



Robotics 1

Kinematic control

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI





Robot motion control

- need to “actually” realize a desired robot motion task ...
 - regulation of pose/configuration (constant reference)
 - trajectory following/tracking (time-varying reference)
- ... despite the presence of
 - external disturbances and/or unmodeled dynamic effects
 - initial errors (or arising later due to disturbances) w.r.t. desired task
 - discrete-time implementation, uncertain robot parameters, ...
- we use a general control scheme based on
 - feedback (from robot state measures, to impose asymptotic stability)
 - feedforward (nominal commands generated in the planning phase)
- the error driving the feedback part of the control law can be defined either in Cartesian or in joint space
 - control action always occurs at the joint level (where actuators drive the robot), but performance has to be evaluated at the task level



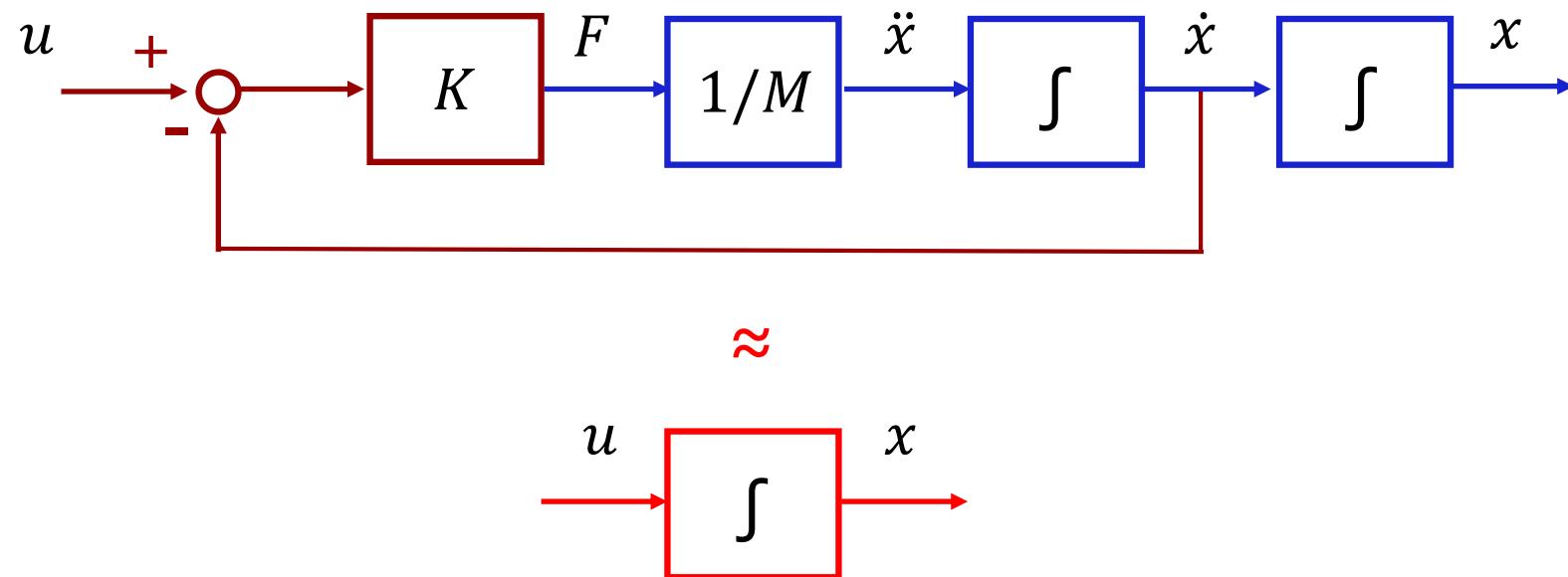
Kinematic control of robots

- a robot is an electro-mechanical system driven by actuating **torques** produced by the motors
- it is possible, however, to consider a **kinematic command** (most often, a **velocity**) as control input to the system...
- ...thanks to the presence of **low-level feedback control** at the robot joints that allows imposing commanded reference velocities (at least, in the “ideal case”)
- these feedback loops are present in industrial robots within a **“closed” control architecture**, where users can only specify reference commands of the kinematic type
- in this way, **performance** can be very satisfactory, provided the desired motion is **not too fast** and/or **does not require large accelerations**



An introductory example

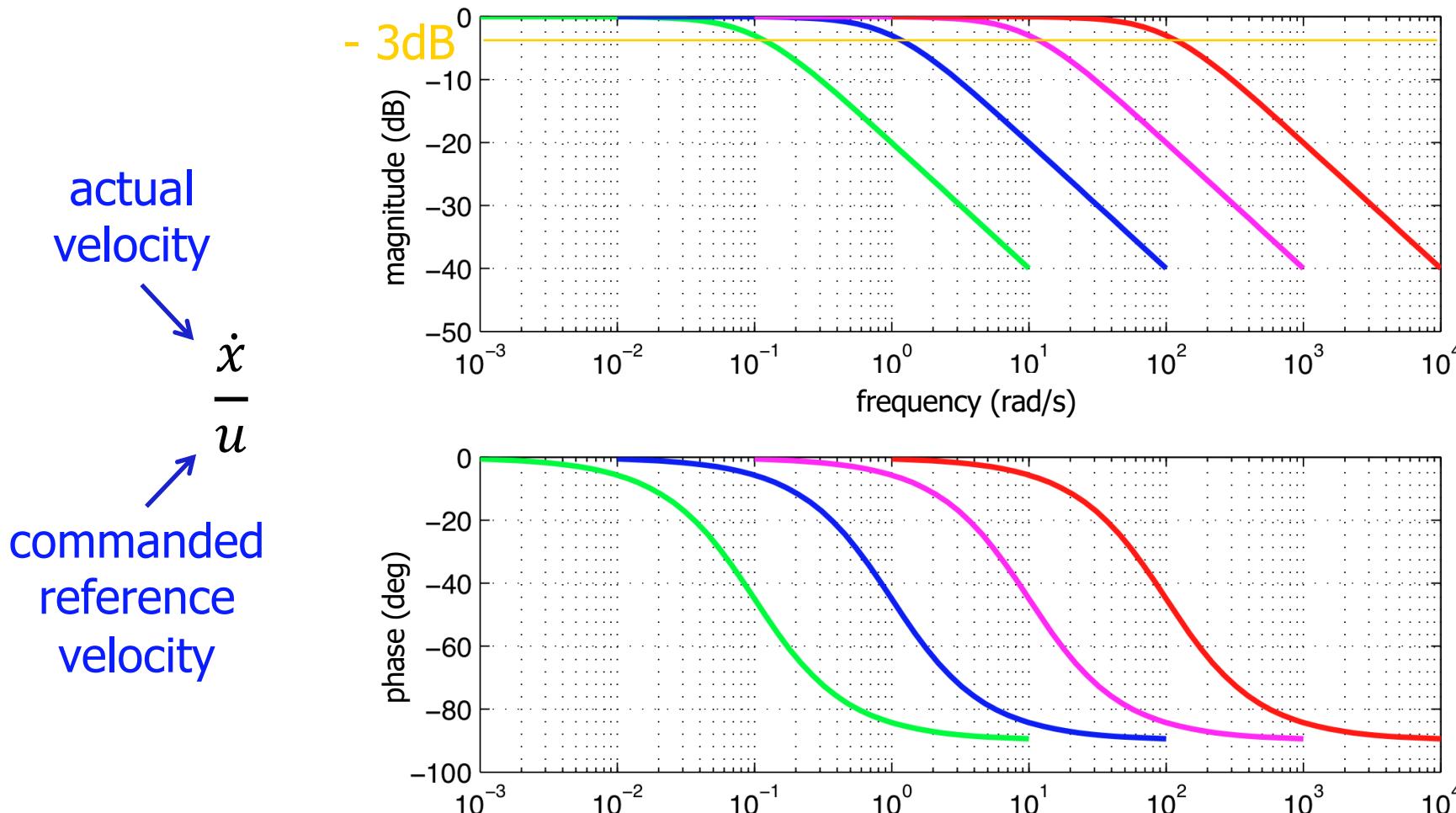
- a mass M in linear motion: $M\ddot{x} = F$
- low-level feedback: $F = K(u - \dot{x})$, with u = reference velocity
- equivalent scheme for $K \rightarrow \infty$: $\dot{x} \approx u$
- in practice, valid in a limited frequency “bandwidth” $\omega \leq K/M$





Frequency response of the closed-loop system

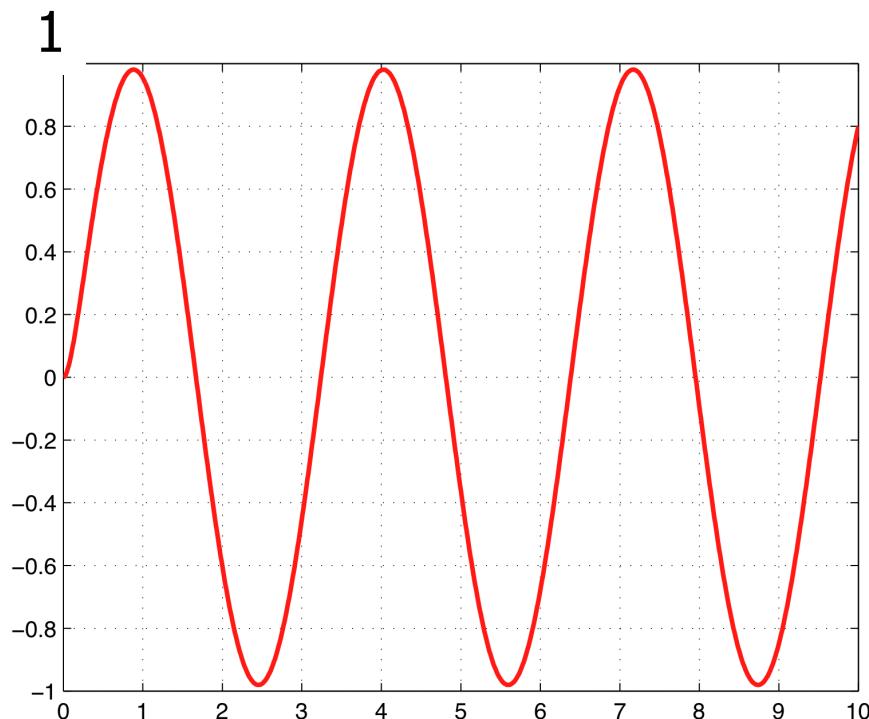
- Bode diagrams of $P(s) = \frac{v(s)}{u(s)} = \frac{sx(s)}{u(s)}$ for $K/M = 0.1, 1, 10, 100$



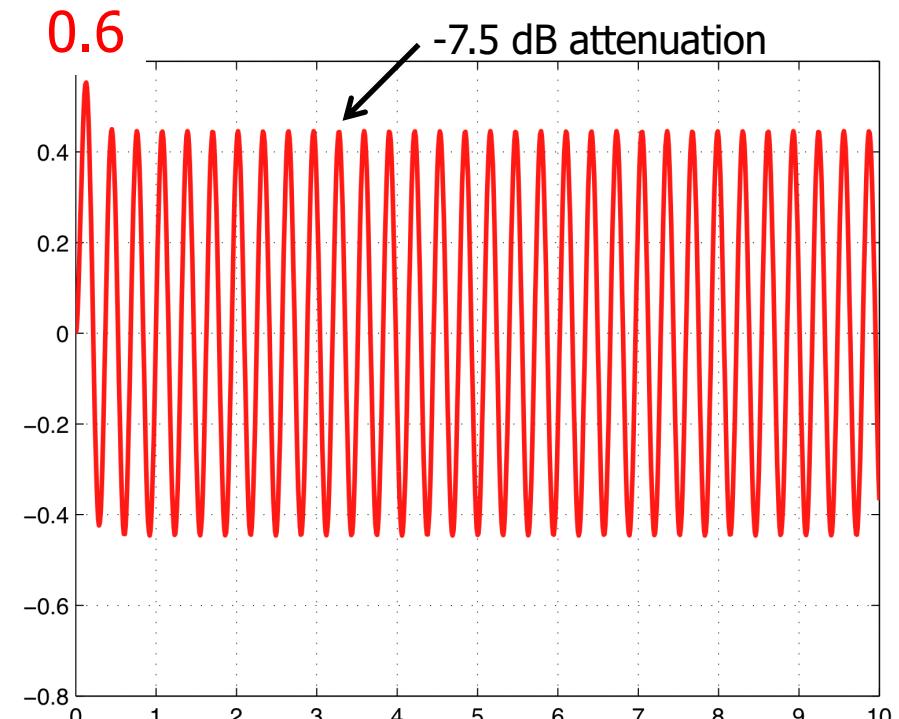


Time response

- setting $K/M = 10$ (bandwidth), we show two possible time responses to unit sinusoidal velocity reference commands at different ω



$\omega = 2 \text{ rad/s}$



$\omega = 20 \text{ rad/s}$

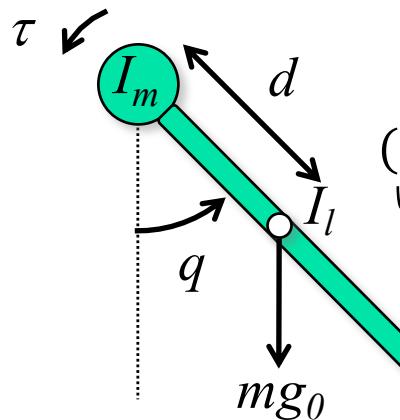
actually realized velocities



A more detailed example

including nonlinear dynamics

- single link (a thin rod) of mass m , center of mass at d from joint axis, inertia M (motor + link) at the joint, rotating in a vertical plane (the gravity torque at the joint is configuration dependent)



dynamic model

$$(I_m + I_l + md^2)\ddot{q} + mg_0d \sin q = \tau$$

$I_m + I_l + md^2$

M

$$\begin{aligned} g_0 &= 9.81 [m/s^2] \\ m &= 10 [kg] \\ d &= l/2 = 0.2 [m] \\ I_l &= ml^2/12 = 0.1333 [kgm^2] \\ I_m &= 0.5333 [kgm^2] \\ &\quad (= I_l + md^2) \\ \Rightarrow M &= 1.0667 [kgm^2] \end{aligned}$$

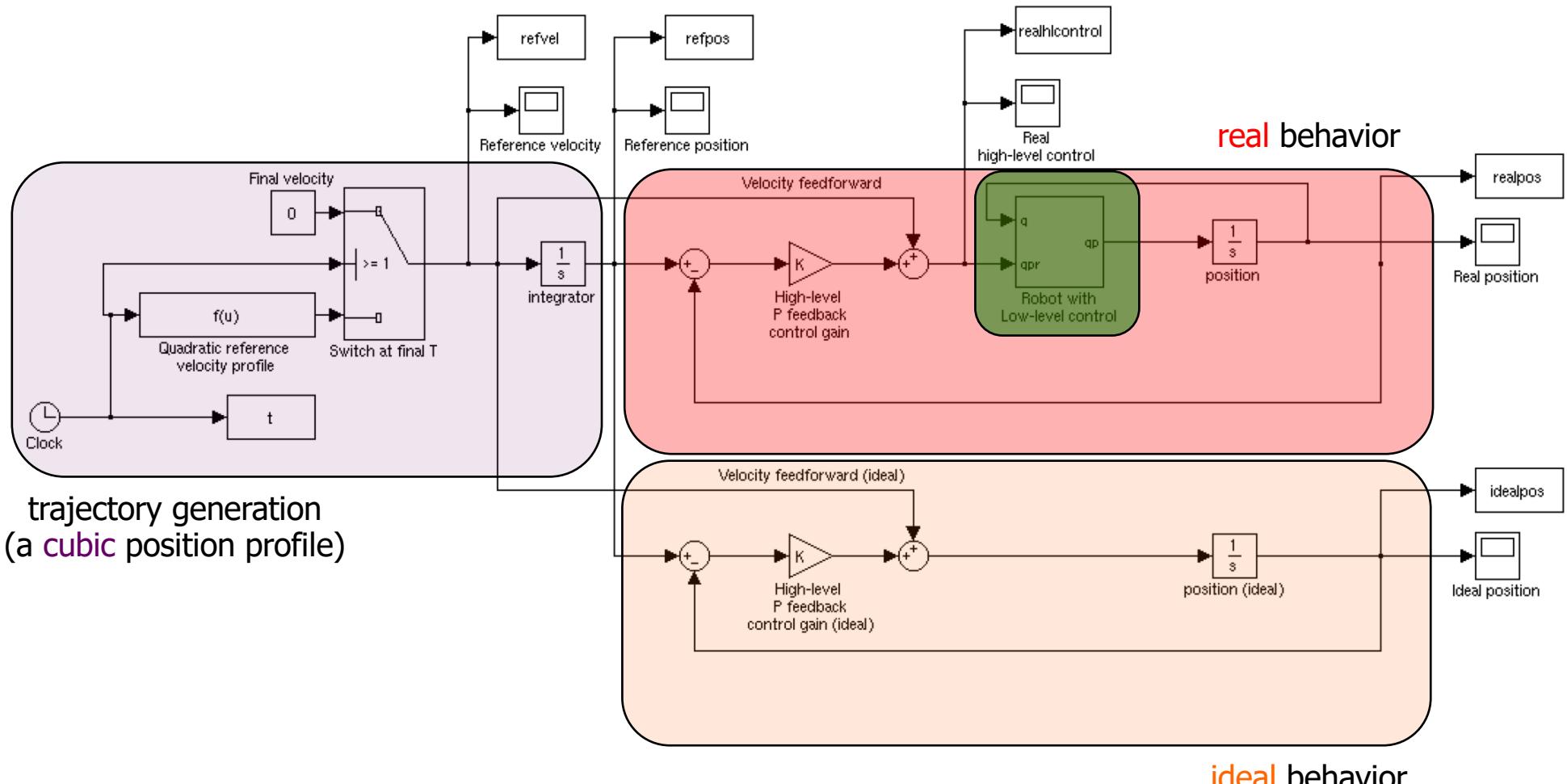
- fast **low-level feedback** control loop based on a PI action on the velocity error + an approximate acceleration feedforward
- kinematic control** loop based on a P feedback action on the position error + feedforward of the velocity reference at the joint level
- evaluation of tracking **performance** for rest-to-rest motion tasks with “increasing dynamics” = higher accelerations



A more detailed example

differences between the ideal and real case

- Simulink scheme

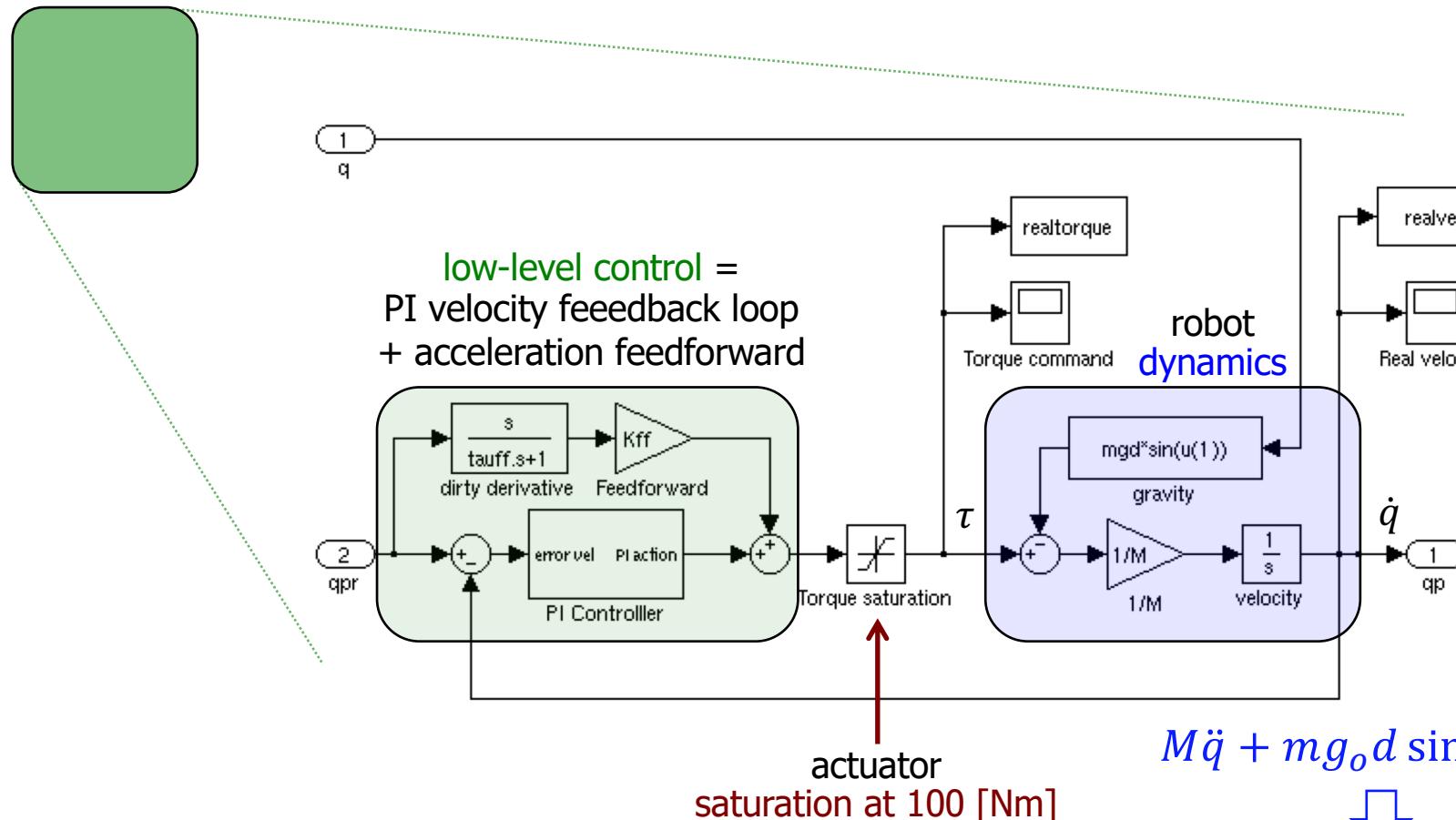




A more detailed example

robot with low-level control

- Simulink scheme

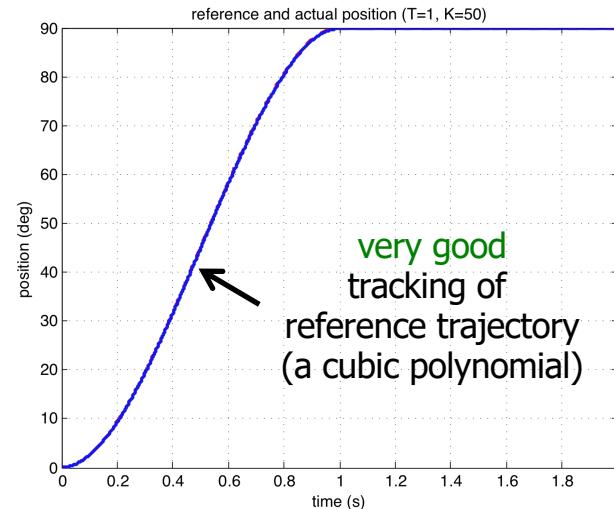




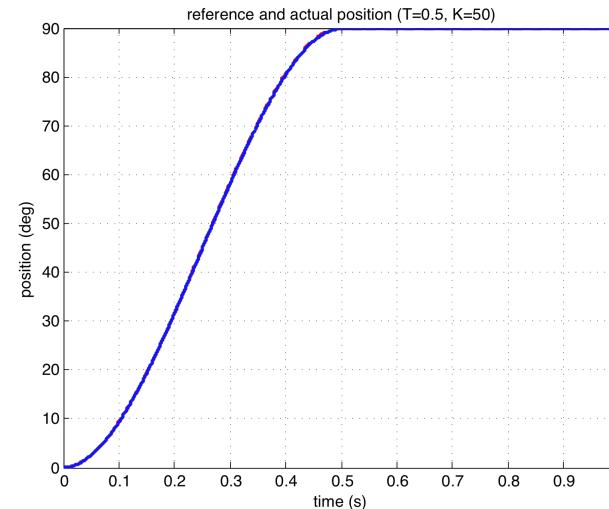
Simulation results

rest-to-rest motion from downward to horizontal position

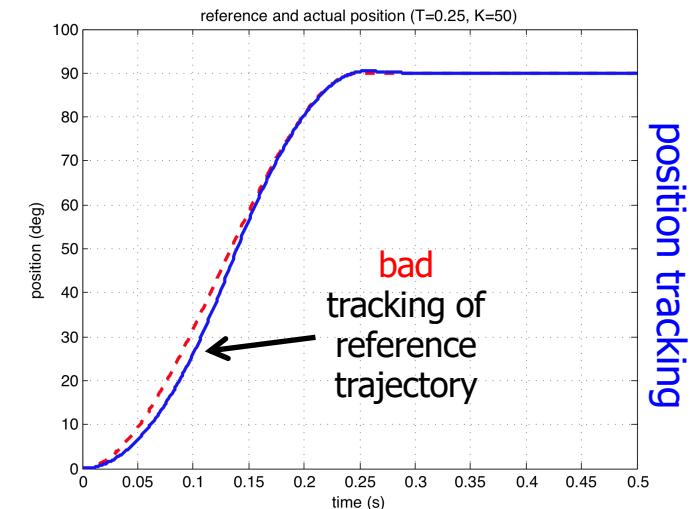
- in $T = 1 \text{ s}$



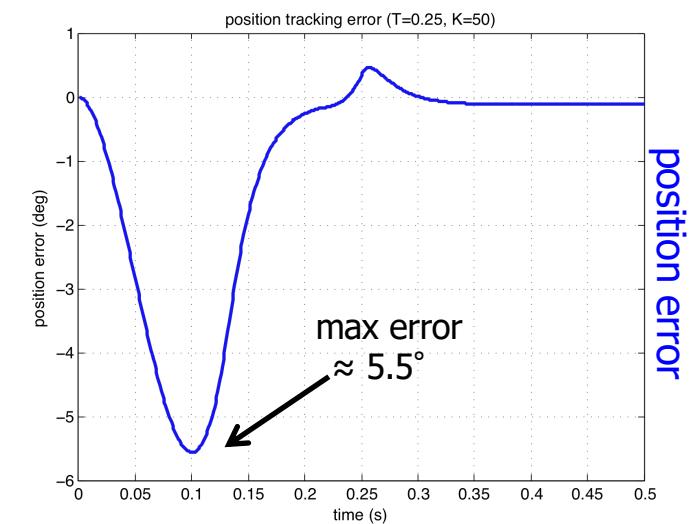
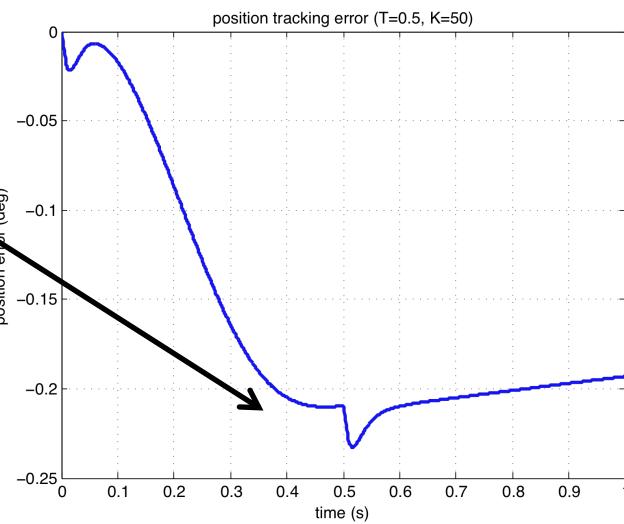
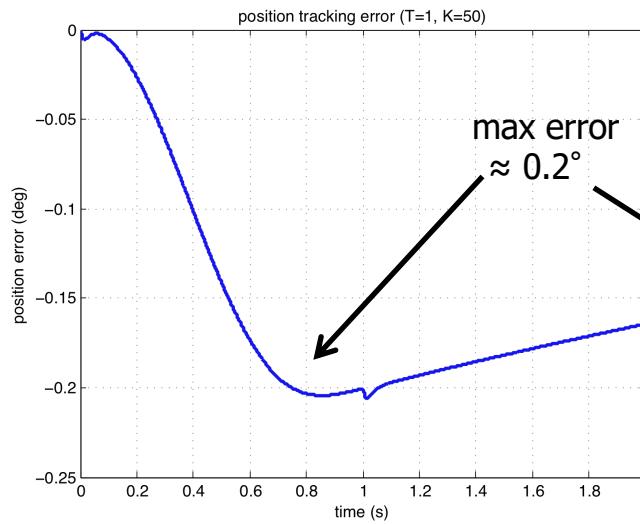
- in $T = 0.5 \text{ s}$



- in $T = 0,25 \text{ s}$



position tracking



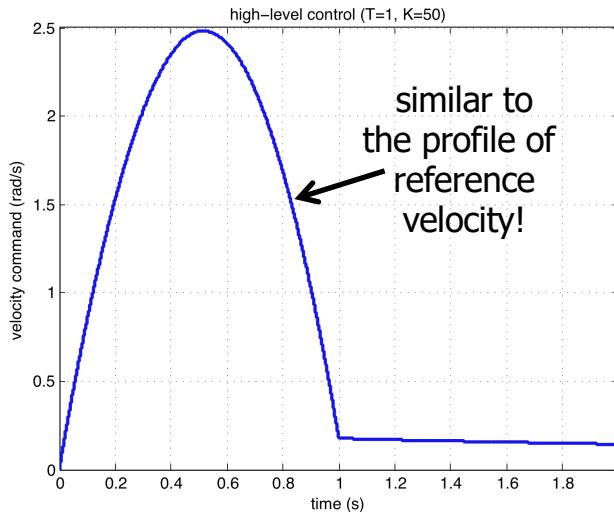
position error



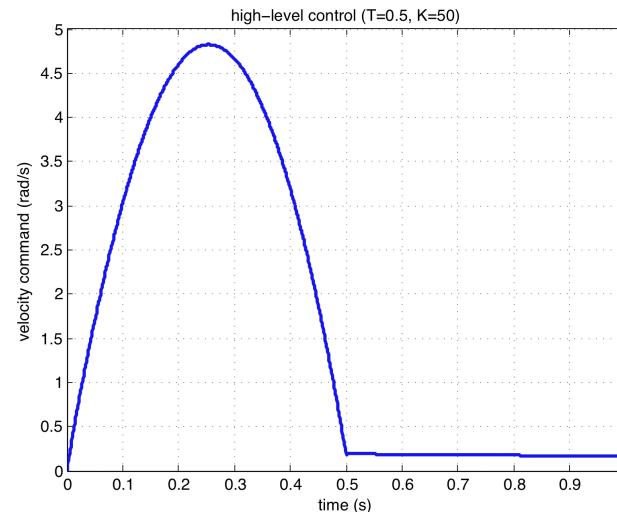
Simulation results

rest-to-rest motion from downward to horizontal position

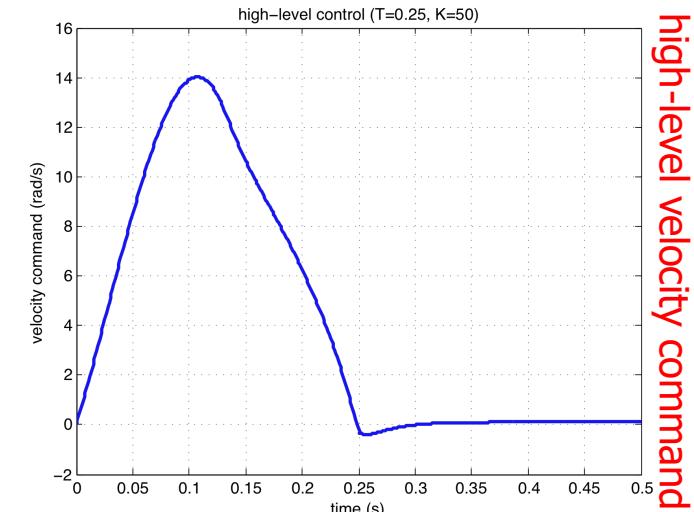
- in $T = 1 \text{ s}$



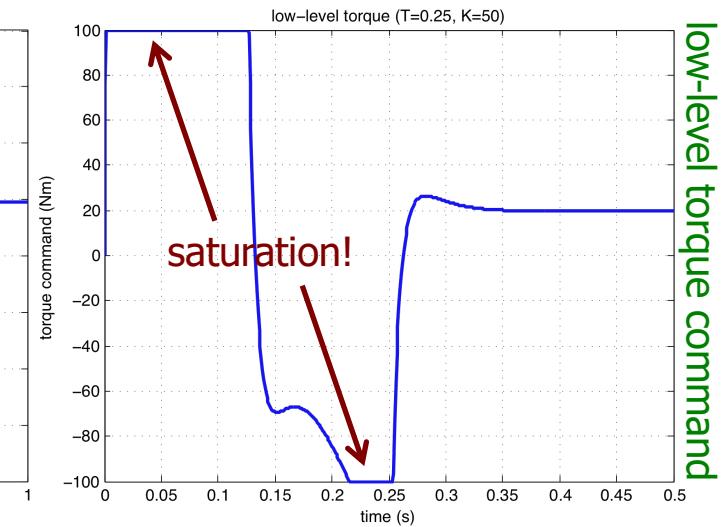
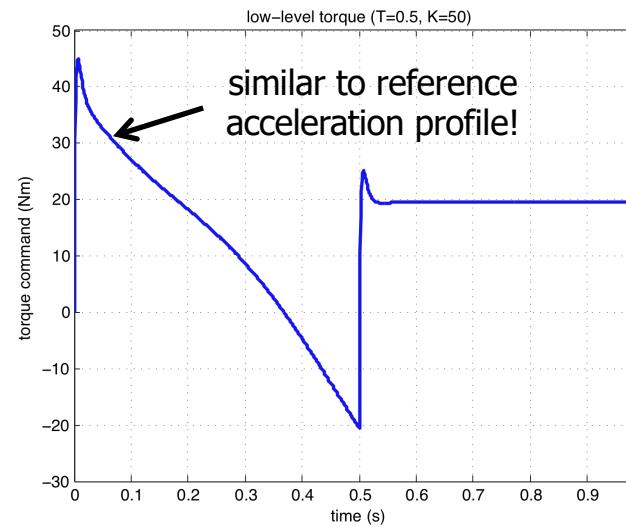
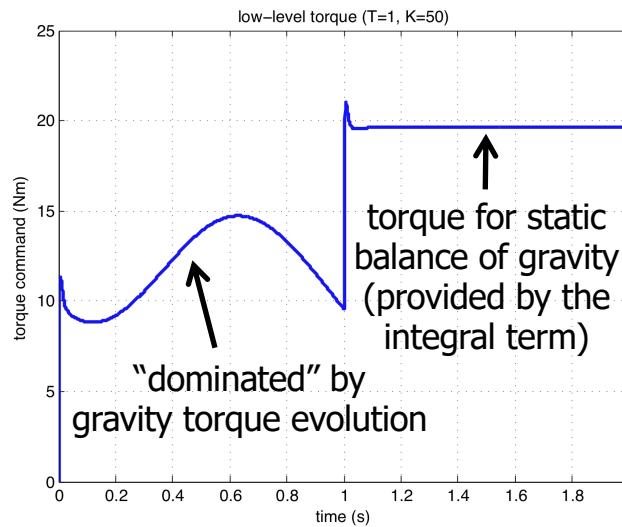
- in $T = 0.5 \text{ s}$



- in $T = 0.25 \text{ s}$



high-level velocity command



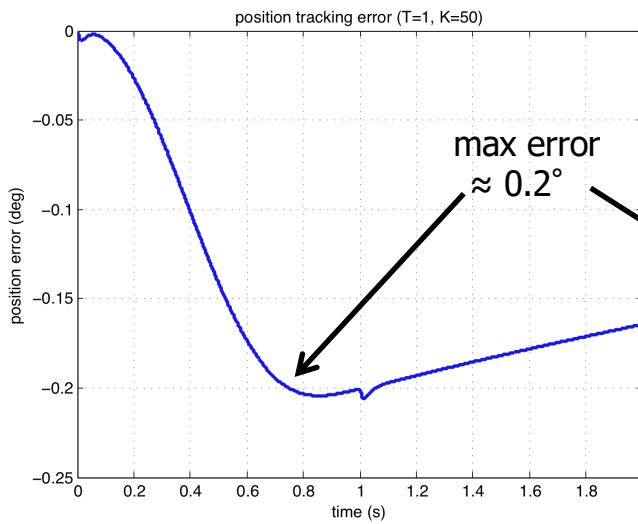
low-level torque command



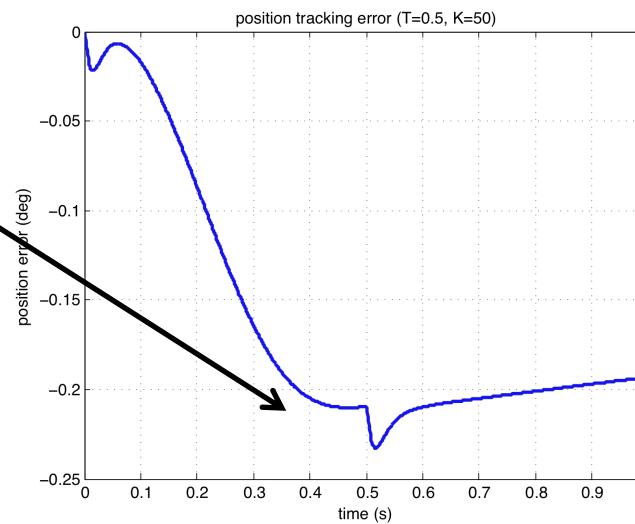
Simulation results

rest-to-rest motion from downward to horizontal position

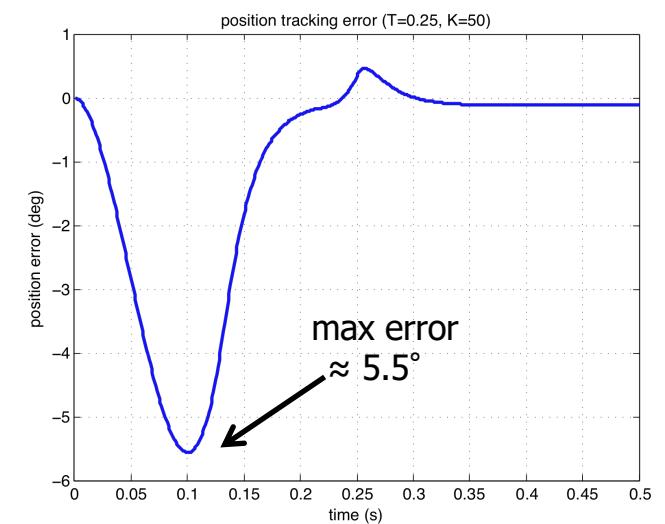
- in $T = 1 \text{ s}$



- in $T = 0.5 \text{ s}$



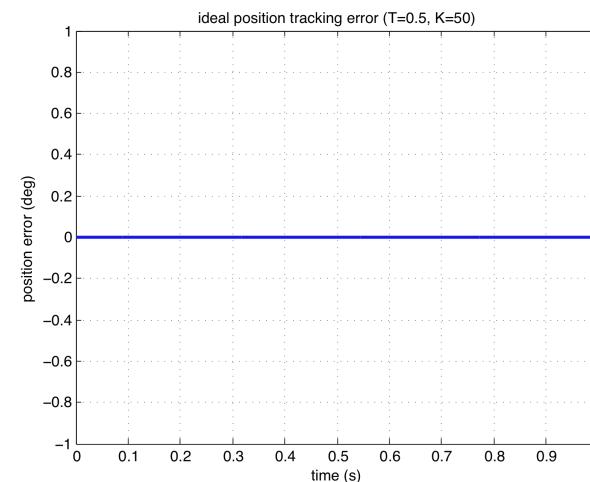
- in $T = 0.25 \text{ s}$



real position errors increase when reducing too much motion time
(\Rightarrow too high accelerations)

while **ideal** position errors
(based only on kinematics)
remain always the same!!

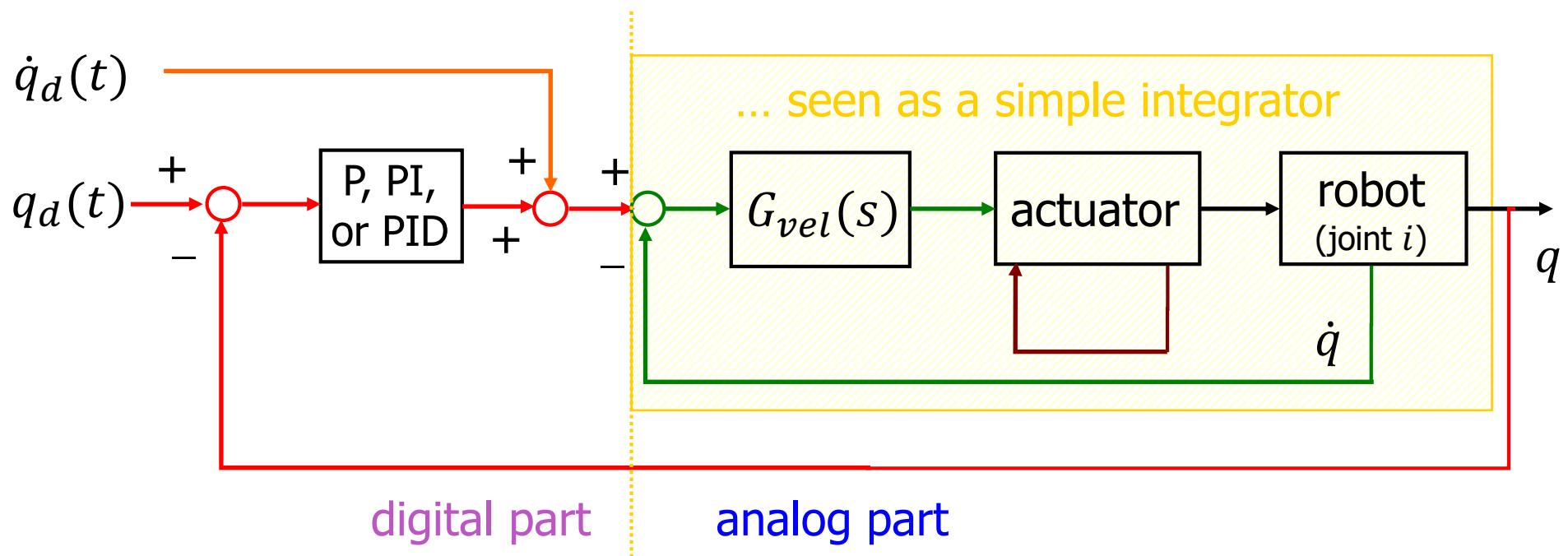
*here $\equiv 0$, thanks to the initial matching
between robot and reference trajectory*





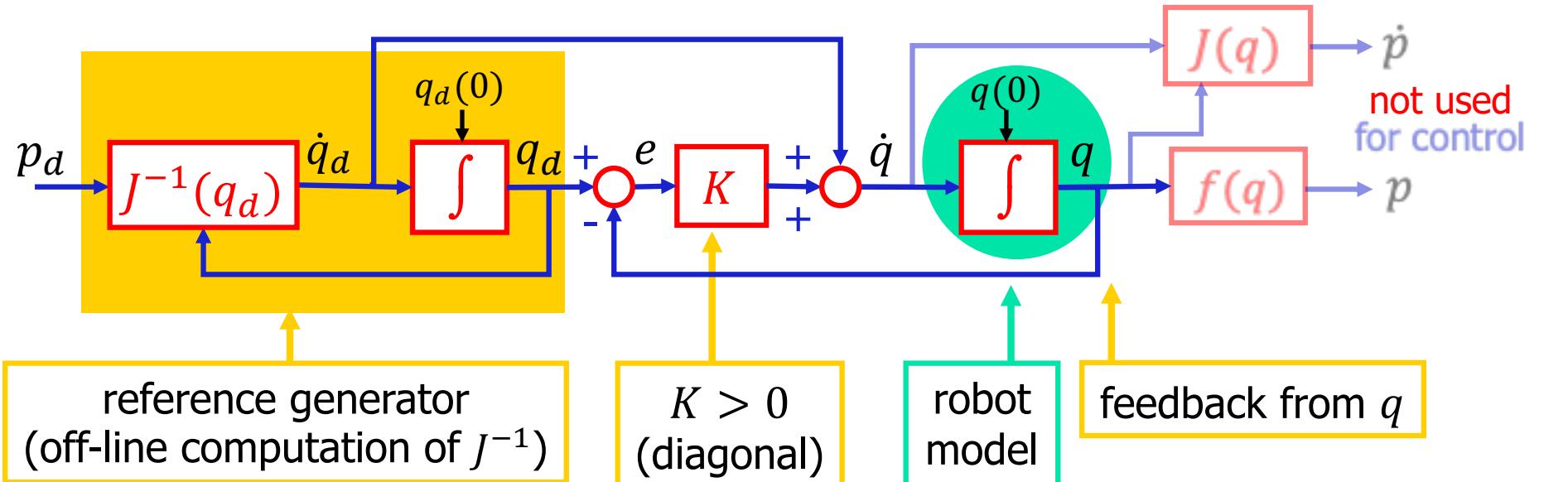
Control loops in industrial robots

- analog loop of large bandwidth on motor current (\propto torque)
- analog loop on velocity ($G_{vel}(s)$, typically a PI)
- digital feedback loop on position, with velocity feedforward
- this scheme is local to each joint (decentralized control)





Kinematic control of joint motion



$$e = q_d - q \rightarrow \dot{e} = \dot{q}_d - \dot{q} = \dot{q}_d - (\dot{q}_d + K(q_d - q)) = -Ke$$

decoupled $e_i \rightarrow 0$
 $(i = 1, \dots, n)$
exponentially,
 $\forall e(0)$

$$e_p = p_d - p \rightarrow \dot{e}_p = \dot{p}_d - \dot{p} = J(q_d)\dot{q}_d - J(q)(\dot{q}_d + K(q_d - q))$$

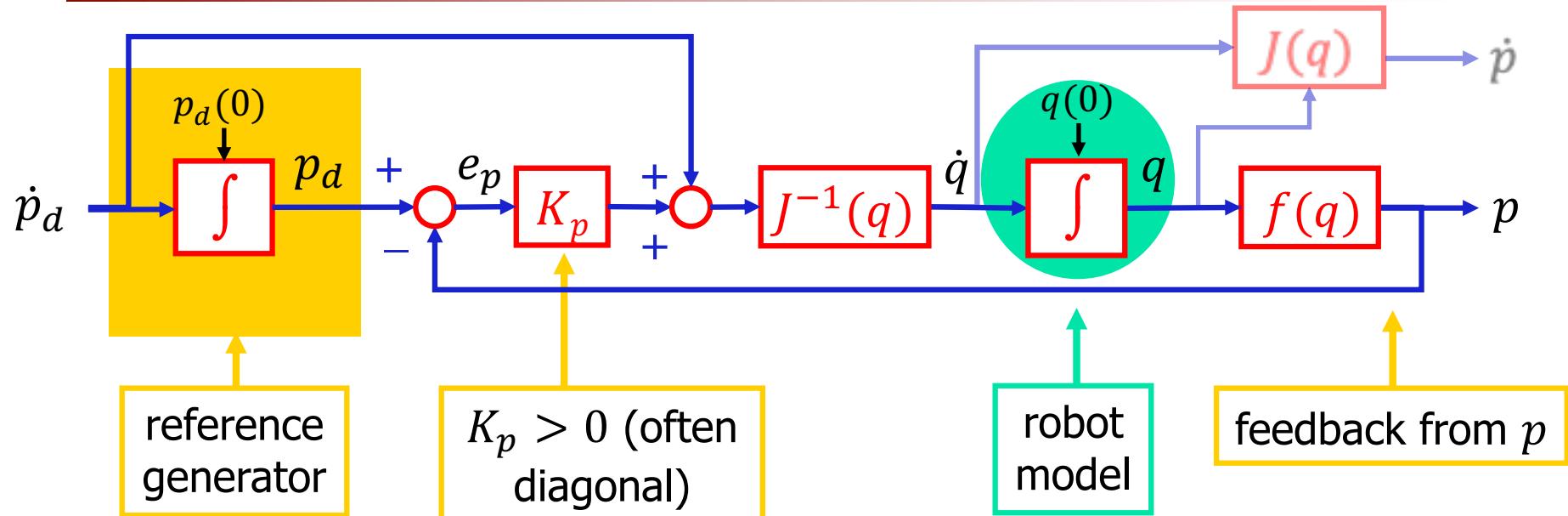
$$\begin{aligned} q &\triangleq q_d \\ e_p &\rightarrow J(q)e \end{aligned}$$

$$\dot{e}_p \approx -J(q)K J^{-1}(q)e_p$$

coupled Cartesian error dynamics



Kinematic control of Cartesian motion



$$e_p = p_d - p \rightarrow \dot{e}_p = \dot{p}_d - \dot{p} = \dot{p}_d - J(q)J^{-1}(q)(\dot{p}_d + K_p(p_d - p)) = -K_p e_p$$

- decoupled $e_{p,i} \rightarrow 0$ ($i = 1, \dots, m$) exponentially, $\forall e_p(0)$
- needs on-line computation of the inverse^(*) $J^{-1}(q)$
- real-time + singularities issues

(*) or pseudoinverse if $m < n$

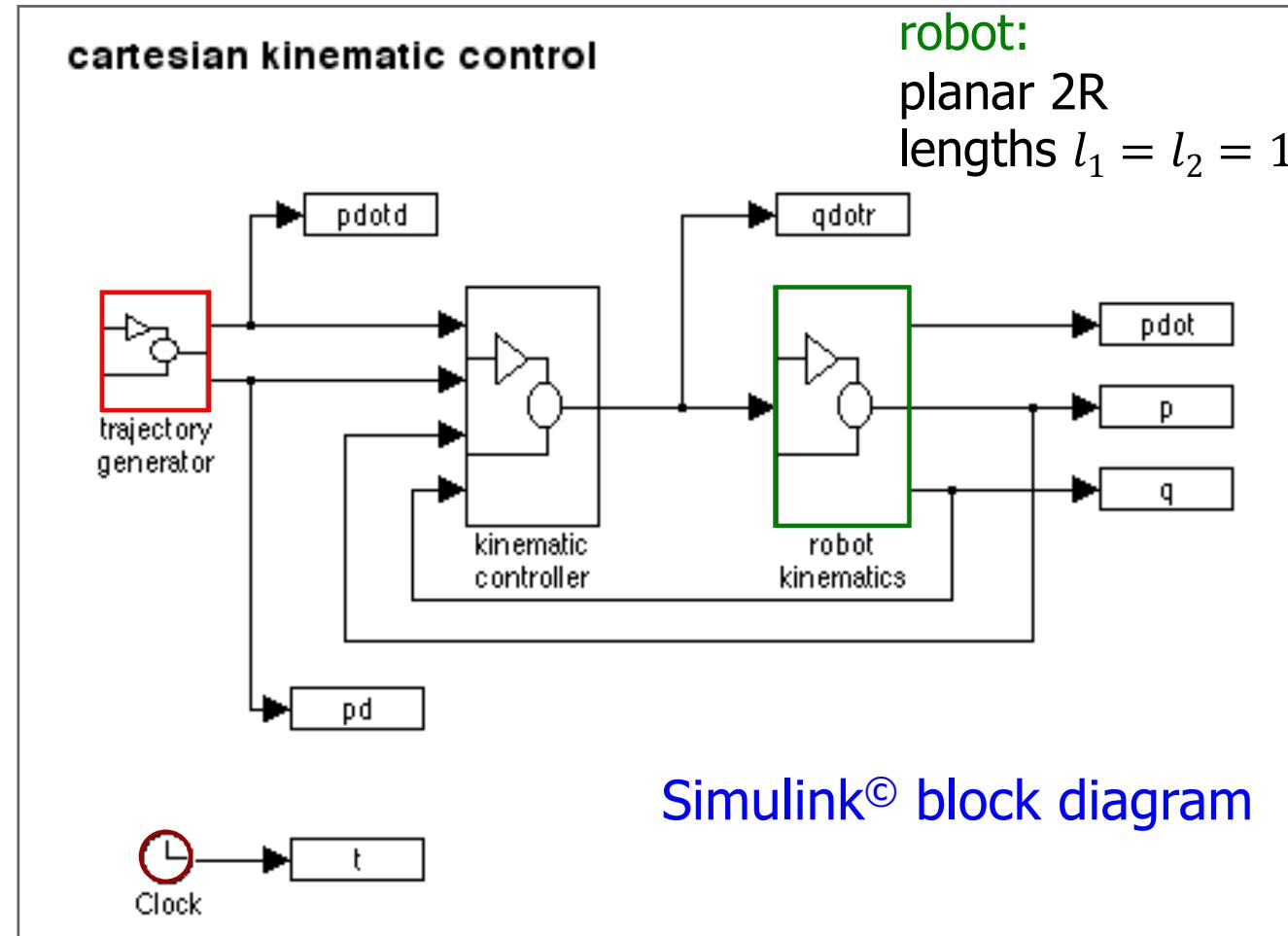
Simulation

features of kinematic control laws



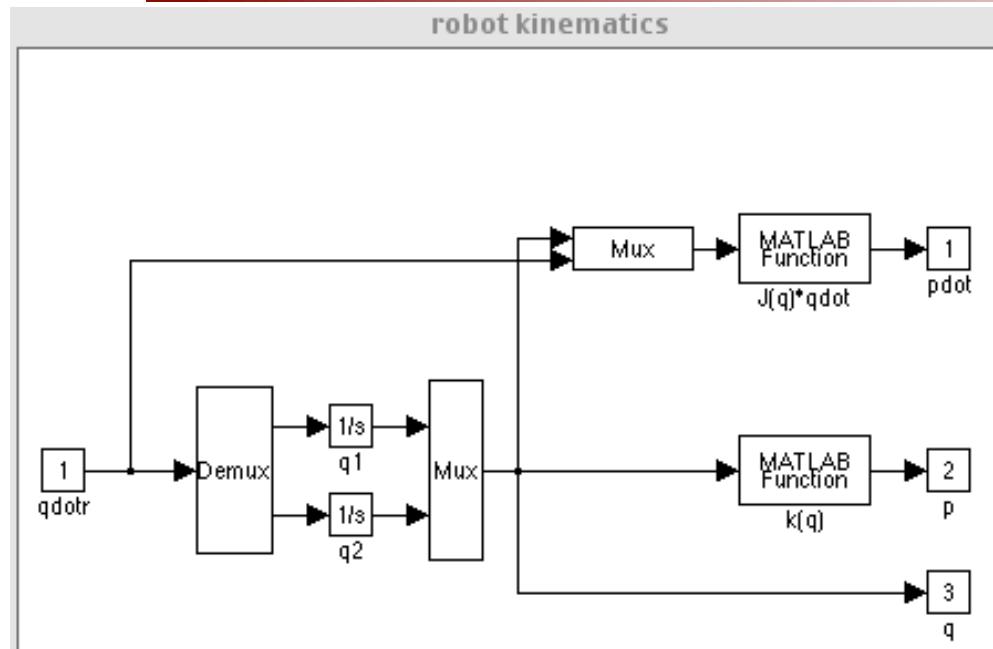
desired reference trajectory:
two types of tasks
1. straight line
2. circular path
both with constant speed

numerical integration method:
fixed step
Runge-Kutta
at 1 msec





Simulink blocks



calls to Matlab functions

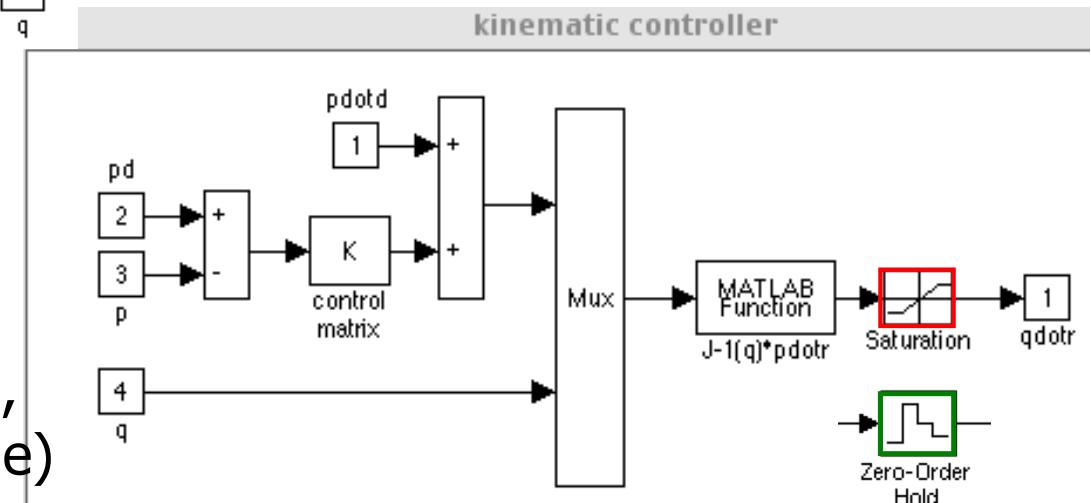
$k(q)=\text{dirkin}$ (user)

$J(q)=\text{jac}$ (user)

$J^{-1}(q)=\text{inv(jac)}$ (library)

- a **saturation** (for task 1.) or a **sample and hold** (for task 2.) added on joint velocity commands
- **system initialization** of kinematics data, desired trajectory, initial state, and control parameters (in **init.m** file)

never put "numbers" inside the blocks !





Matlab functions

dirkin.m

```
function [p] = dirkin(q)
```

```
global l1 l2
```

```
px=l1*cos(q(1))+l2*cos(q(1)+q(2));  
py=l1*sin(q(1))+l2*sin(q(1)+q(2));
```

jac.m

```
function [J] = jac(q)
```

```
global l1 l2
```

```
J(1,1)=-l1*sin(q(1))-l2*sin(q(1)+q(2));  
J(1,2)=-l2*sin(q(1)+q(2));  
J(2,1)=l1*cos(q(1))+l2*cos(q(1)+q(2));  
J(2,2)=l2*cos(q(1)+q(2));
```

init.m

```
% controllo cartesiano di un robot 2R  
% initialization  
  
clear all; close all  
global l1 l2  
  
% lunghezze bracci robot 2R  
  
l1=1; l2=1;  
  
% velocità cartesiana desiderata (costante)  
  
vxd=0; vyd=0.5;  
  
% tempo totale  
  
T=2;  
  
% configurazione desiderata iniziale  
  
q1d0=-45*pi/180; q2d0=135*pi/180;  
  
pd0=dirkin([q1d0 q2d0]');  
pxd0=pd0(1); pyd0=pd0(2);  
  
% configurazione attuale del robot  
  
q10=-45*pi/180; q20=90*pi/180;  
  
p0=dirkin([q10 q20]');  
  
% matrice dei guadagni cartesiani  
  
K=[20 20]; K=diag(K);  
  
%saturazioni di velocità ai giunti (input in deg/sec, convertito in rad/sec)  
  
vmax1=120*pi/180; vmax2=90*pi/180;
```

init.m
script
(for task 1.)



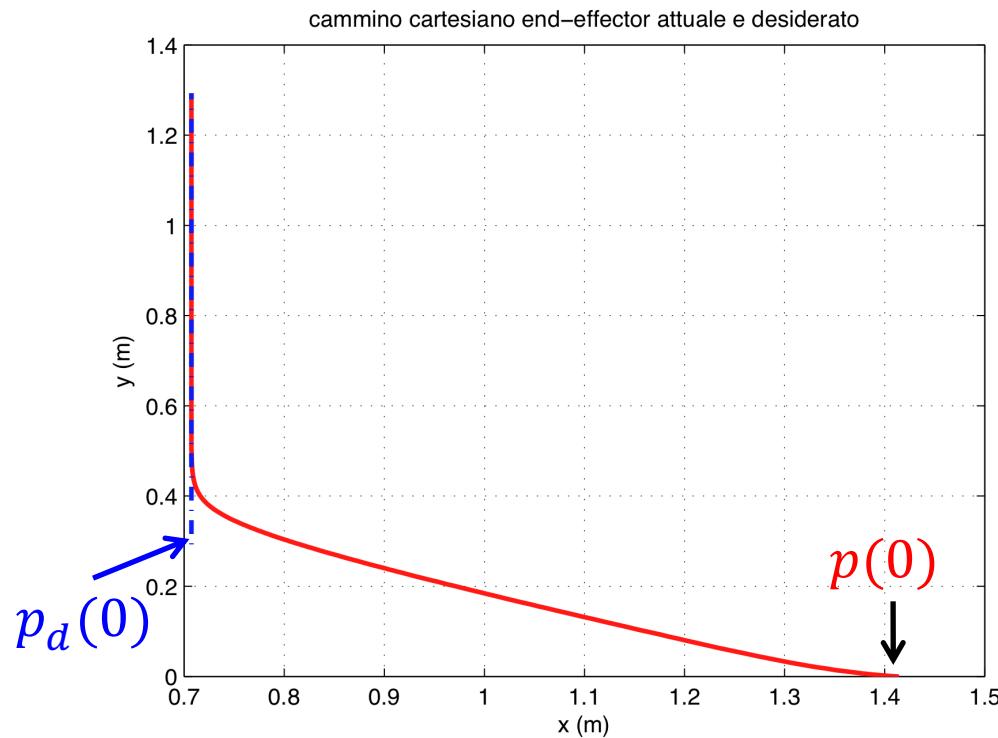
Simulation data for task 1

- straight line path with constant velocity
 - $x_d(0) = 0.7 \text{ m}$, $y_d(0) = 0.3 \text{ m}$; $v_{d,y} = 0.5 \text{ m/s}$, for $T = 2 \text{ s}$
- large initial error on end-effector position
 - $q(0) = (-45^\circ, 90^\circ) \Rightarrow e_p(0) = (-0.7, 0.3) \text{ m}$
- Cartesian control gains
 - $K_p = \text{diag}\{20, 20\}$
- (a) without joint velocity command saturation
- (b) with saturation ...
 - $v_{max,1} = 120^\circ/\text{s}$, $v_{max,2} = 90^\circ/\text{s}$

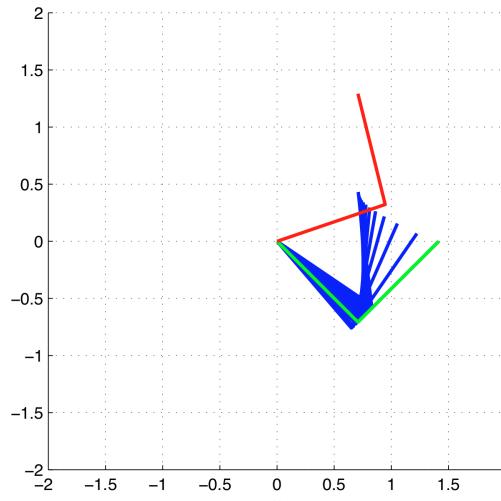


Results for task 1a

straight line: initial error, **no saturation**

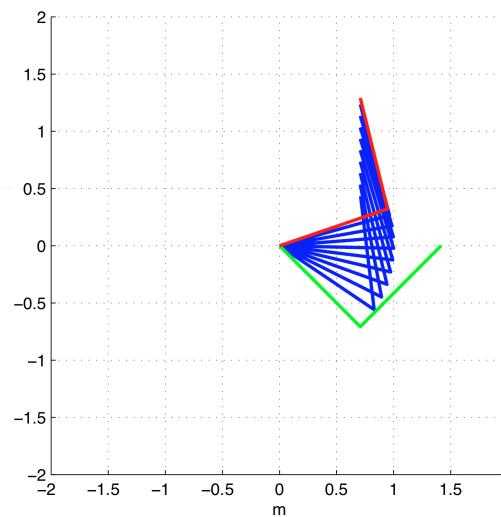


path executed by the
robot end-effector
(**actual** and **desired**)



initial
transient
phase
(about 0.2 s)

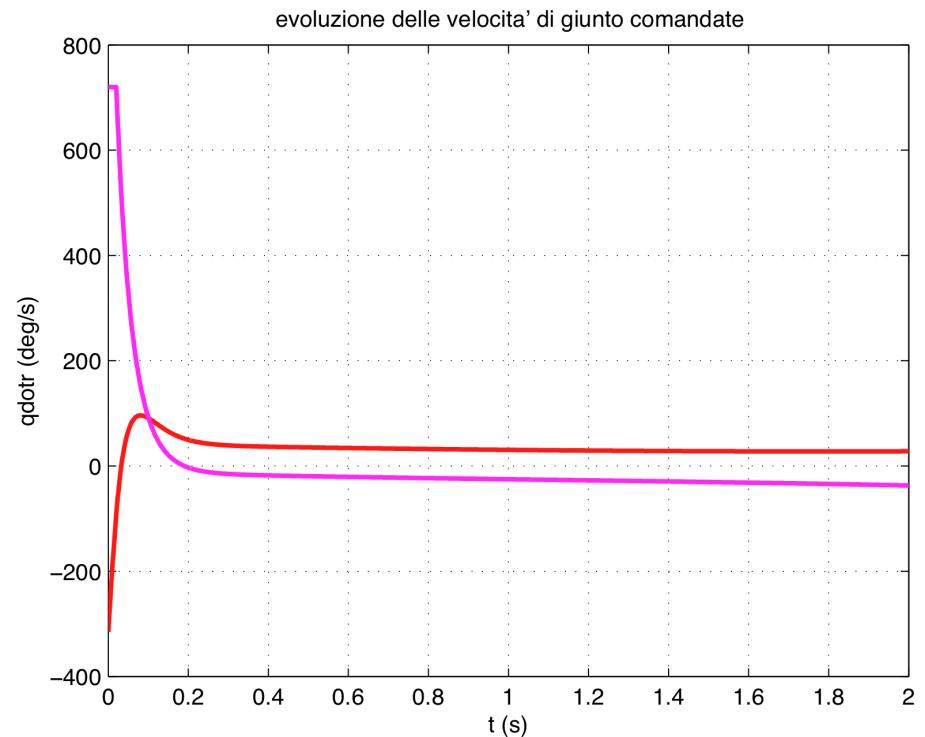
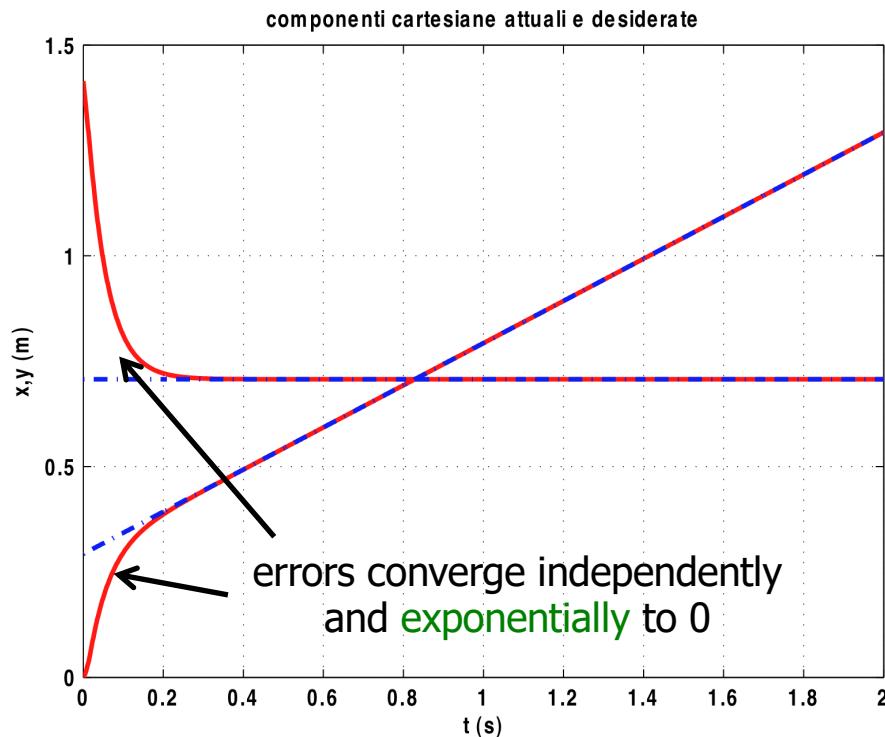
stroboscopic view of motion
(start and **end** configurations)



trajectory
following
phase
(about 1.8 s)

Results for task 1a (cont)

straight line: initial error, **no** saturation



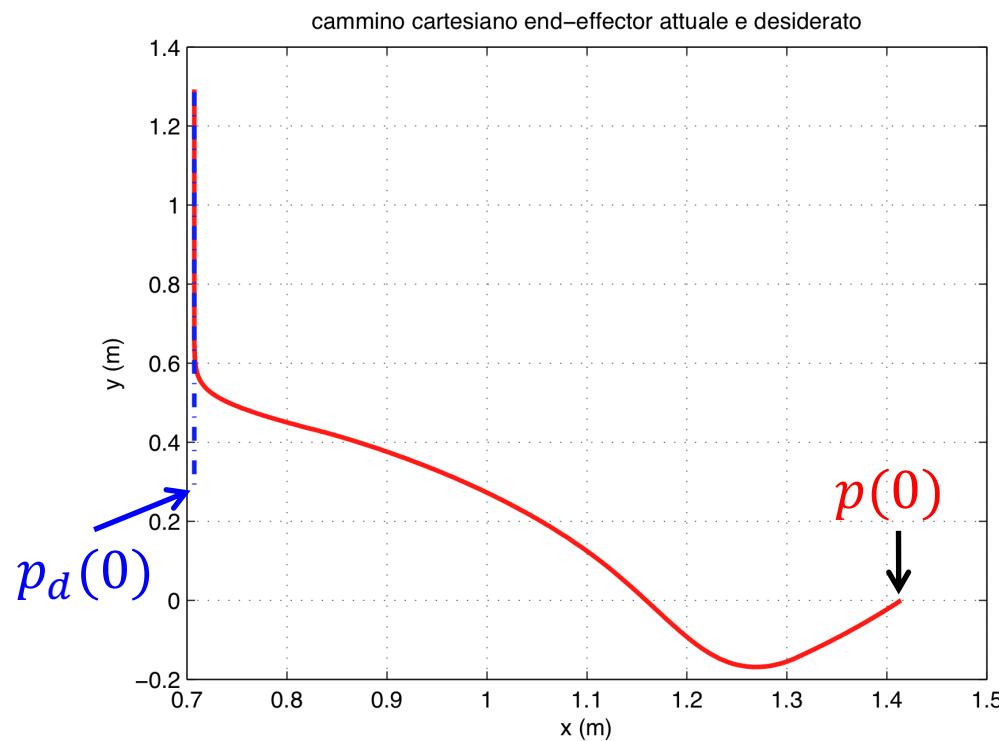
p_x, p_y actual and desired

control inputs $\dot{q}_{r1}, \dot{q}_{r2}$

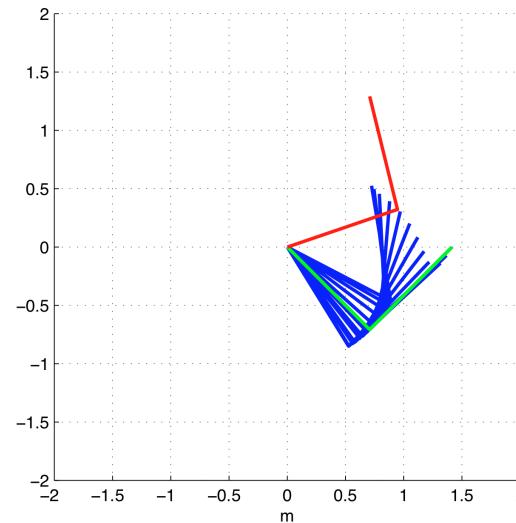


Results for task 1b

straight line: initial error, **with** saturation

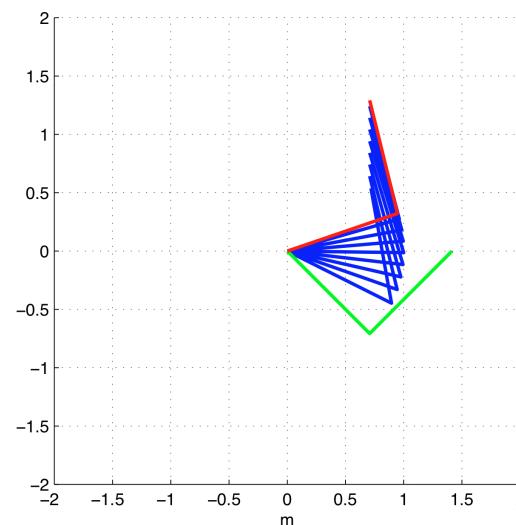


path executed by the
robot end-effector
(**actual** and **desired**)



initial
transient
phase
(about 0.5 s)

stroboscopic view of motion
(start and **end** configurations)

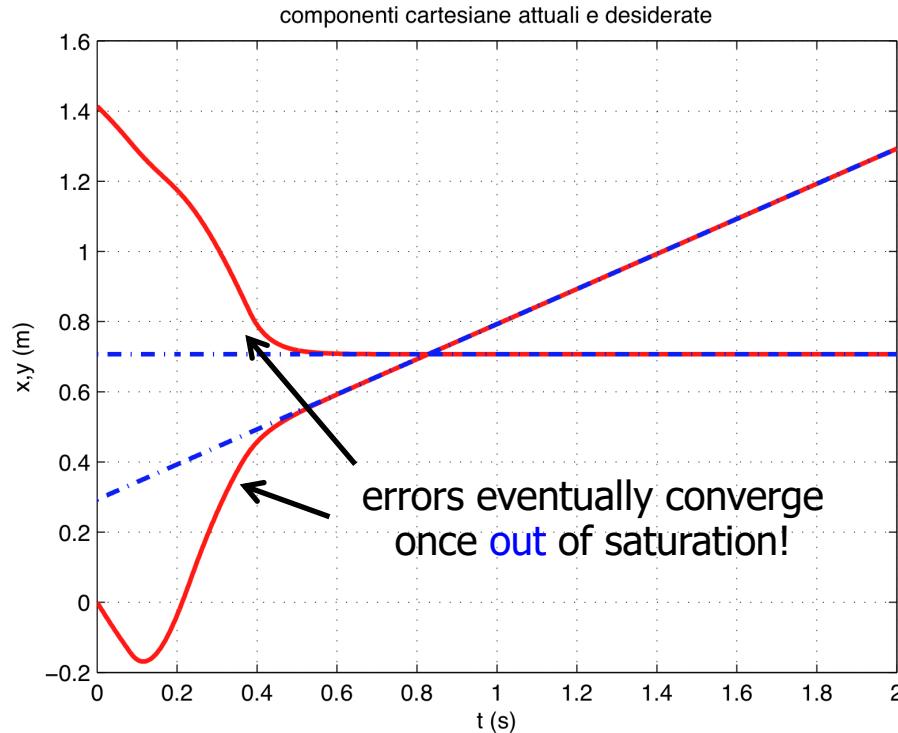


trajectory
following
phase
(about 1.5 s)

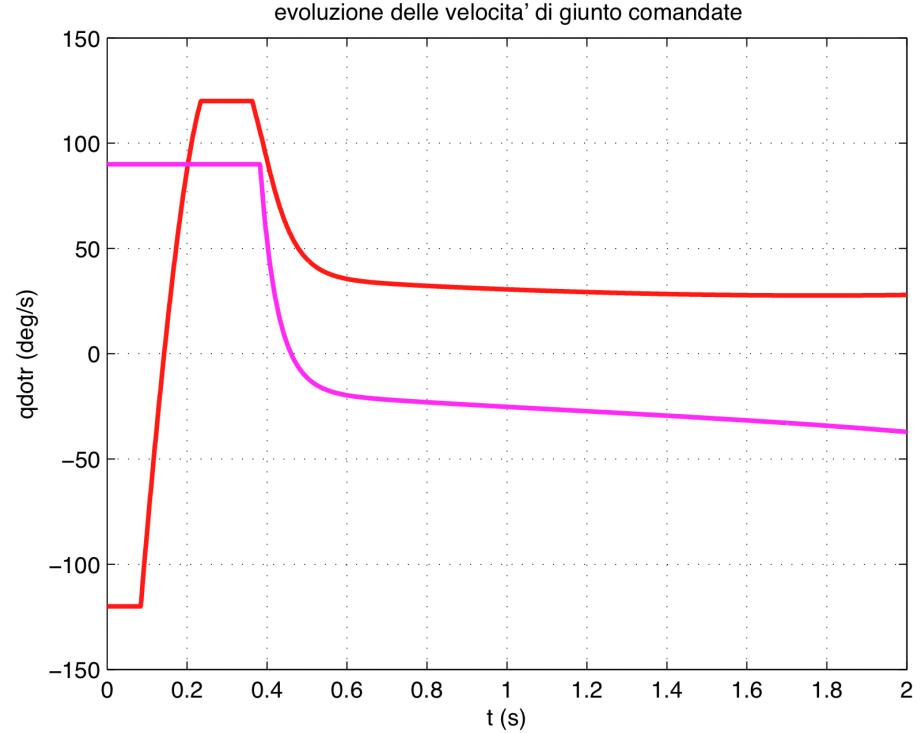


Results for task 1b (cont)

straight line: initial error, **with** saturation



p_x, p_y **actual** and **desired**



control inputs $\dot{q}_{r1}, \dot{q}_{r2}$
(saturated at $\pm v_{max,1}, \pm v_{max,2}$)



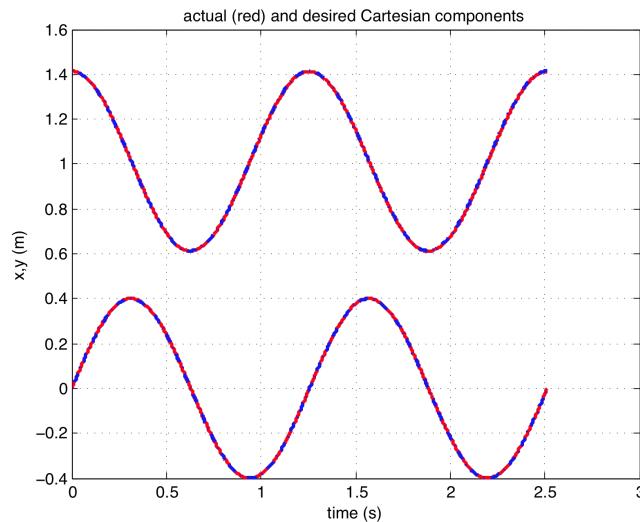
Simulation data for task 2

- **circular** path with constant velocity
 - centered at $(1.014, 0)$ with radius $R = 0.4$ m;
 - $v = 2$ m/s, performing **two** rounds $\Rightarrow T \approx 2.5$ s
- zero initial error on Cartesian position ("match")
 - $q(0) = (-45^\circ, 90^\circ) \Rightarrow e_p(0) = 0$
- (a) ideal **continuous** case (1 kHz), even **without** feedback
- (b) **with** sample and hold (ZOH) of $T_{hold} = 0.02$ s (joint velocity command updated at 50 Hz), but **without** feedback
- (c) as before, but **with** Cartesian feedback using the gains
 - $K_p = \text{diag}\{25, 25\}$

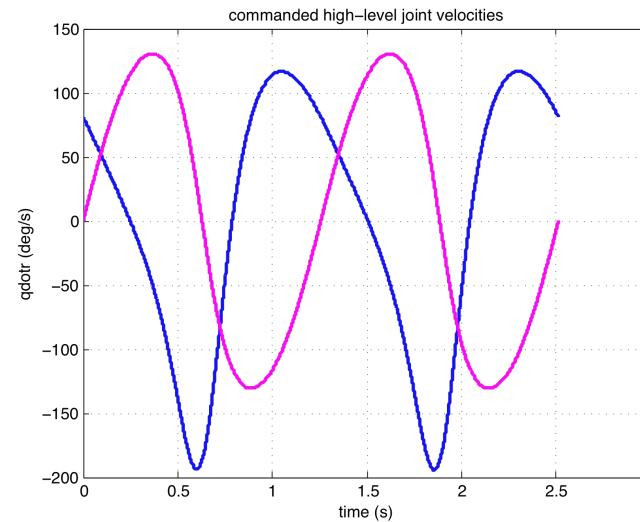


Results for task 2a

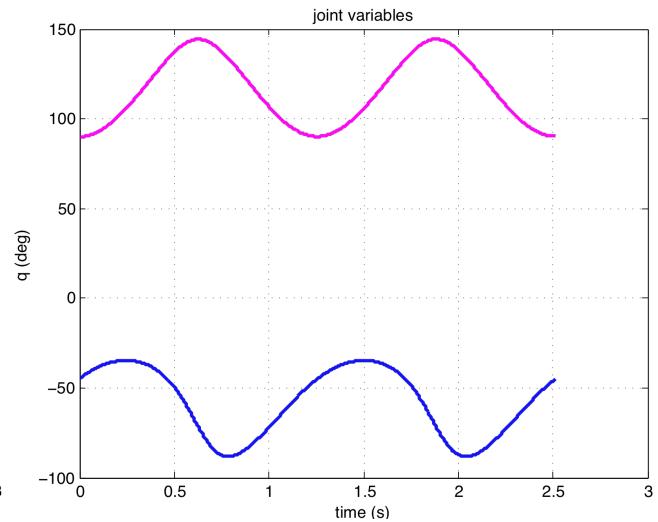
circular path: no initial error, **continuous** control (ideal case)



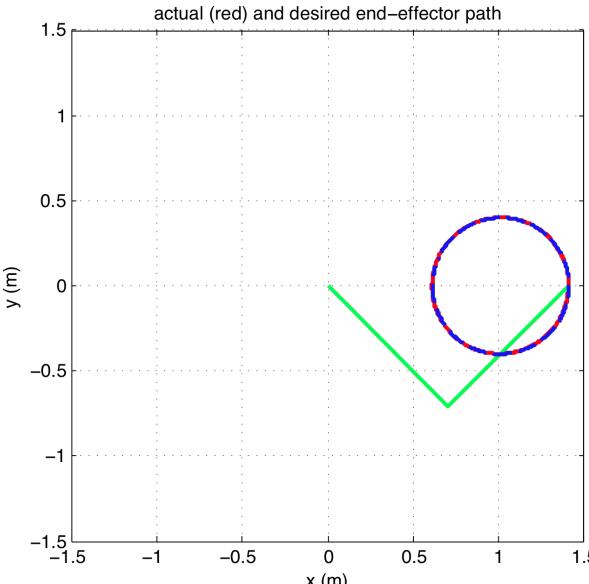
p_x, p_y actual and desired



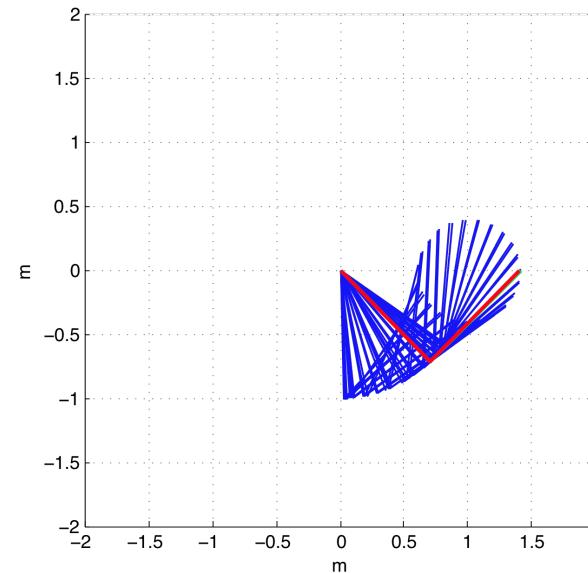
control inputs $\dot{q}_{r1}, \dot{q}_{r2}$



joint variables q_1, q_2



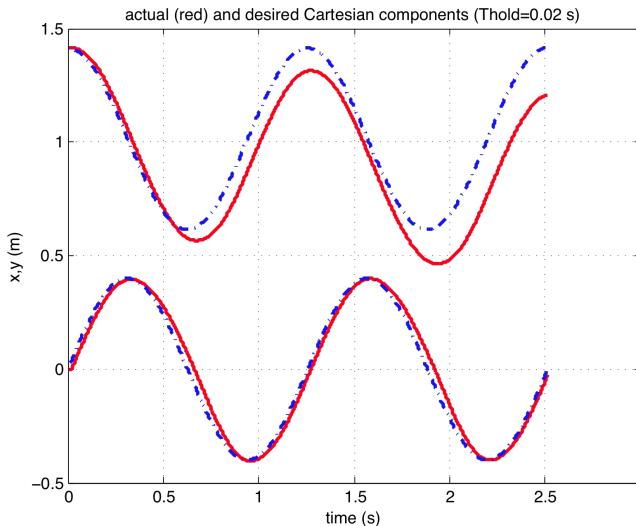
zero tracking
error is kept
at all times



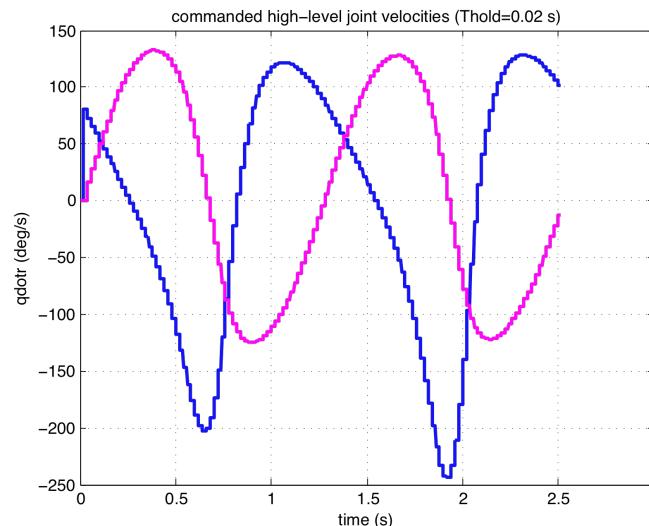


Results for task 2b

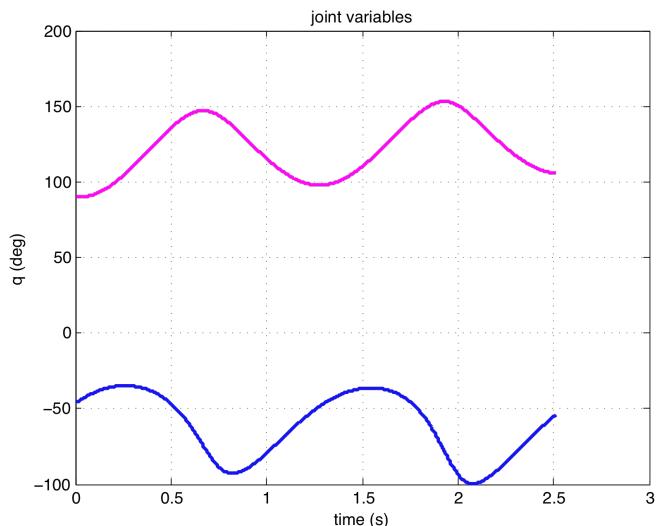
circular path: no initial error, **ZOH** at 50 Hz, **no feedback**



p_x, p_y actual and desired

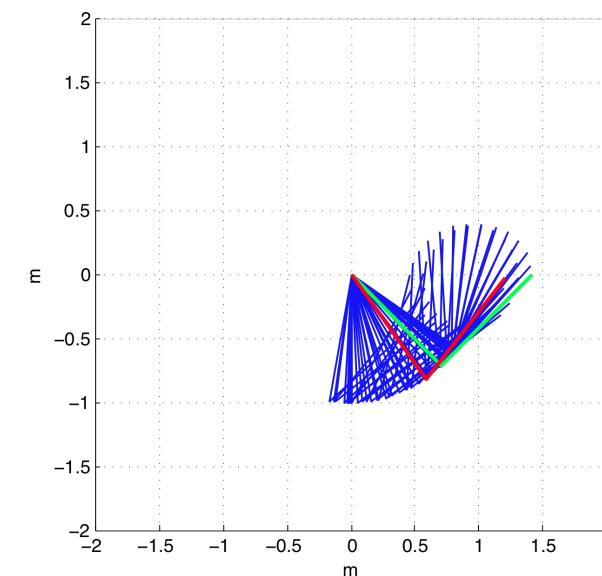
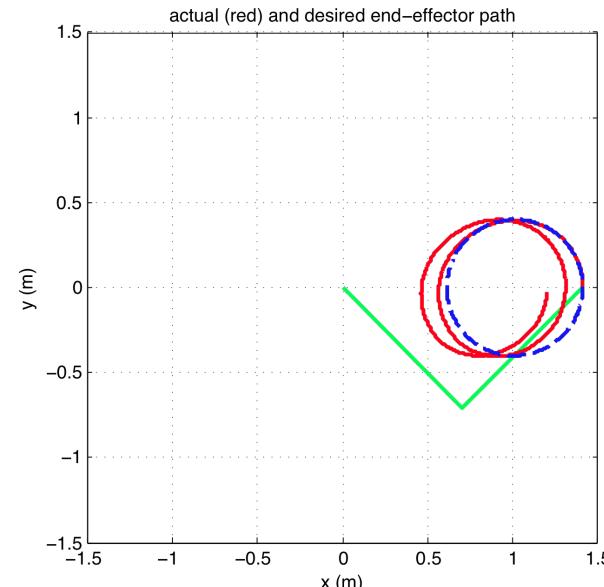


control inputs $\dot{q}_{r1}, \dot{q}_{r2}$



joint variables q_1, q_2

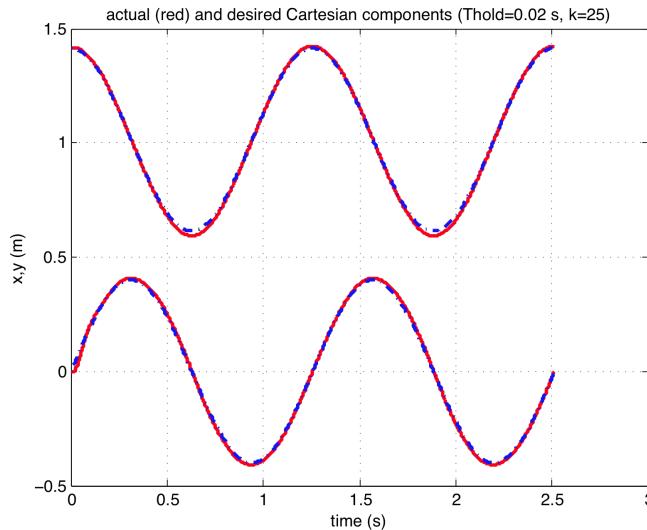
a drift occurs along the path due to the "linearization error" along the path tangent



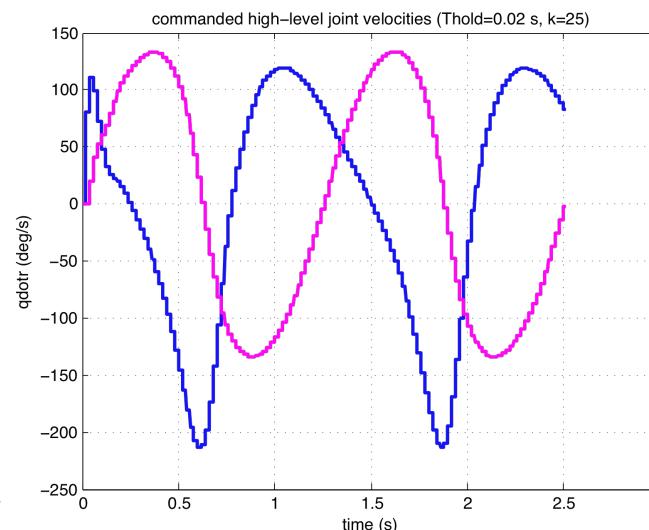


Results for task 2c

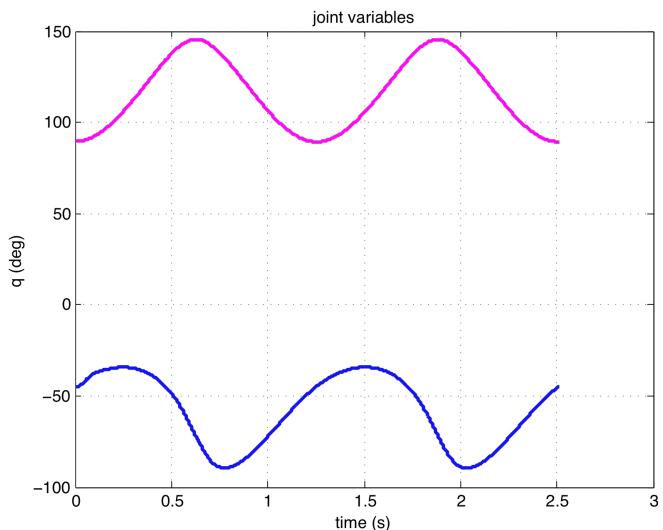
circular path: no initial error, **ZOH** at 50 Hz, **with feedback**



p_x, p_y actual and desired

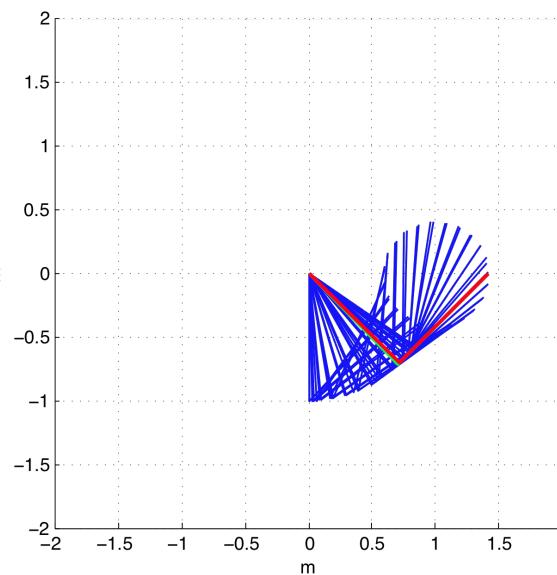
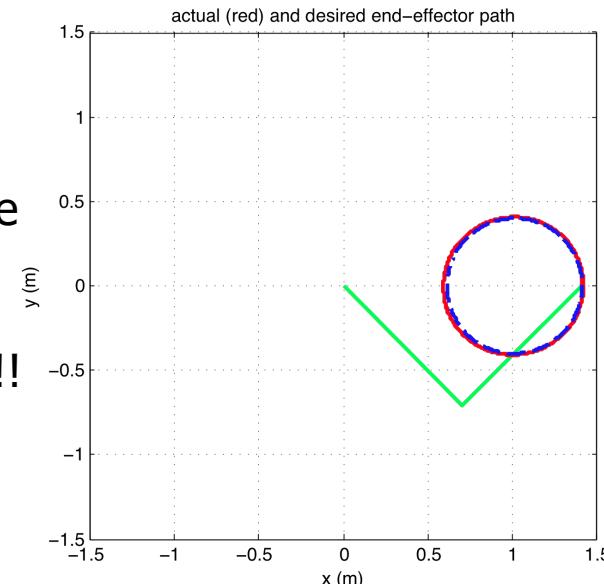


control inputs $\dot{q}_{r1}, \dot{q}_{r2}$



joint variables q_1, q_2

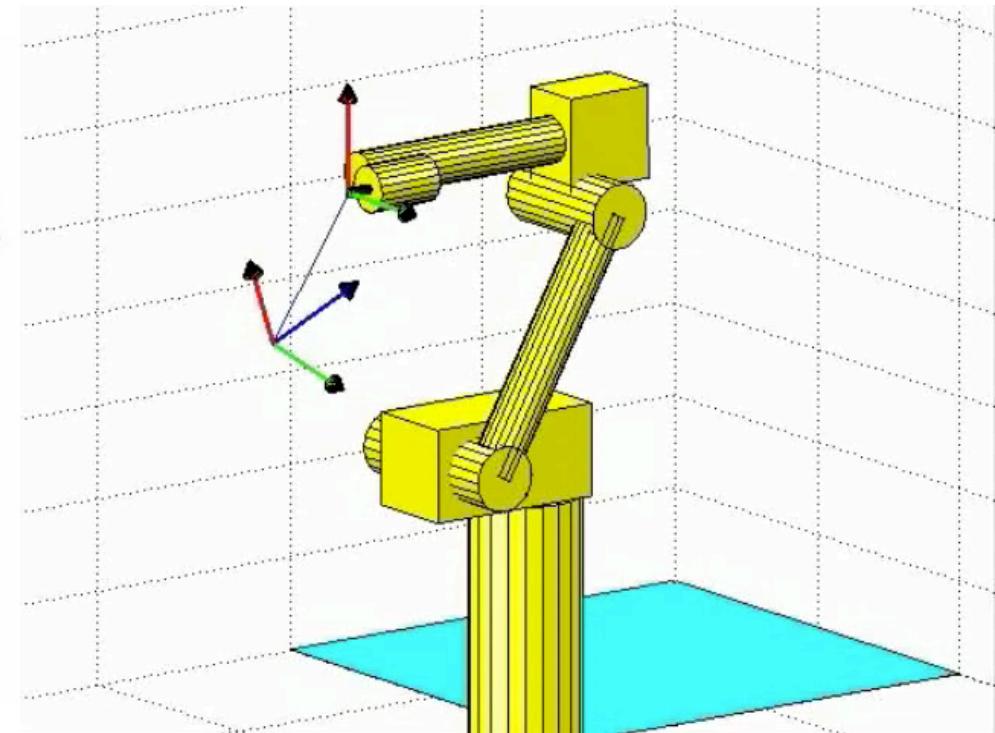
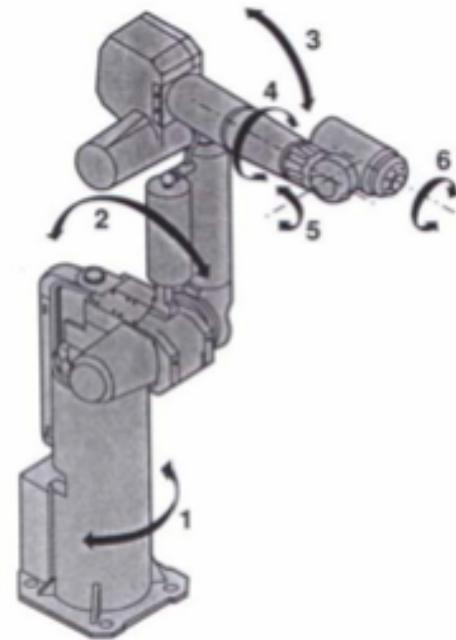
(almost) the same performance of the continuous case is recovered!!



note however that larger P gains will eventually lead to unstable behavior (see: stability problems for discrete-time control systems)



3D simulation



video

kinematic control of Cartesian motion of Fanuc 6R (Arc Mate S-5) robot
simulation and visualization in Matlab



Kinematic control of KUKA LWR



video

Discrete-Time Redundancy Resolution at the Velocity Level with Acceleration/Torque Optimization Properties

Fabrizio Flacco Alessandro De luca

Robotics Lab, DIAG
Sapienza University of Rome

September 2014

kinematic control of Cartesian motion with redundancy exploitation
velocity vs. acceleration level