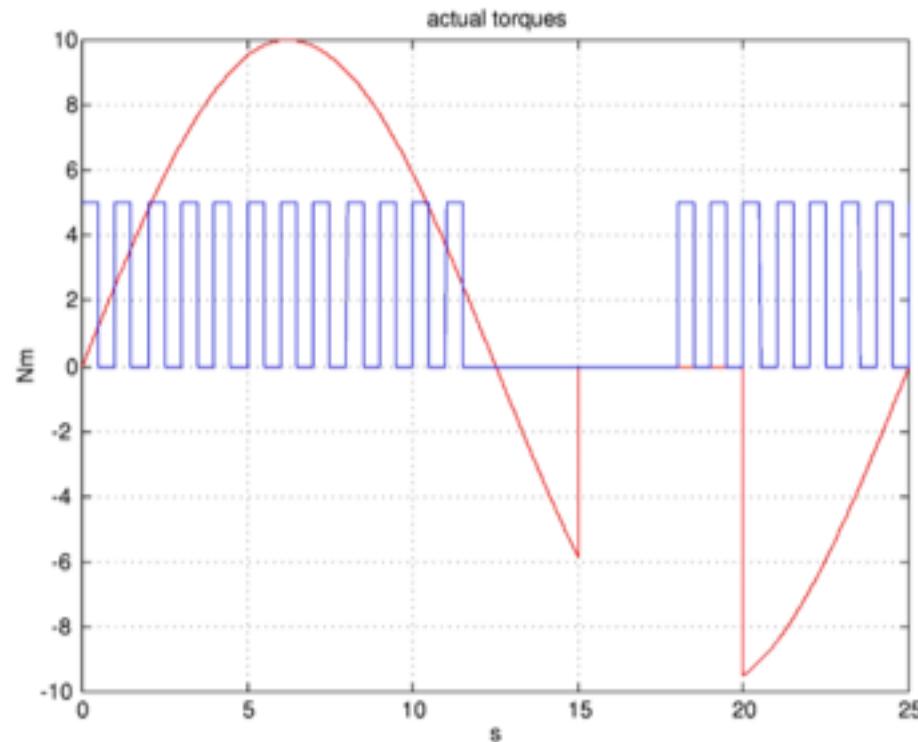


time interval of
fault **concurrence**
 $t \in [15 \div 18]$ sec



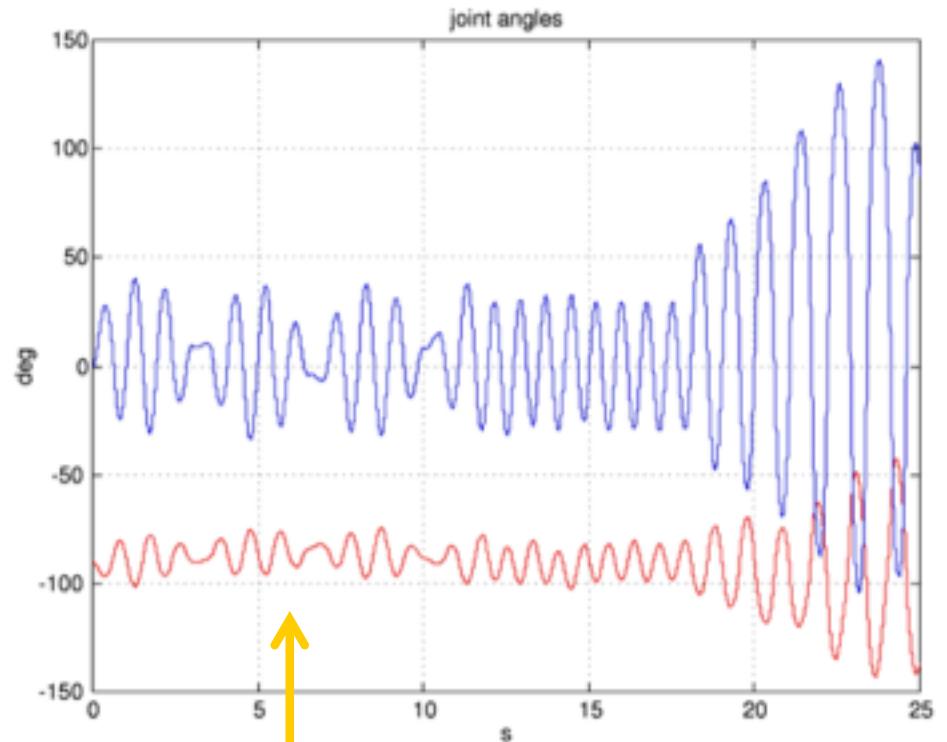
First simulation

actual torques (to the robot)



- = first joint
- = second joint

(measured) joint positions

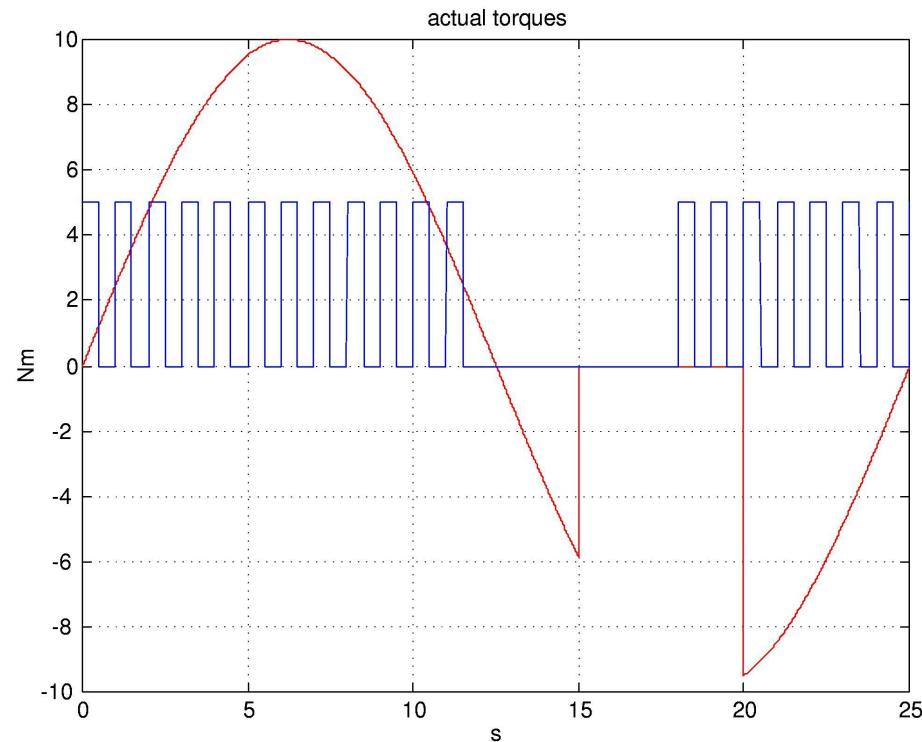


no clear evidence of faults in the dynamic evolution of the system!



First simulation – FDI

actual torques (to the robot)

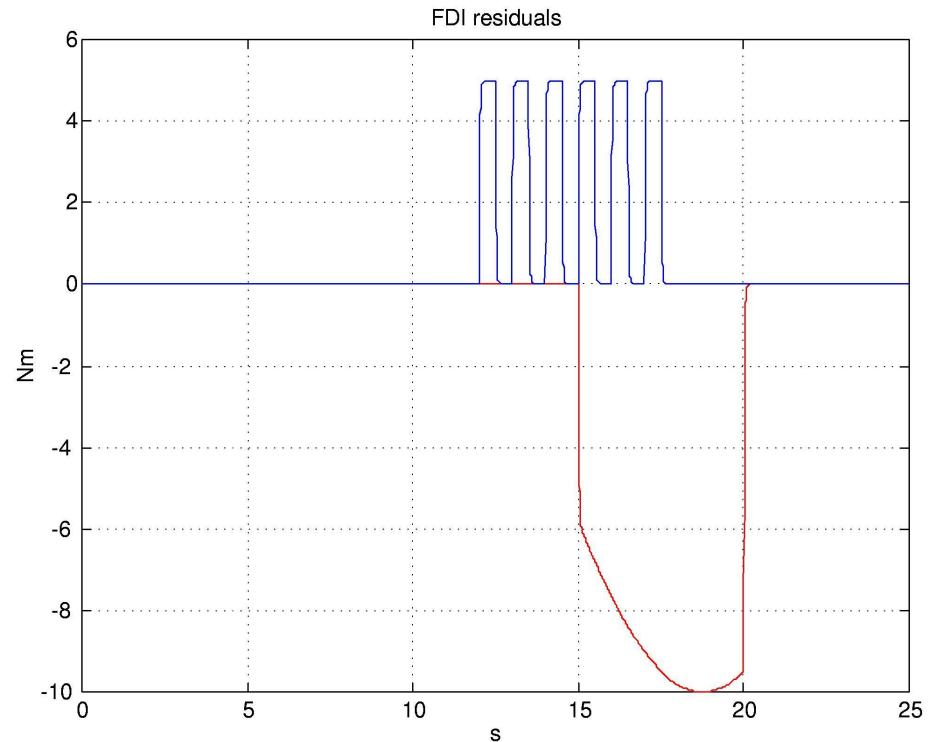


— = first joint

— = second joint

$$K = \text{diag}\{50, 50\}$$

residuals



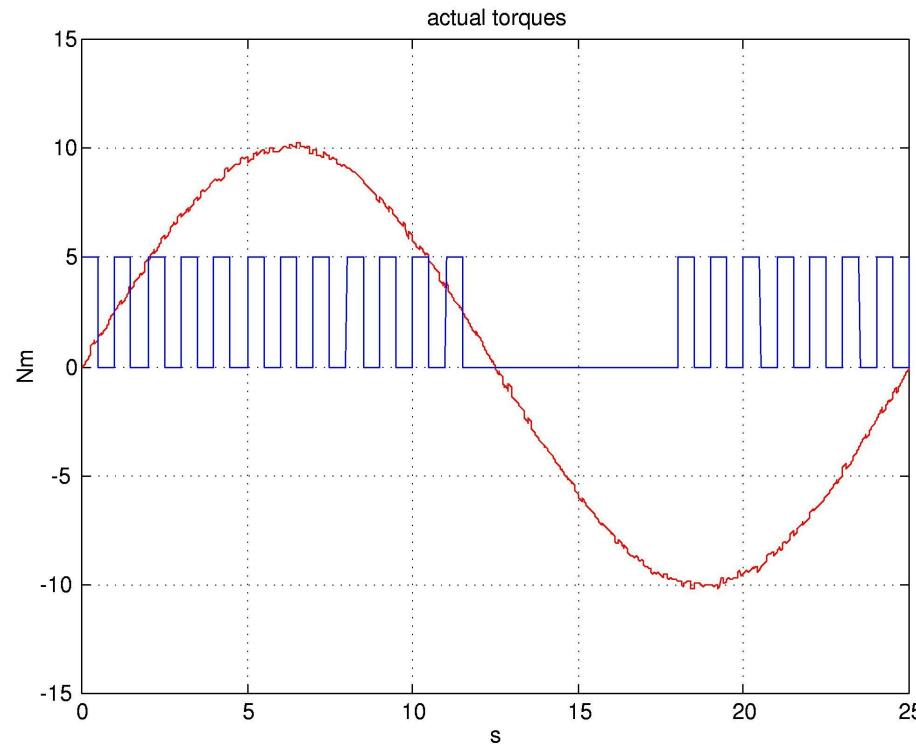
residuals reconstruct the
“missing” parts of the torques
(identification property!)



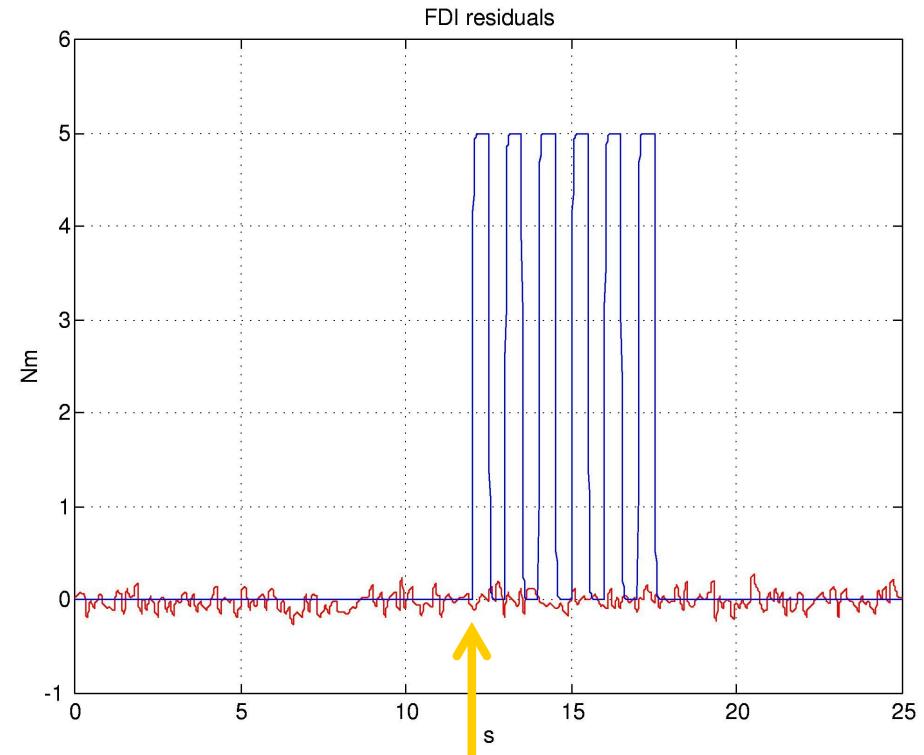
Second simulation – FDI

(total fault on second actuator, added noise on first channel)

actual torques (to the robot)



residuals



— = first joint

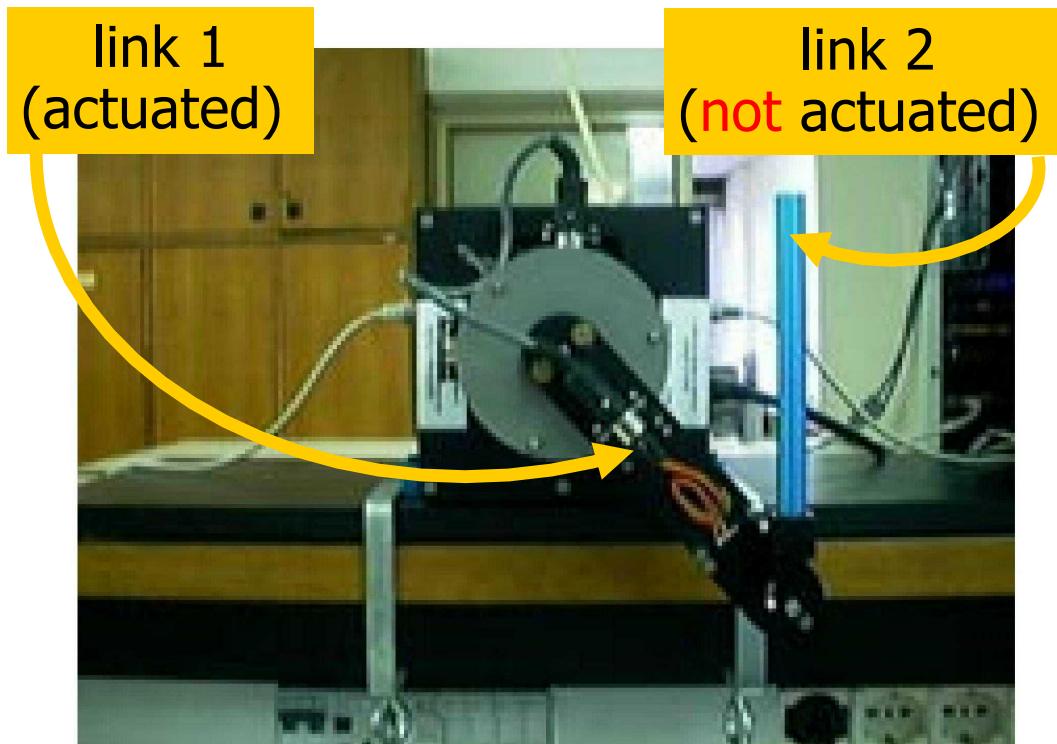
— = second joint (fault for $t \in [12 \div 18]$ sec)

residual r_1 is not affected by faulty actuation, while residual r_2 is not affected by the disturbance on first channel (decoupling property)

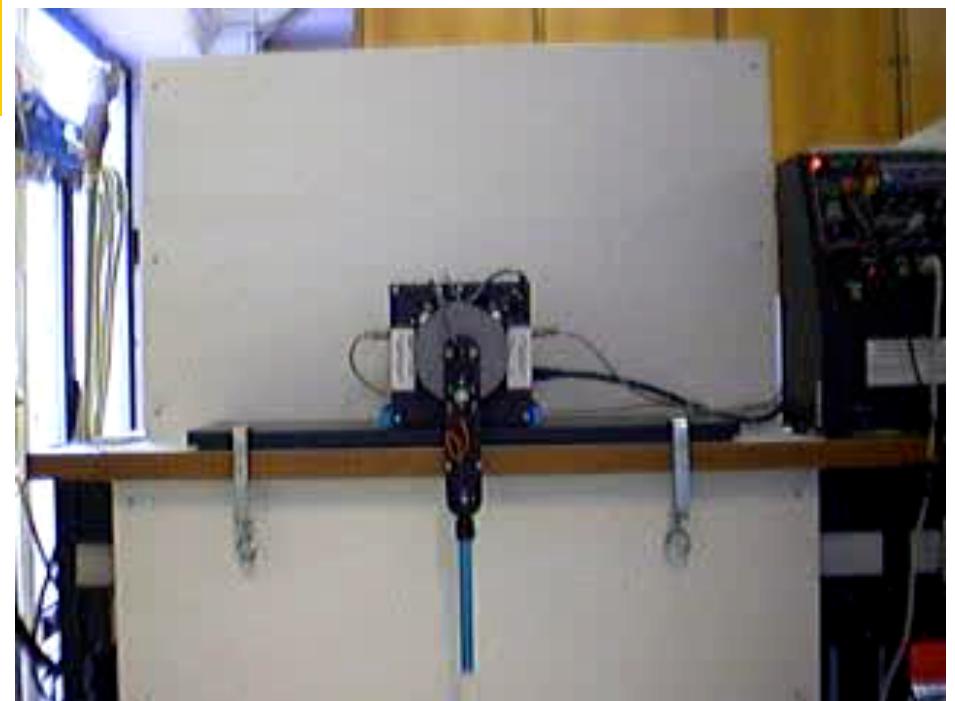


Experimental setup

Quanser Pendubot



with encoders on both joints



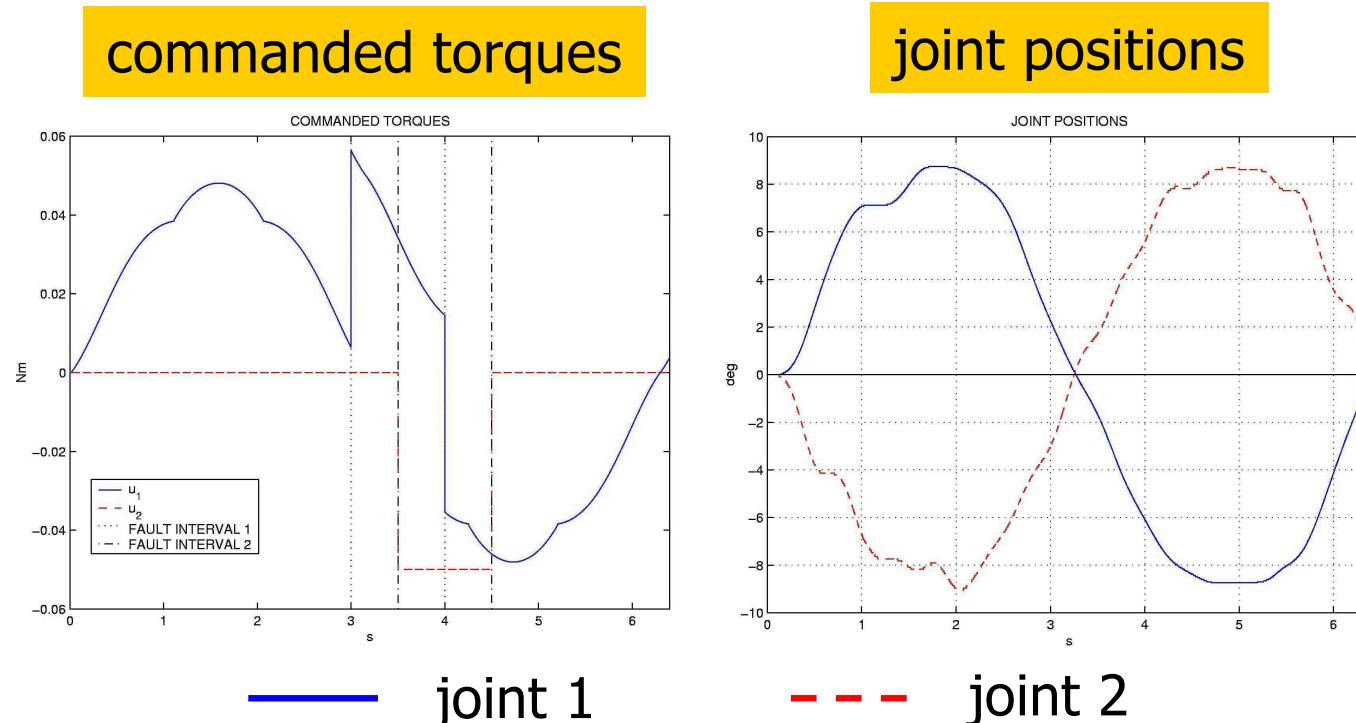
nonlinear control for swing-up

sampling time $T_c = 1$ ms, residual gains $K_i = 50$,
practical thresholds of fault detection $\cong 10^{-2} \div 10^{-3}$ Nm



First experiment

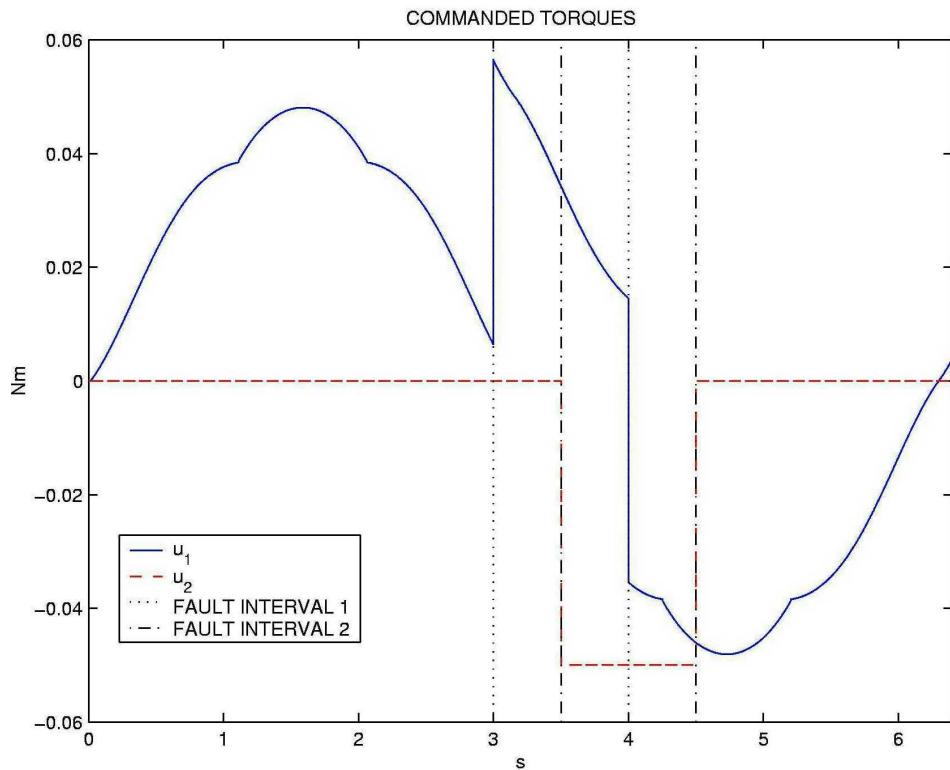
- motor 1 driven by sinusoidal voltage of period 2π sec (open loop)
- **bias fault** on u_1 for $t \in [3 \div 4]$ sec
- **total fault** on second joint for $t \in [3.5 \div 4.5]$ sec (a constant torque is requested, but **no motor at the joint to provide 0.05 Nm...**)
- **fault concurrency** for $t \in [3.5 \div 4]$ sec



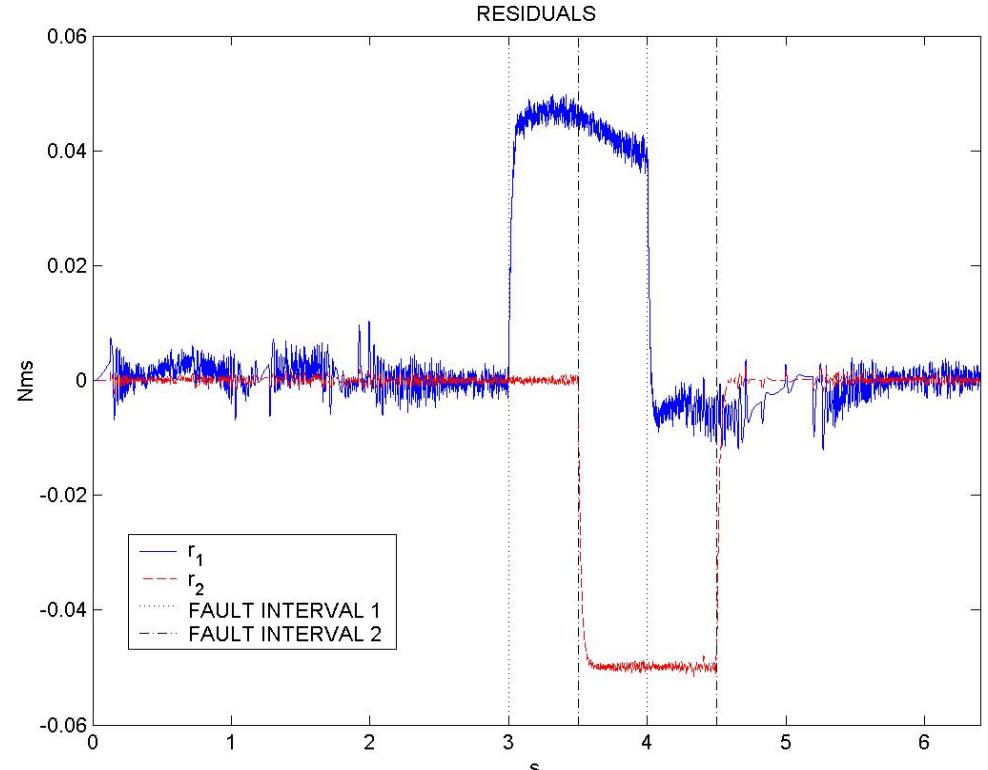


First experiment – FDI

commanded torques



residuals

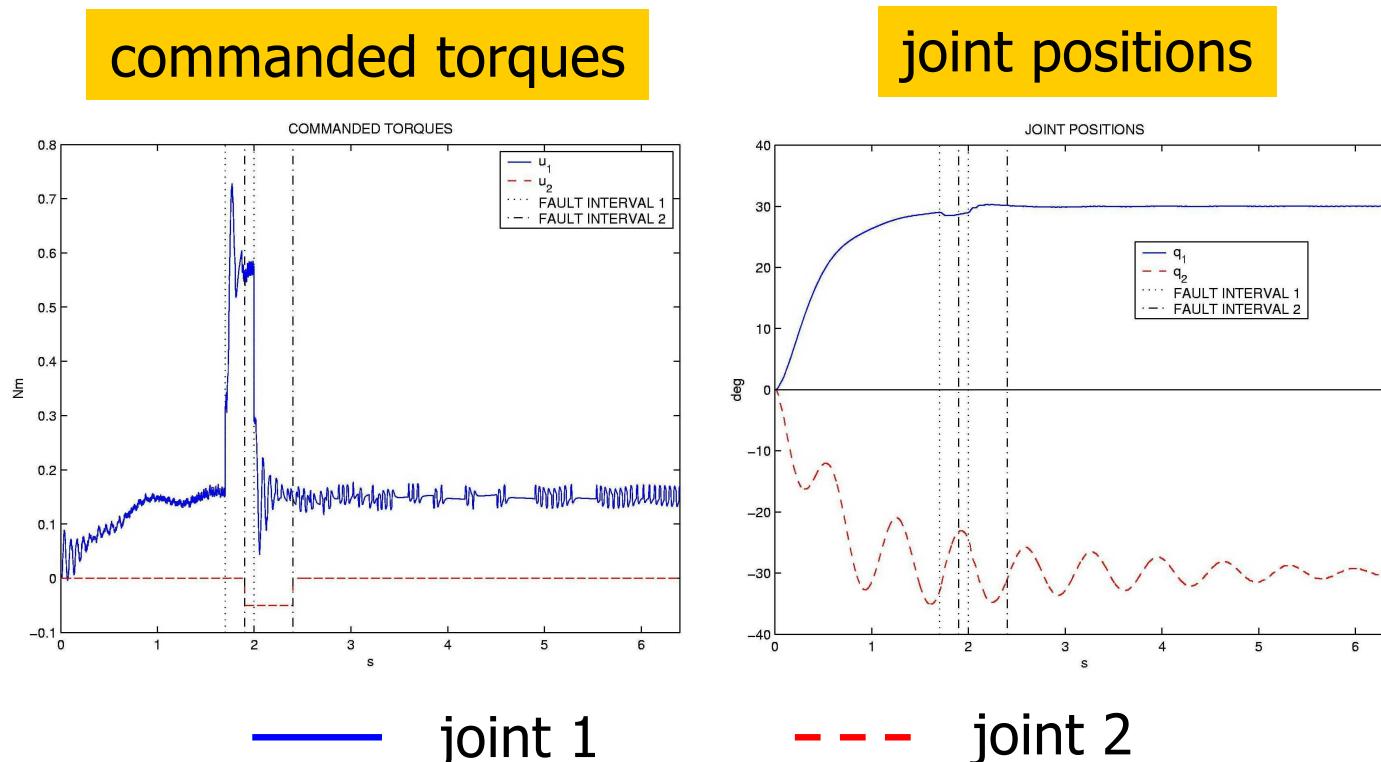


— joint 1

- - - joint 2

Second experiment

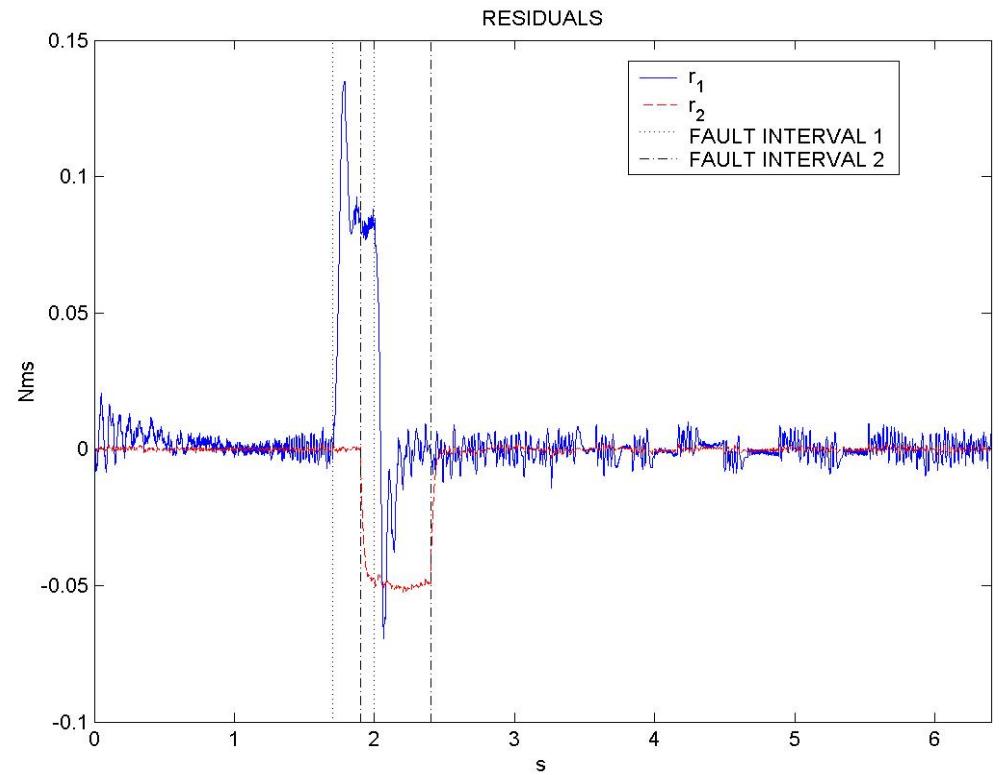
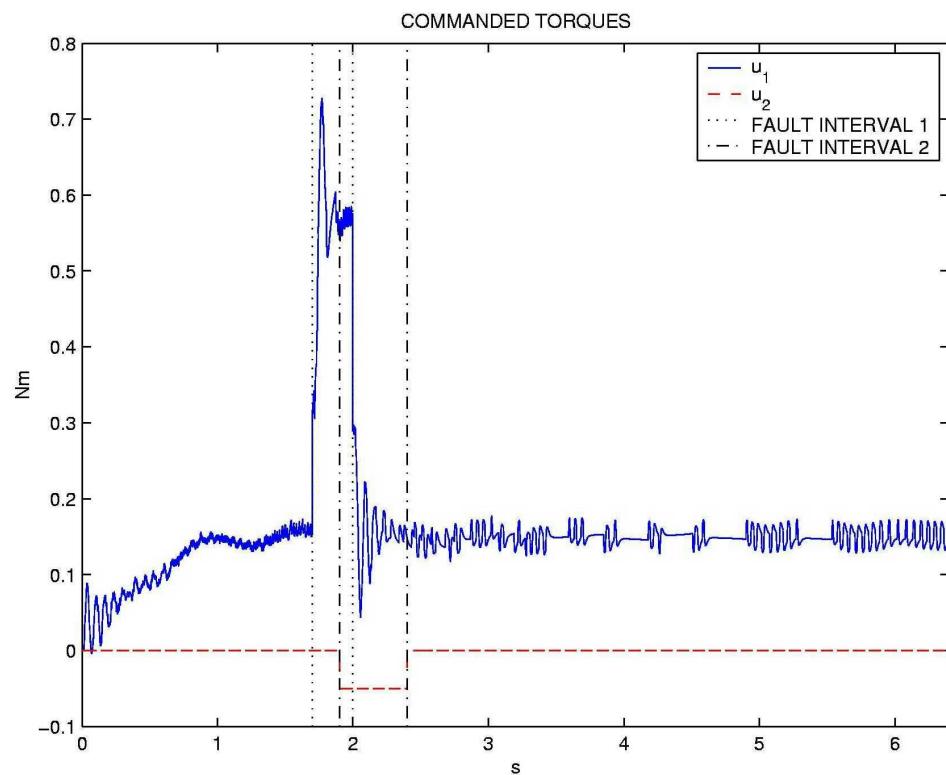
- position regulation of the first joint at $q_{d1} = 30^\circ$ (**PID control**)
- 50% power loss** on motor 1 for $t \in [1.7 \div 2]$ sec
- total fault** on joint 2 for $t \in [1.9 \div 2.4]$ sec (**no motor...**)
- fault concurrency** for $t \in [1.7 \div 1.9]$ sec



Second experiment – FDI

commanded torques

residuals

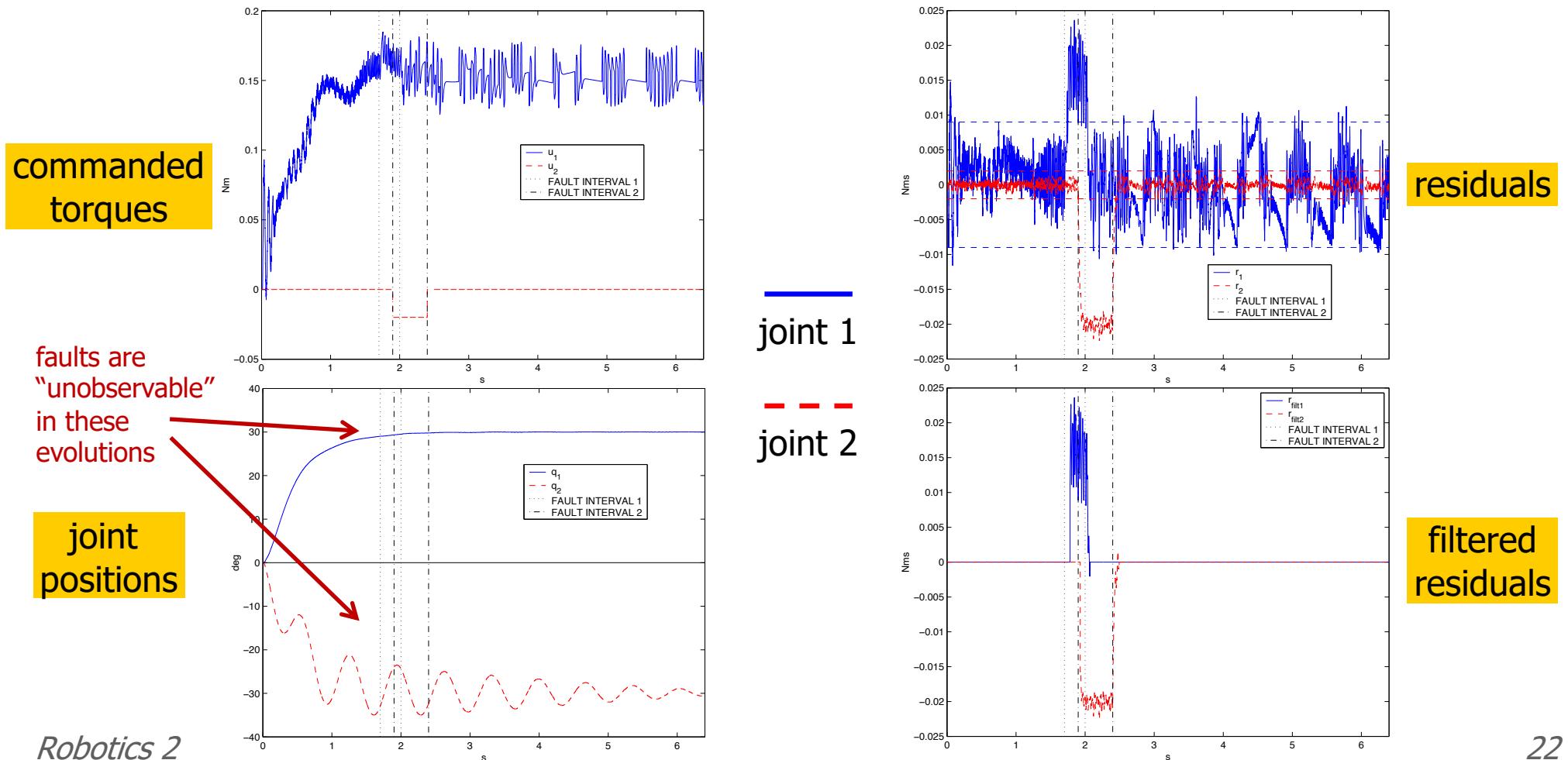


— joint 1

- - - joint 2

Third experiment – FDI

- same as in second experiment, but with only **10% power loss** on motor 1
 - due to noisy PWM signals driving the DC motor, a **dynamic filtering** of residuals is used, staying above [below] a threshold ($r_{1,thres} = 9 \cdot 10^{-3}$ Nm, $r_{2,thres} = 2 \cdot 10^{-3}$ Nm) for a time $T_{set} = 0.02$ s [$T_{reset} = 0.03$ s] before detecting a fault [reset to normal operation]





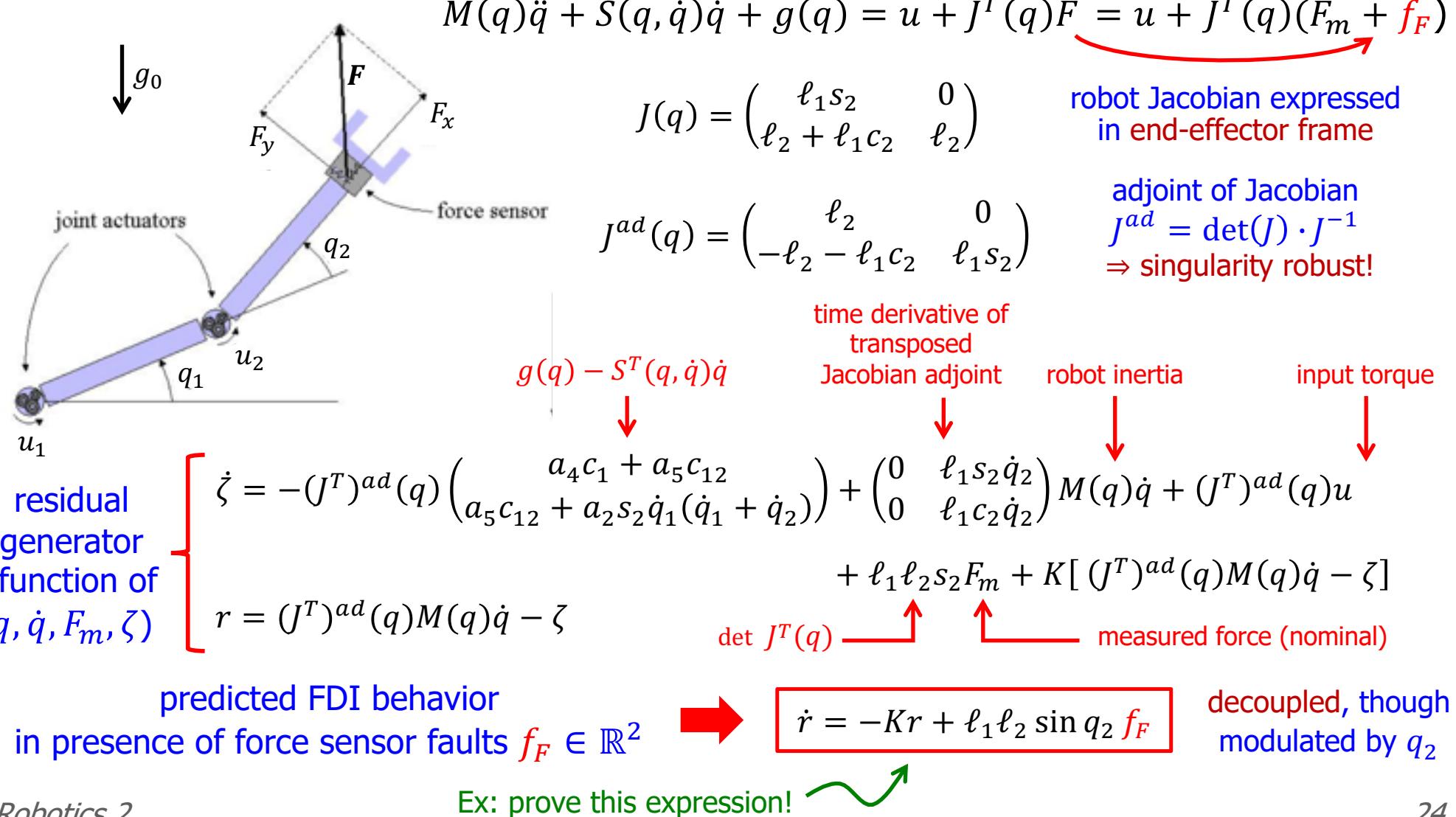
Extensions

- FDI method based on generalized momentum is easily extended to the presence of **flexible transmissions** (elastic joints), **actuator dynamics**, ...
- the scheme can be made **adaptive**, so as to handle parametric uncertainties in the robot dynamic model
- the method can be modified for detection and isolation of significant classes of **sensor faults** (e.g., faults in force/torque sensor at the wrist)
 - applies to all faults that instantaneously affect robot **acceleration** or **torque** (i.e., occurring at the second-order differential level)
- assuming **non-concurrency** (at most a single fault occurs at the same time) of a given set of faults, **relaxed FDI conditions** have been derived
 - of interest when the necessary conditions for multiple FDI are violated
 - involves processing of **continuous** residuals + **discrete** logic for isolation
- the same FDI-type approach has been applied also for **compensation of unmodeled friction** (treated as a “permanent fault” on the system)
- combination of **model-** and **signal-based** approaches to FDI



Isolation of F/T sensor faults

- planar 2R robot with **fault** on force **measure** of sensor on the end-effector

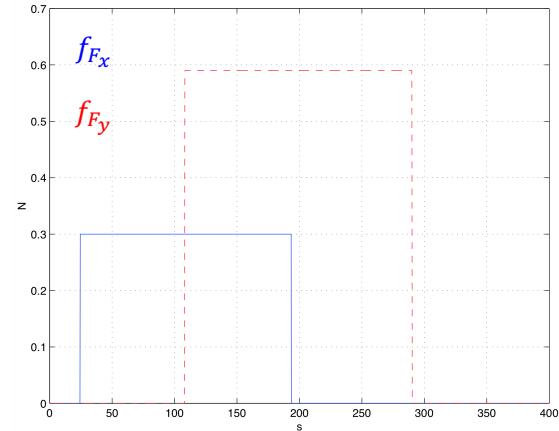




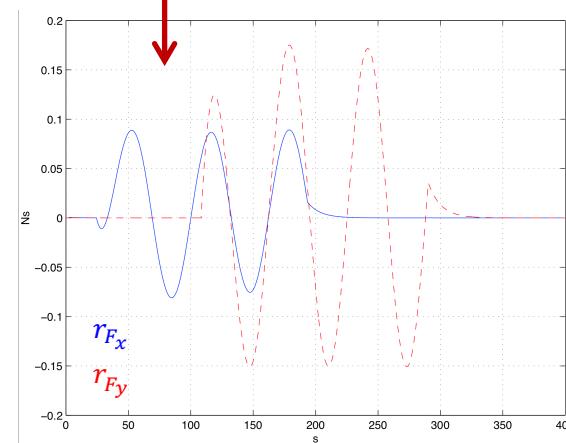
Isolation of F/T sensor faults

- simulation on the 2R robot

bias faults
on the two components
of force sensor measures
0.3N on f_{F_x} in $t \in [25 \div 190]$
0.6N on f_{F_y} in $t \in [109 \div 285]$



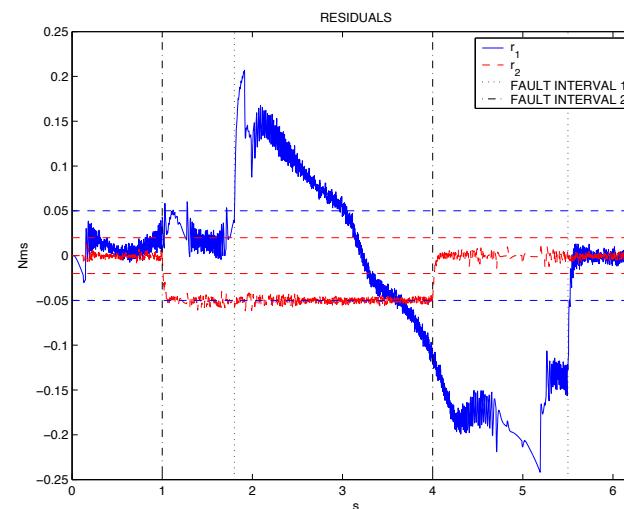
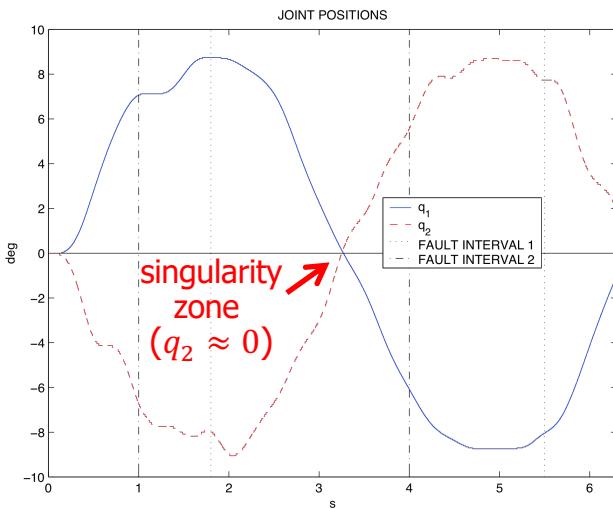
q_2 is tracking a sinusoid ($A = \pi/8$ rad, $\omega = 0.1$ rad/s)



FDI residual
components
(with $K = 0.1I$)

- experiment on the Pendubot (no force sensor and no contact!)

evolution
of joint
variables



residuals
for emulated bias
measurement faults
-1N on F_x in $t \in [1.8 \div 5.5]$
0.05N on F_y in $t \in [1 \div 4]$

$(J^T)^{ad} \rightarrow \text{diag}\{s_2, 1\} J^{-T}$
in previous scheme



Isolation of non-concurrent faults

- faults of the actuators **AND** faults of the force sensor components (possibly occurring **simultaneously**) **CANNOT** be detected **AND** isolated
 - for a mechanical system with N dofs, the **max # of faults allowing FDI** is $N!$
- with **non-concurrency**, e.g., 2 actuator + 2 F/T sensor faults in 2R robot

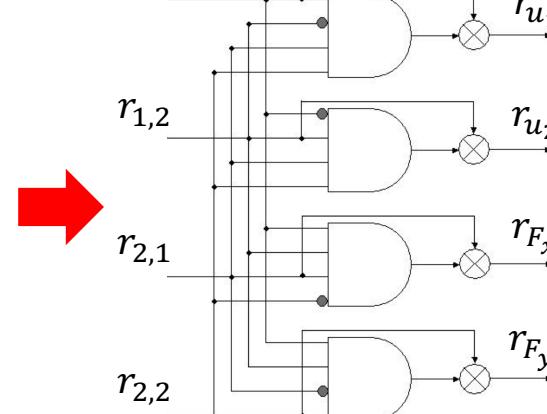
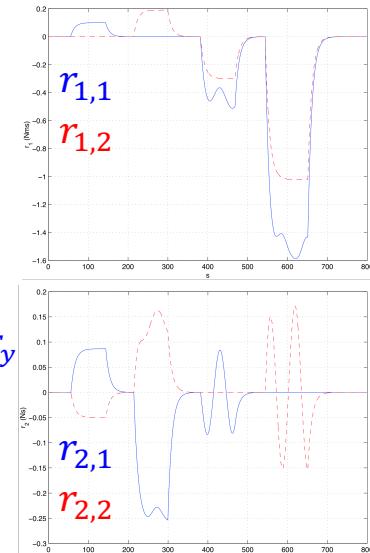
dependence of residuals on considered faults

residual fault	$r_{1,1}$	$r_{1,2}$	$r_{2,1}$	$r_{2,2}$
f_{u_1}	1	0	1	1
f_{u_2}	0	1	1	1
f_{F_x}	1	1	1	0
f_{F_y}	1	1	0	1

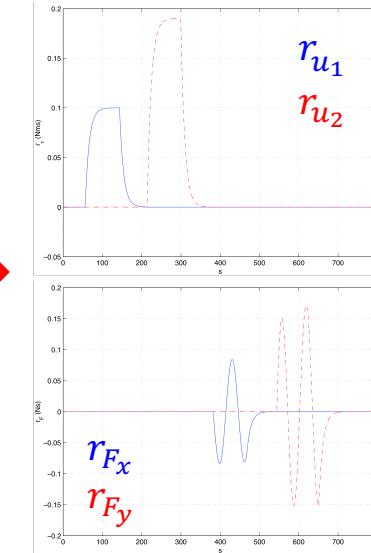
isolation matrix

$r_{2,1} \ r_{2,2}$	11	10	01	00
$r_{1,1} \ r_{1,2}$				
10	f_{u_1}	NA	NA	NA
01	f_{u_2}	NA	NA	NA
11	NC	f_{F_x}	f_{F_y}	NA
00	NA	NA	NA	no fault

time sequence of non-concurrent bias faults:
 $f_{u_1} \rightarrow f_{u_2} \rightarrow f_{F_x} \rightarrow f_{F_y}$



isolation logics



hybrid residuals allowing isolation of 4 faults

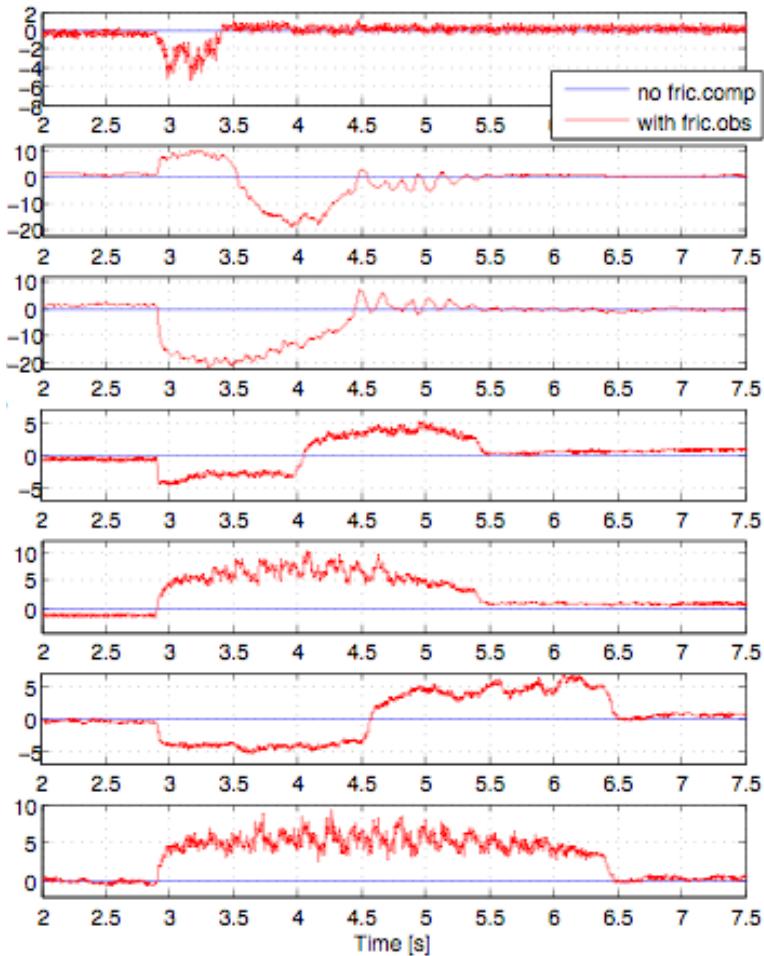


Experiments on friction compensation

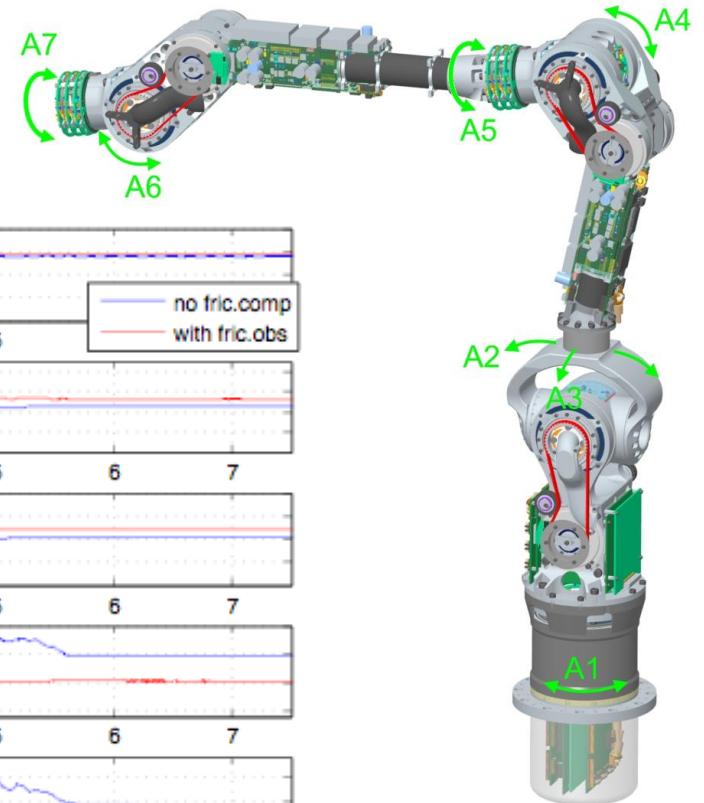
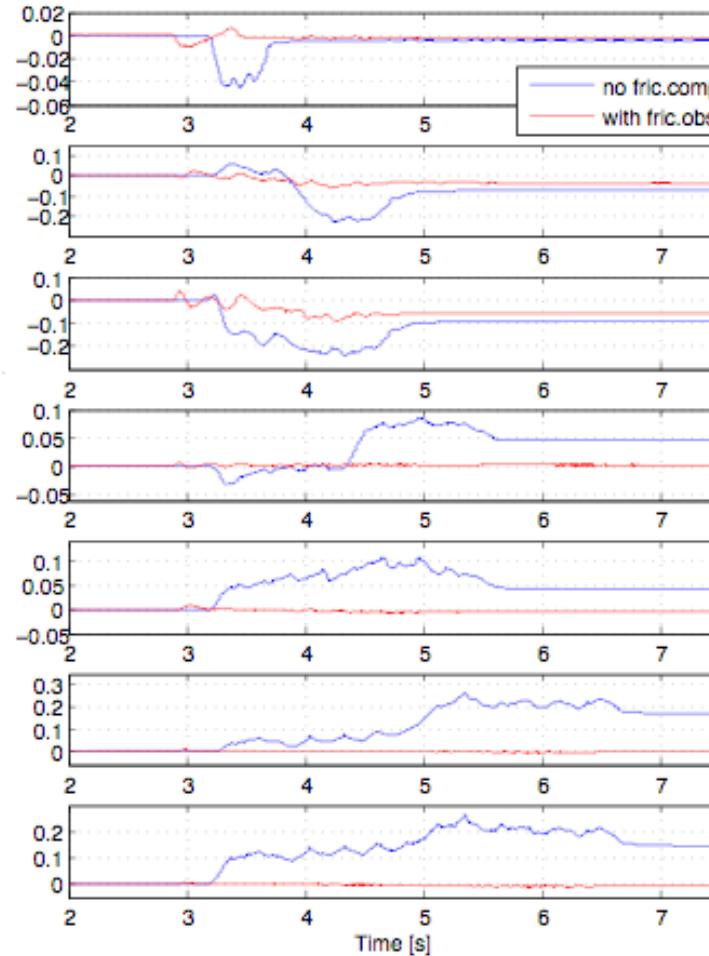
- results on the DLR 7R medical robot

used then on-line
in control law...

friction estimate via residuals



position error

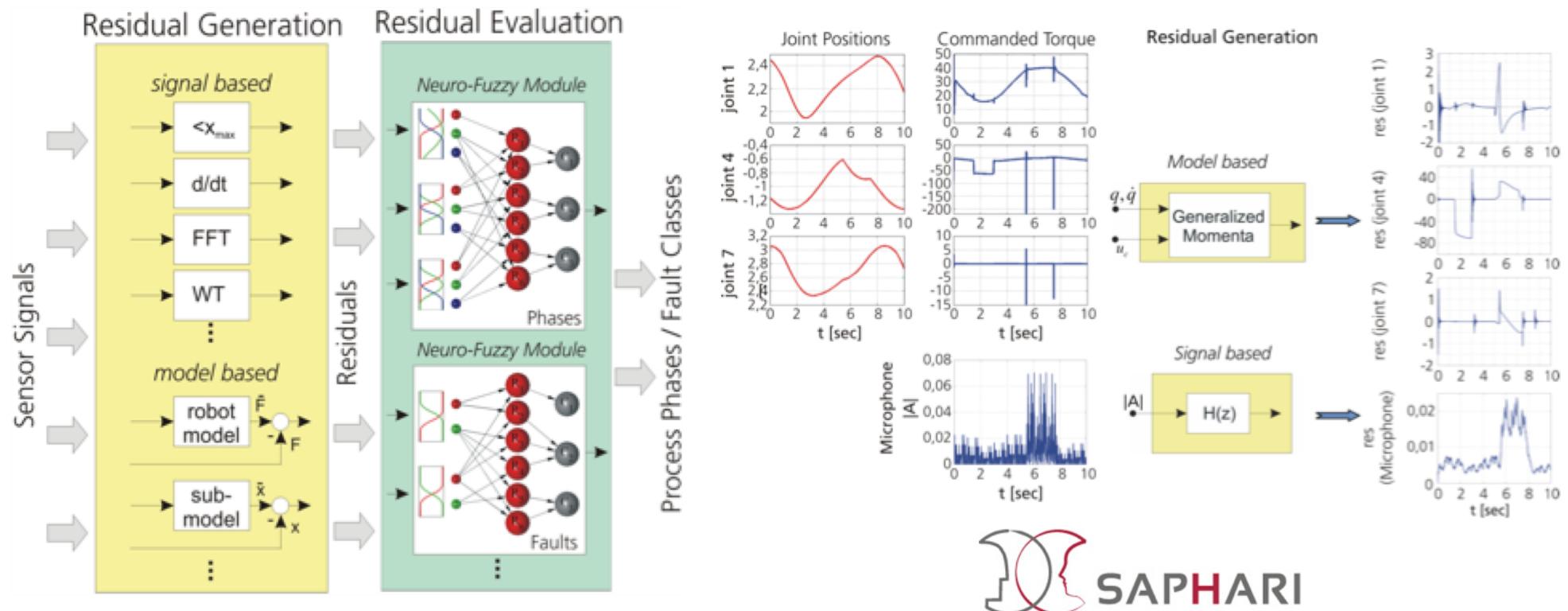


HD at the joints
⇒ elastic joint
dynamic model



Model- and signal-based FDI

- detection and isolation features can be enhanced by combining multiple sensor inputs and different approaches
 - model-based (exact, but require accurate models)
 - signal-based (approximate, but without special requirements)
- so as to obtain the “best of both worlds”





Bibliography

- X. Zhang, M. Polycarpou, T. Parisini, "Robust detection and isolation scheme for abrupt and incipient faults in nonlinear systems," *IEEE Trans. on Automatic Control*, vol. 47, no. 4, pp. 576-592, 2002.
- A. De Luca, R. Mattone, "Actuator failure detection and isolation using generalized momenta," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 634-639, 2003.
- A. De Luca, R. Mattone, "An adapt-and-detect actuator FDI scheme for robot manipulators," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 4975-4980, 2004.
- A. De Luca, R. Mattone, "An identification scheme for robot actuator faults," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp.1127-1131, 2005.
- R. Mattone, A. De Luca, "Relaxed fault detection and isolation: An application to a nonlinear case study," *Automatica*, vol. 42, no. 1, pp. 109-116, 2006.
- R. Mattone, A. De Luca, "Nonlinear fault detection and isolation in a three-tank heating system," *IEEE Trans. on Control Systems Technology*, vol. 14, no. 6, pp. 1158-1166, 2006.
- L. Le Tien, A. Albu-Schäffer, A. De Luca, G. Hirzinger, "Friction observer and compensation for control of robots with joint torque measurements," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 3789-3795, 2008.
- C. Gaz, A. Cristofaro, A. De Luca, "Detection and isolation of actuator faults and collisions for a flexible robot arm," *Proc. 59th IEEE Conf. on Decision and Control*, pp. 2684-2689, 2020.