Intelligent Robot Control

Lecture 2: Trajectory planning

Tadej Petrič

tadej.petric@ijs.si

Institut
"Jožef Stefan"
Ljubljana, Slovenija

Trajectory planning

Trajectory planning is part of the task planning procedure. Typically, the robot task includes some actions, which are described as a sequence of poses or robot configurations. planning the motion from one to another pose is crucial for the correct operation of the robot.

When planning the motion of the robot we have to specify two aspect of the desired trajectory:

- Geometric path, which defines where the robot should be.
- Timing, which defines the time evolution of the path (when the robot has to be at a certain position)

and consider all constraints (continuity, smoothness, . . .) important for the task or due to the robot itself (joint limit, actuator limits, . . .).

Trajectories

Geometric path can be defined in the task space or in the joint space:

$$\boxed{\boldsymbol{x} = \boldsymbol{x}(s)} \qquad \text{or} \qquad \boxed{\boldsymbol{q} = \boldsymbol{q}(s)}$$

where s is the path parameter. Definig s as a function of time (**timing**)

$$s = s(t)$$

we get the trajectory.

In practice we want to have smooth motion and therefore, the time evolution has to be such that the trajectory is a continous function up to a given order of derivation (usually at least bounded accelerations).

Discrete path: point-to-point (PTP) or multiple points (via points, nodes).

Continuous path: linear, circular, or other analytic functions.

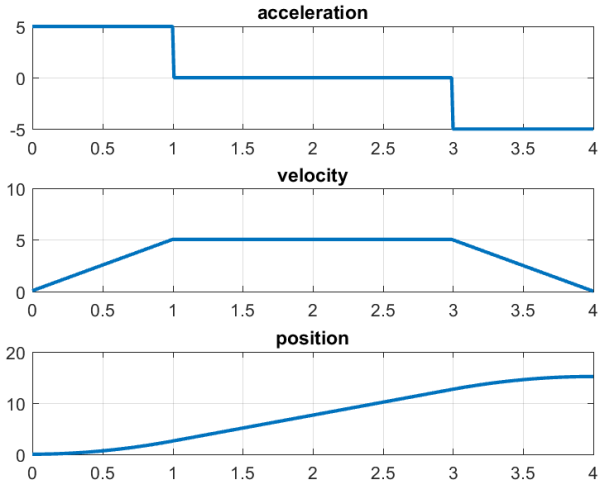
Polynomial trajectories

The trajectory is specified by defining some points and interpolation between them using polynomial functions of proper degree n.

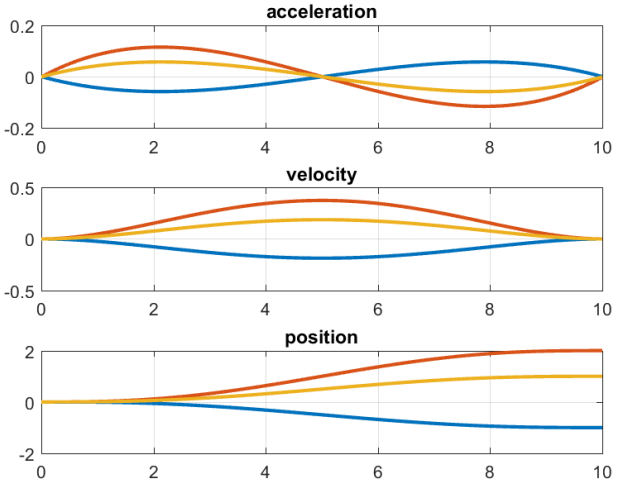
$$\boldsymbol{x}(t) = \boldsymbol{a}_0 + \boldsymbol{a}_1 t + \boldsymbol{a}_2 t^2 + \cdots + \boldsymbol{a}_n t^n$$

- Common in industrial robotics.
- Can be applied for Cartesian or joint space motion.
- The degree n defines the **smoothness**.
- Polinomial parameters calculated using boundary conditions in points.

Trapezoidal velocity profile



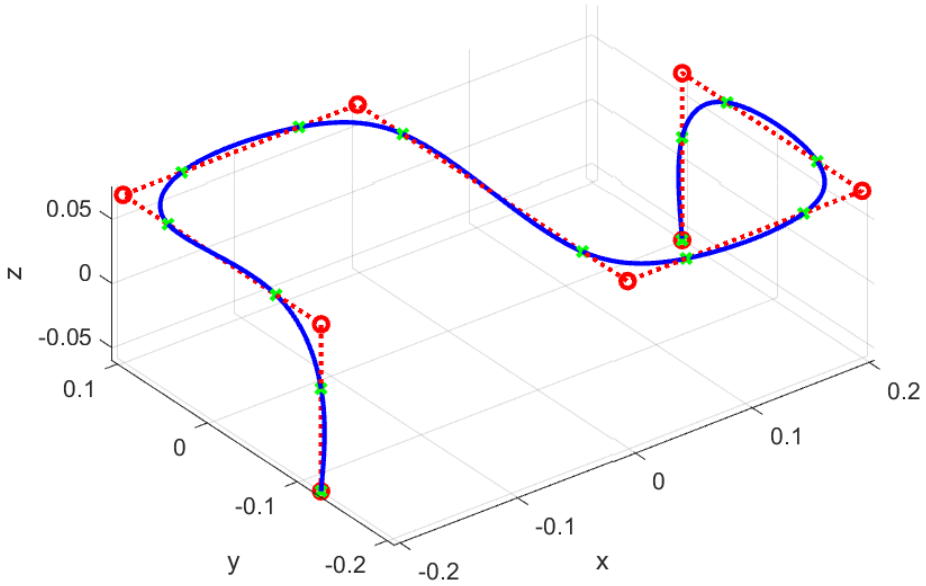
5^{th} order polynomial profile

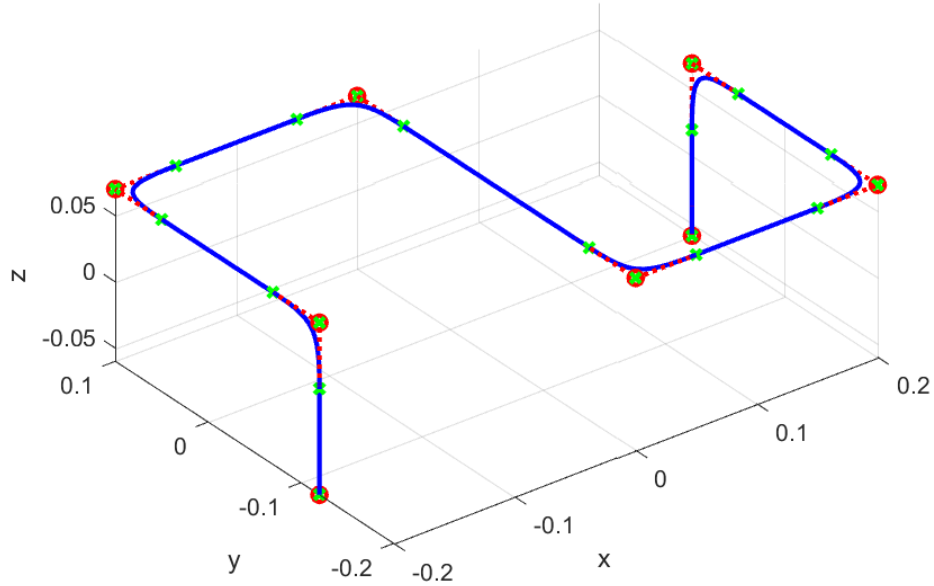


Spline trajectories

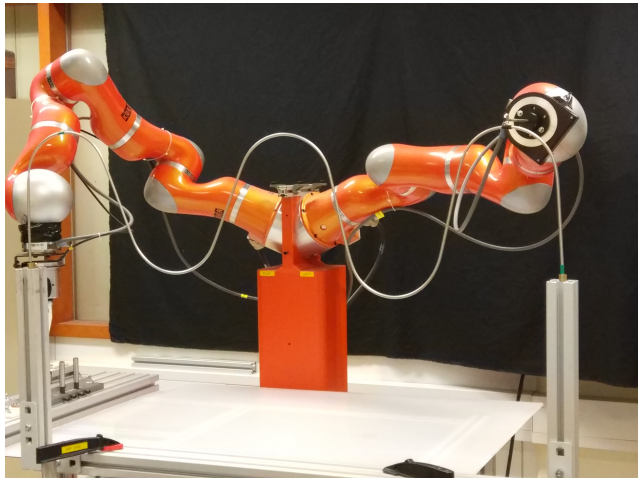
In general, defining path over n points requires a polynomial function of degree $n-1$. However, the resulting motion can have unnecessary large osculations between the points.

Solution: Use more **lower degree polynomials** representing path segments and glue them smoothly (using boundary conditions in intermediate points) into final path $\Rightarrow$ Splines.





Path parametrization using kernel functions



Task is to follow the wire. The path is obtained by **captureing a lot of points**.

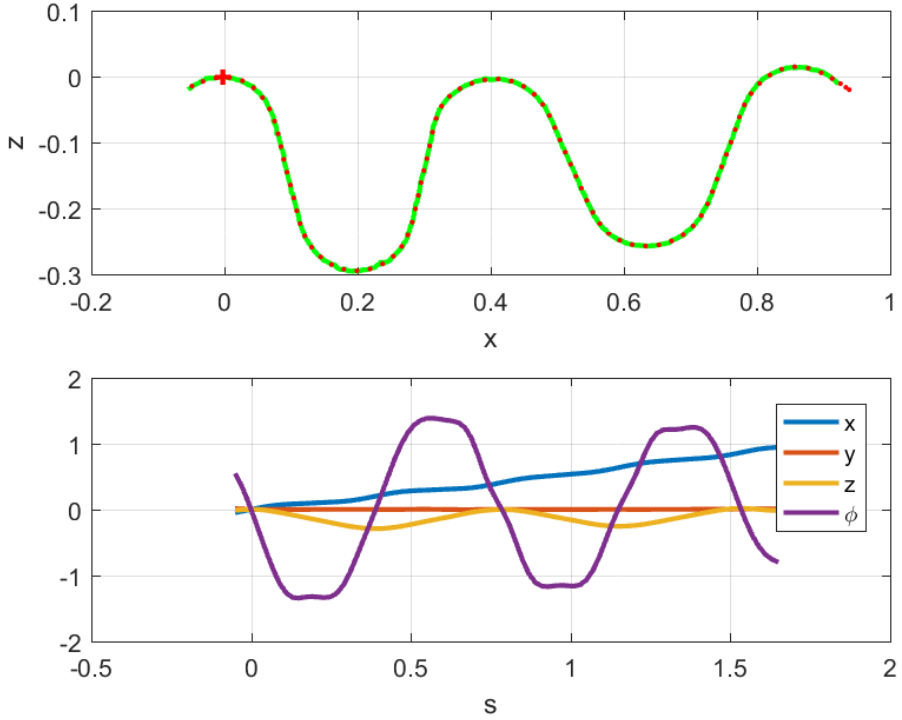
To obtain **computational efficient representation** of the path, an appropriate parametrization of captured points is needed. Parametrization has to assure required accuracy of the approximated path.

Gausian kernel functions:

$$\Psi_i(s) = e^{-h_i(s-c_i)^2}$$

$$\boldsymbol{x}(s) = \frac{\sum_{i=1}^{m} \boldsymbol{w}_i \Psi_i(s)}{\sum_{i=1}^{m} \Psi_i(s)}$$

Weights $\boldsymbol{w}_i$ are obtained by minimizing $\frac{1}{2}\|\boldsymbol{x}_m - \boldsymbol{x}(s)\|^2$, regression, or other methods.



What is robot control?

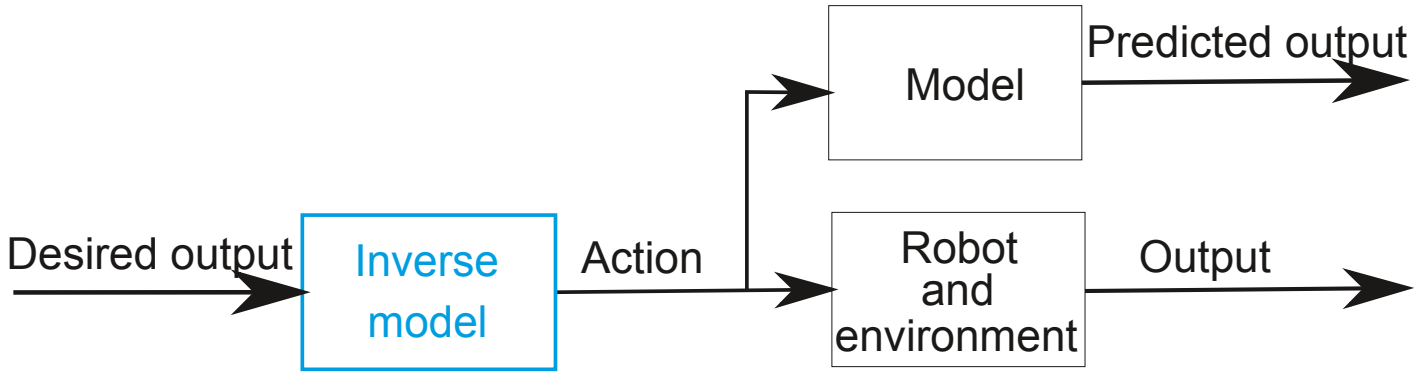
Robot control is the means by which the sensing and actions of a robot are coordinated.

Robot control is defining joint torques for actuators in joints necessary to execute the desired task.

The achievable performances depend on:

- control techniques used to solve control problem
- implementation of the control algorithms
- mechanical design of the robot
- configuration of the robot

Models in control



To **predict** the motion caused by action we need the model of the robot and the environment.

To **define the action** necessary for the desired motion we need the inverse model.

Using observations of the systems states to change actions $\Rightarrow$ Close-loop control

Robot control levels

Robot control is not just a close-loop control.

Control levels:

Strategic	Learning
	Task planning
Tactic	Motion coordination
	Trajectories generation
Executive	Movement execution
	Close-loop control

Higher control levels include more "intelligence".

Which levels are implemented depends on the task complexity.

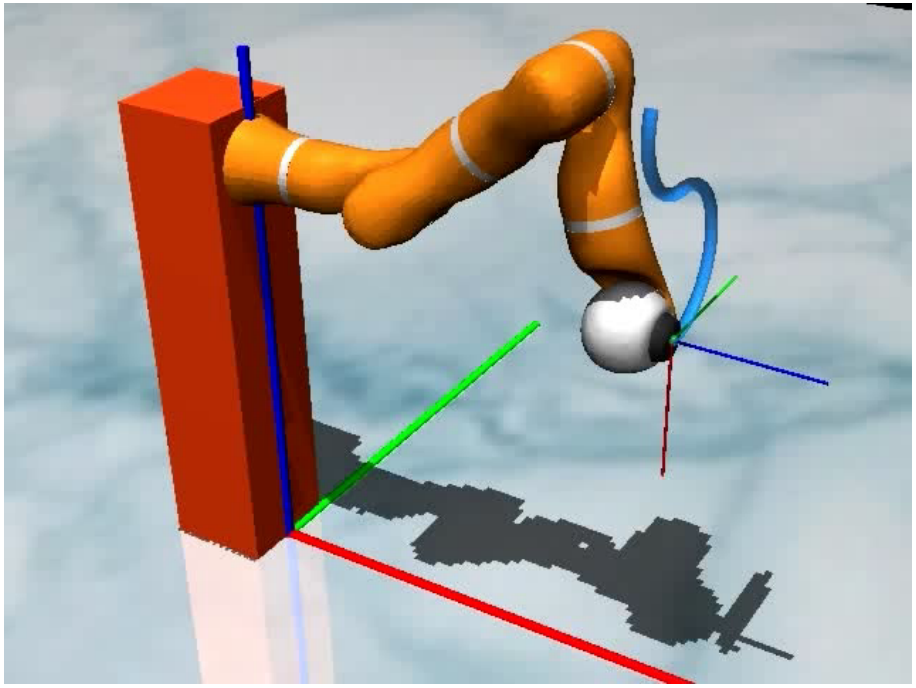
Higher task complexity $\Rightarrow$ Higher levels included

Control strategies

The selected control strategy influences the characteristics of the robot system.

The strategy selection depends on:

- robot task
  - ⋆ task workspace
  - ⋆ free motion or motion in contact
  - ⋆ constraints (e.g. velocity, acceleration, torques)
  - ⋆ ...
- mechanical structure of the robot manipulator
- actuators (electric, hydraulic, pneumatic)
- gears (ration, compliance,...)
- sensors (joint torques, external forces, vision, ...)
- ...



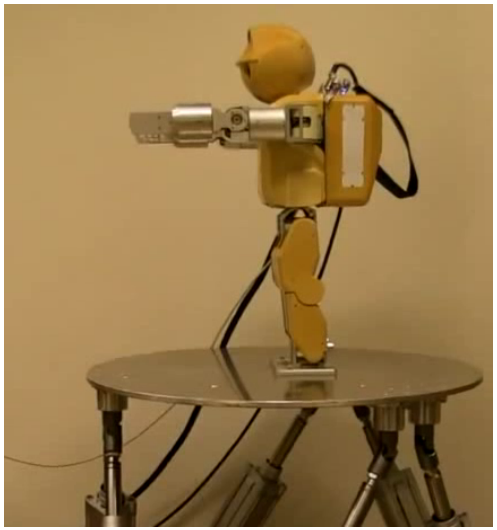
Robot control approaches

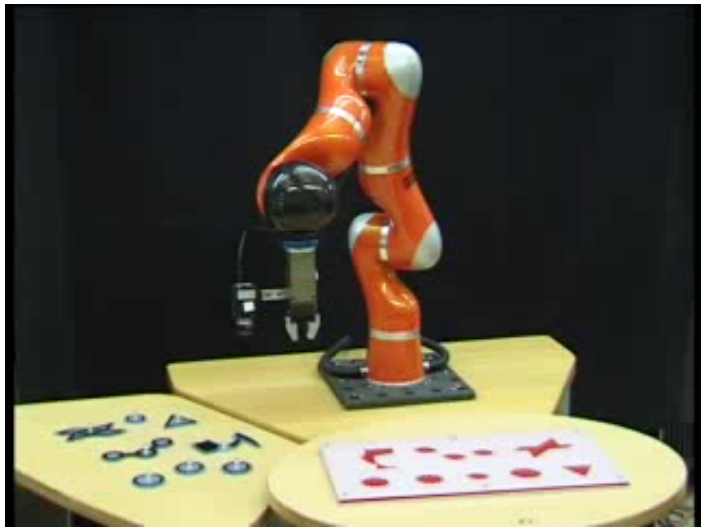
Reactive Control: Don't think, (re)act.

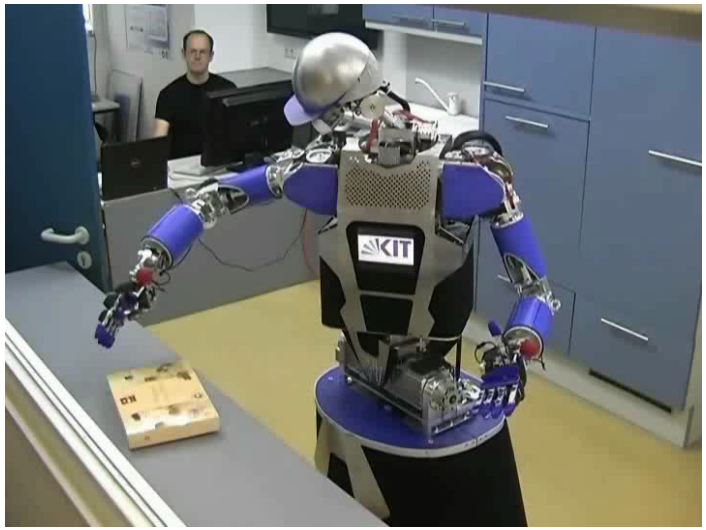
Deliberative (preplanned) Control: Think hard, act later.

Hybrid Control: Think and act separately and concurrently.

Behavior-Based Control: Think the way you act.





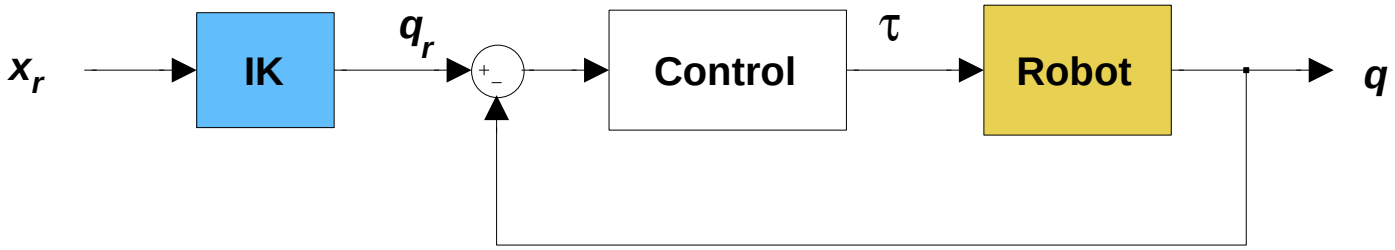


Basic robot control schemes

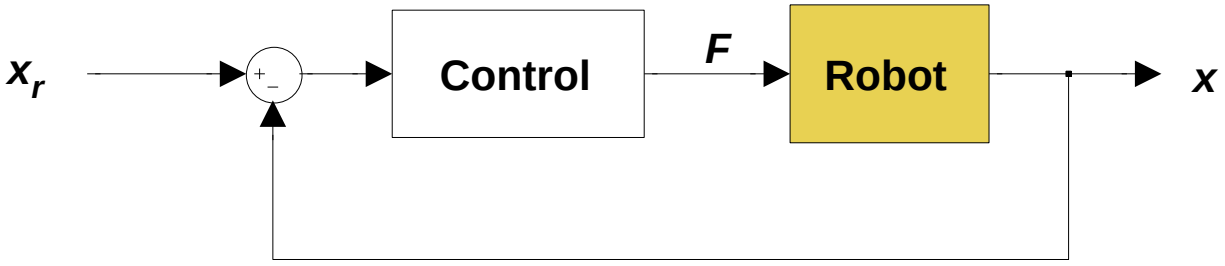
Typically for robots:

- Tasks are defined in operational workspace as end-effector motion or end-effector forces.
- Robot actions are generated in the joint space.

Joint space control:

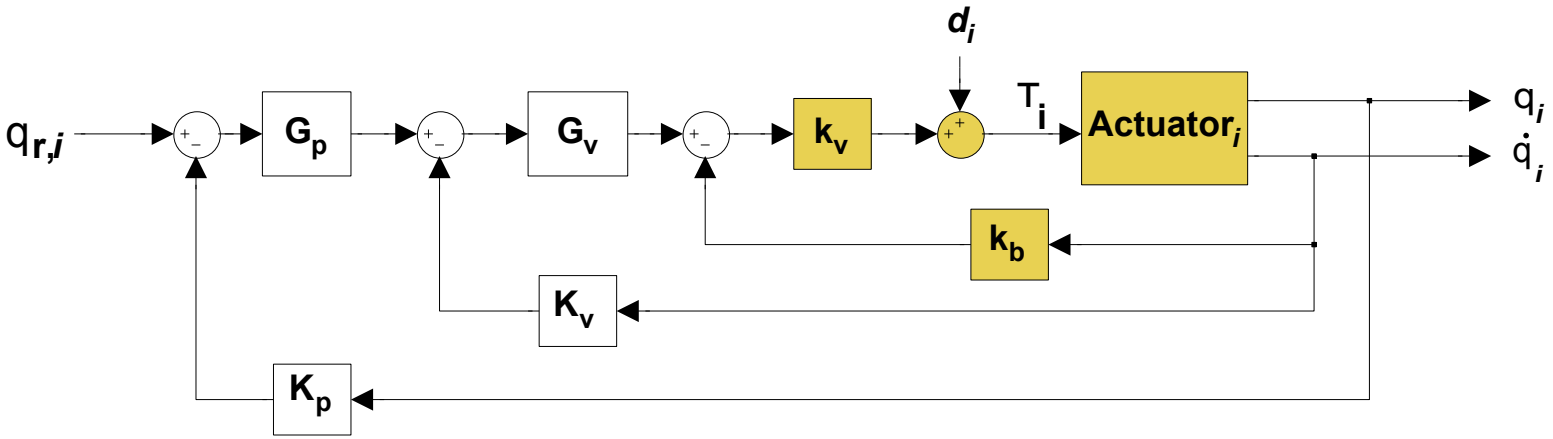


Task space control



Independent control

Basic robot control is the decentralized control: each robot joint is an independent servosystem and interactions between joint are considered as disturbances.



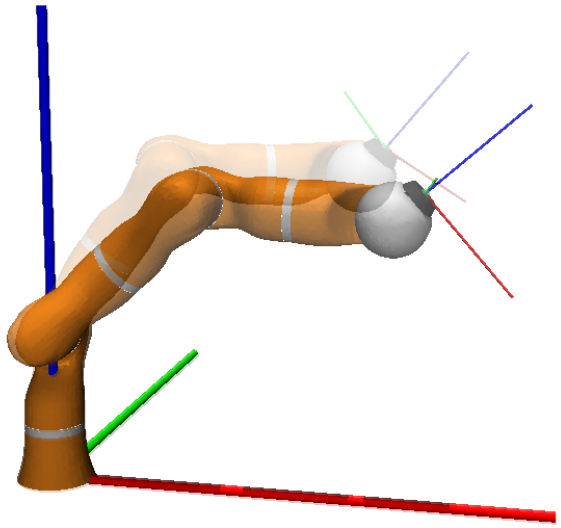
- Positioning and manipulation
- Standard techniques for control design (RL, LQR, . . .)
- Small interactions for high-ratio gears
- Widely used in industrial robots

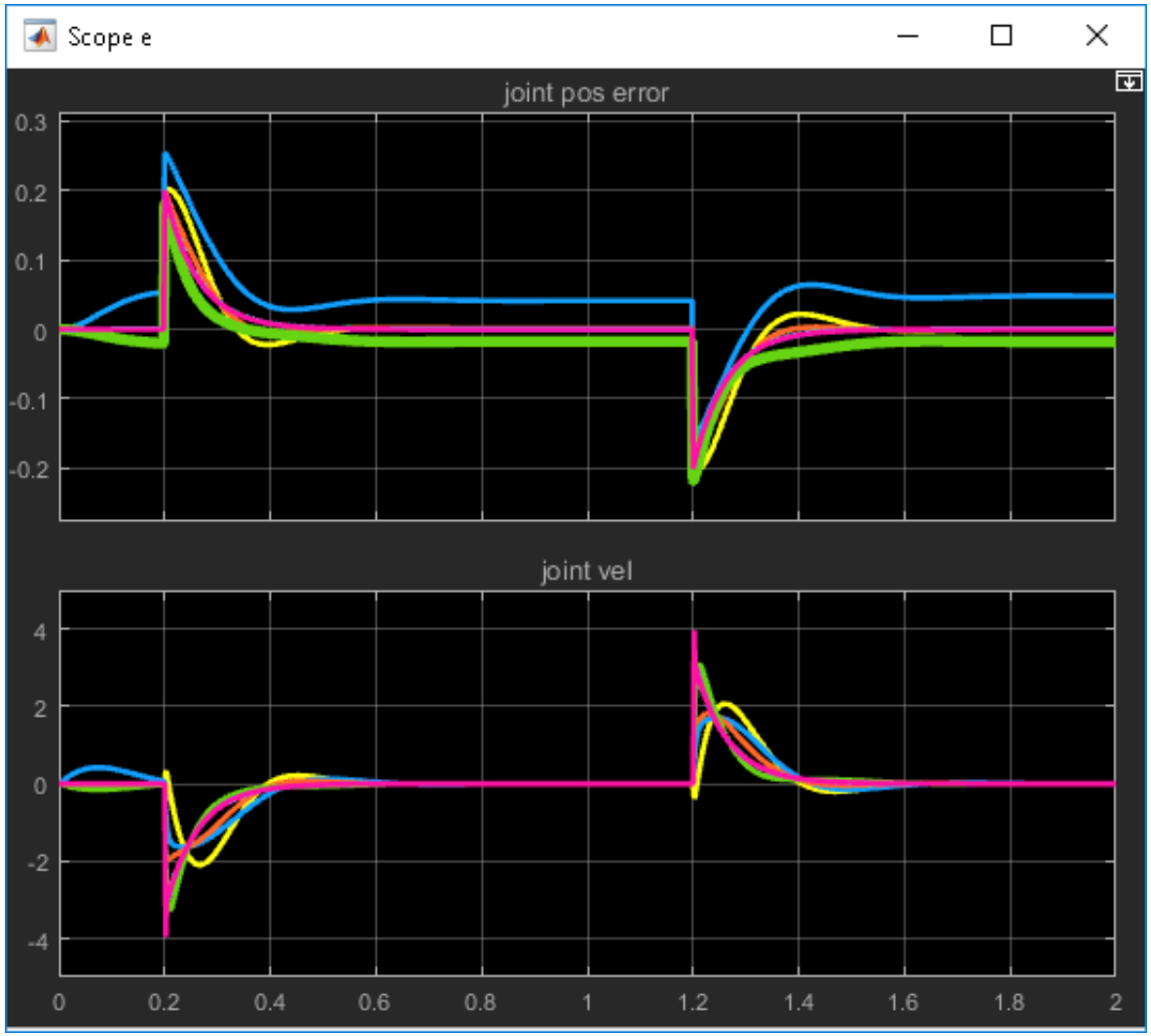
Independent control: Example KUKA LWR

PTP motion:

$$\Delta q_i = 0.2,\ i = 1, \ldots, 7$$

Same K_p and K_d gains for all joint controllers.

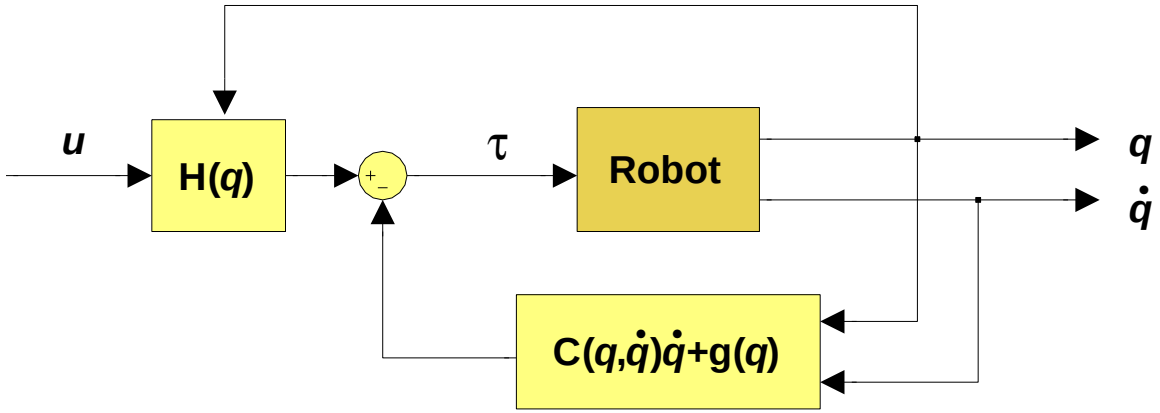




Inverse dynamics control

For high accuracy and fast motion the robot dynamics has to be considered in the control.

Computed torques technique:



$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{H}(q)\boldsymbol{u} + \mathbf{C}(q,\dot{q})\dot{q} + \boldsymbol{g}(q) \\ \boldsymbol{u} &= \ddot{\boldsymbol{q}} \end{aligned}$$

Due to compensation of nonlinearities and interactions in the close-loop the robot system is decomposed into independent subsystems and the control problem is reduced to a simpler problem of controlling independent linear systems.

Inverse dynamics control . . .

The dynamic close-loop properties are defined by the selection of the outer loop control algorithm. For example, using PD controller

$$\boldsymbol{u} = \ddot{\boldsymbol{q}}_r + \mathbf{K}_v \dot{\boldsymbol{e}}_q + \mathbf{K}_p \boldsymbol{e}_q$$

the close-loop dynamics of a system becomes

$$\boxed{\ddot{\boldsymbol{e}}_q + \mathbf{K}_v \dot{\boldsymbol{e}}_q + \mathbf{K}_p \boldsymbol{e}_q = 0}$$

Drawbacks:

- Exact dynamic robot model must be known
- High computational complexity

Solution: An approximate or partial model can be used. The errors are compensated by the outer-loop control algorithms. Suitable approaches:

- Variable structure control
- Adaptive control
- Independent joint compensation

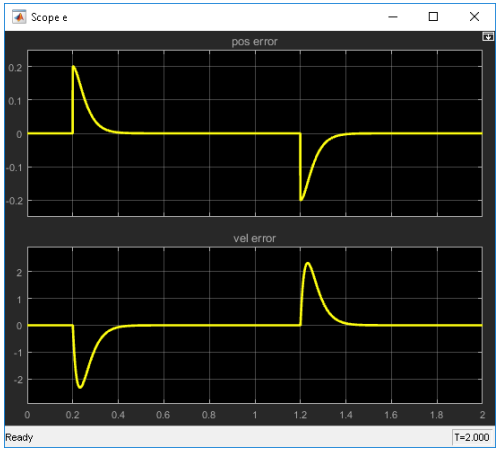
Inverse dynamics control: Example KUKA LWR

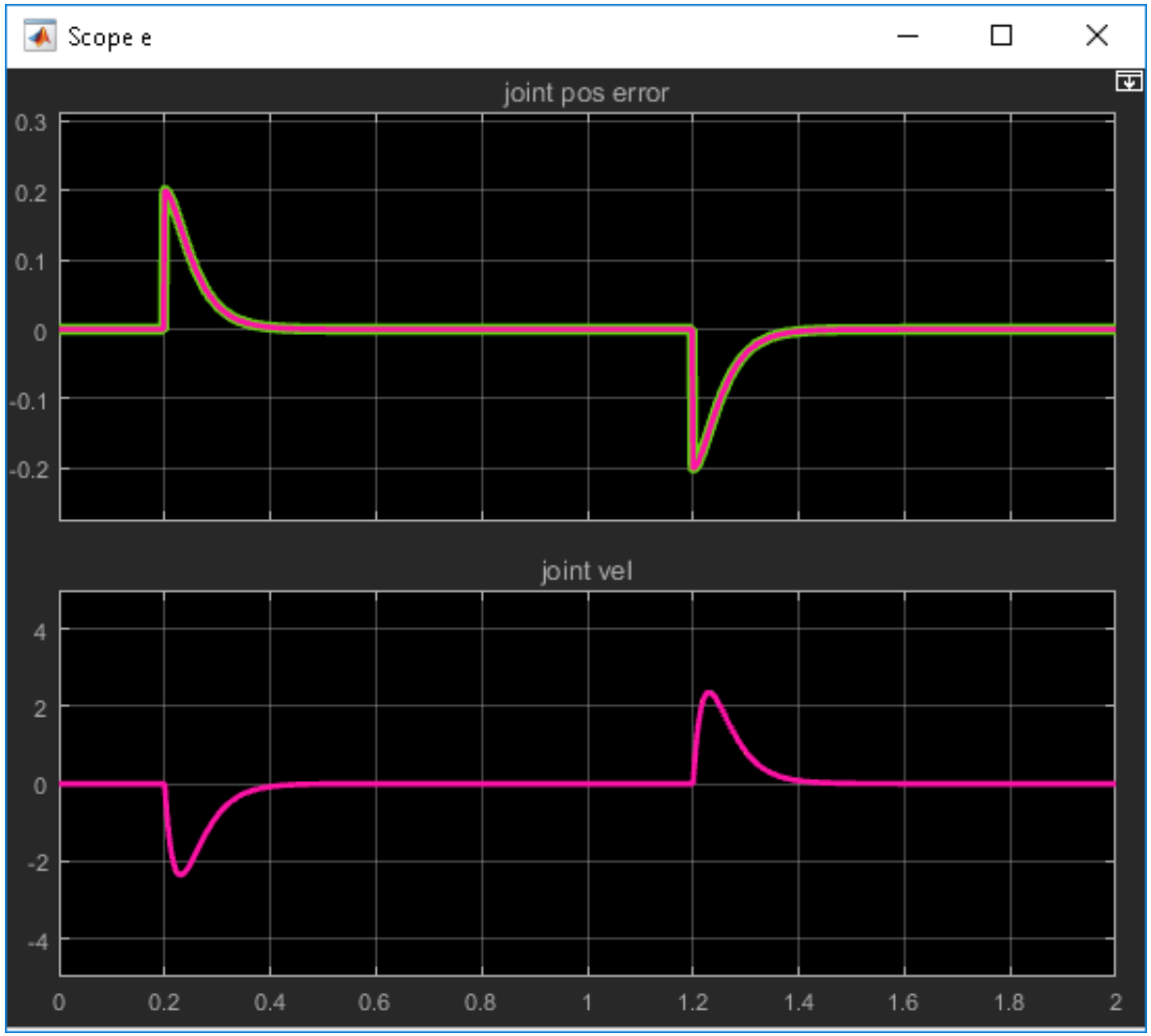
Joint space PTP motion:

$$\Delta q_i = 0.2,\ i = 1, \ldots, 7$$

Gains K_p and K_d for the desired close-loop behavior:

$$\ddot{e}_q + 2\sqrt{1000}\dot{e}_q + 1000 e_q = 0$$



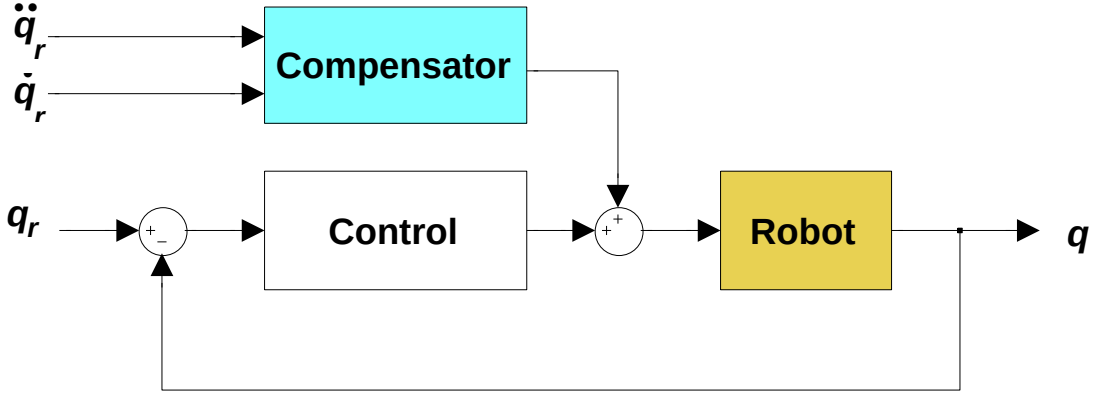


Compensators

- Some tasks require exact motion execution, i.e. good trajectory tracking for fast motion.
- Close-loop control schemes do not assure the required tracking accuracy.

$$\ddot{\boldsymbol{e}}_q + \mathbf{K}_v \dot{\boldsymbol{e}}_q + \mathbf{K}_p \boldsymbol{e}_q = 0$$

Tracking error can be reduced by using addition feedforward compensators.



- Better accuracy without affecting the stability of the system.
- It is necessary to know the system (models).
- Real system constraints can influence the accuracy of the system.

Compensators: Example KUKA LWR

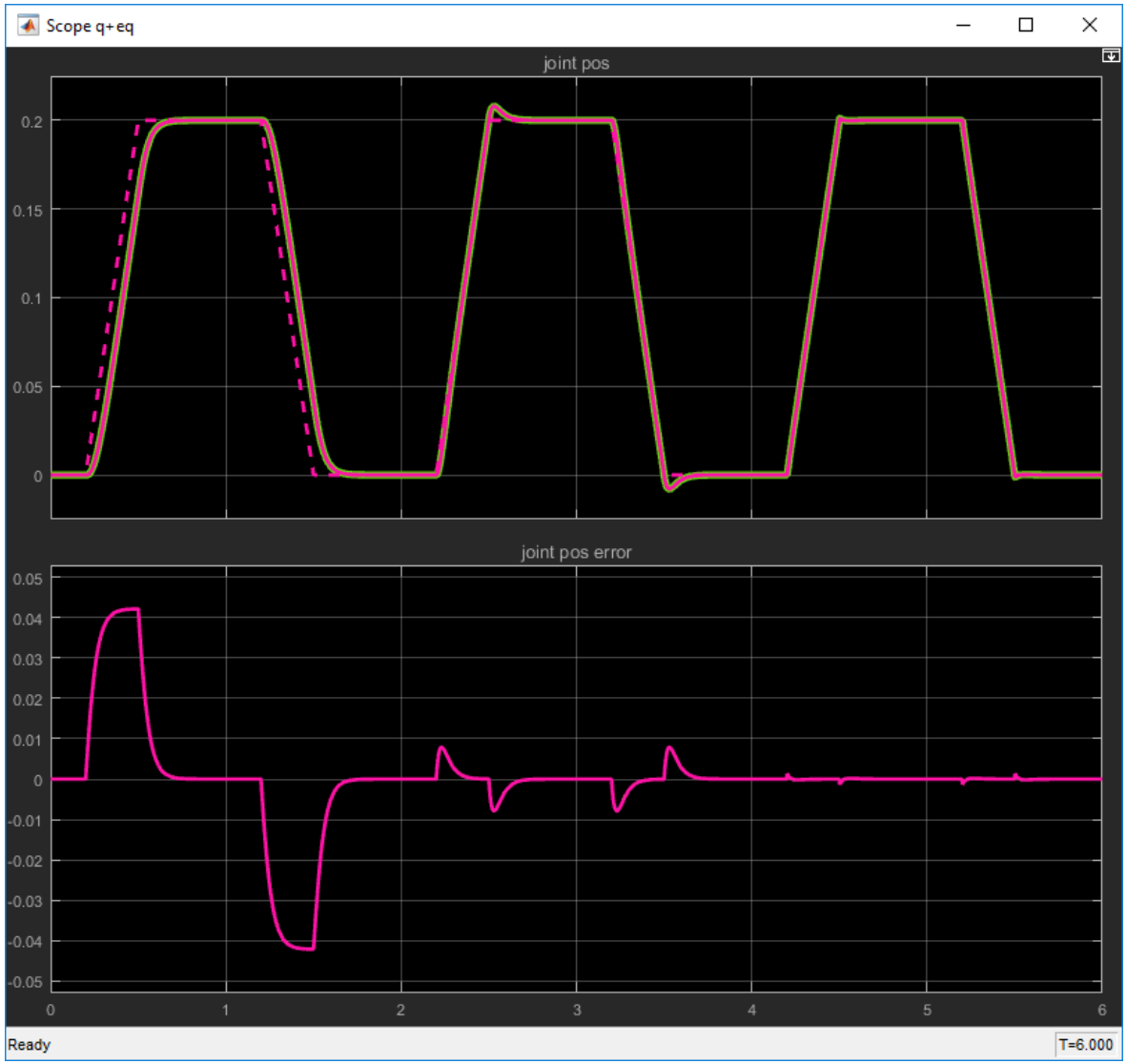
Linear motion between two joint configurations.

Gains K_p and K_d for the desired close-loop behavior:

$$\ddot{\boldsymbol{e}}_q + 2\sqrt{1000}\dot{\boldsymbol{e}}_q + 1000\boldsymbol{e}_q = 0$$

After $t > 2s$: $\dot{\boldsymbol{q}}$ dependent compensator is active.

After $t > 4s$: $\ddot{\boldsymbol{q}}$ dependent compensator is added.



Task space control

Most tasks are defined as the end-effector motion, i.e. a motion the the task space.

Using the kinematic transformation we could transform the task form task space to joint space and design the control in joint space. When accurate motion is required or different behavior in different task space directions ir required, it is necessary to design the control in the task space.

Relation between the joint space velocities and the task space velocities

$$\dot{\boldsymbol{x}} = \mathbf{J}\dot{\boldsymbol{q}}$$

If the inverse relation exists

$$\dot{\boldsymbol{q}} = \mathbf{J}^{\dagger}\dot{\boldsymbol{x}}$$

then the dynamics of the robot system in the task space is given in the form

$$\boldsymbol{F} = \boldsymbol{\Lambda}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{\Gamma}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \boldsymbol{\xi}(\boldsymbol{x}) - \boldsymbol{F}_{ext}$$

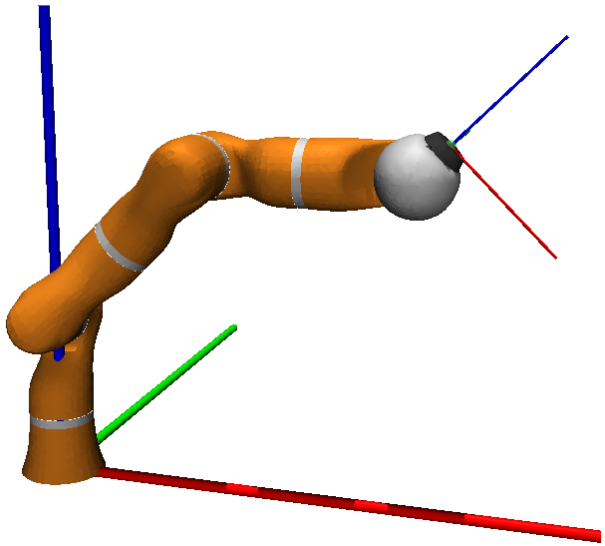
Task space control: Example KUKA LWR

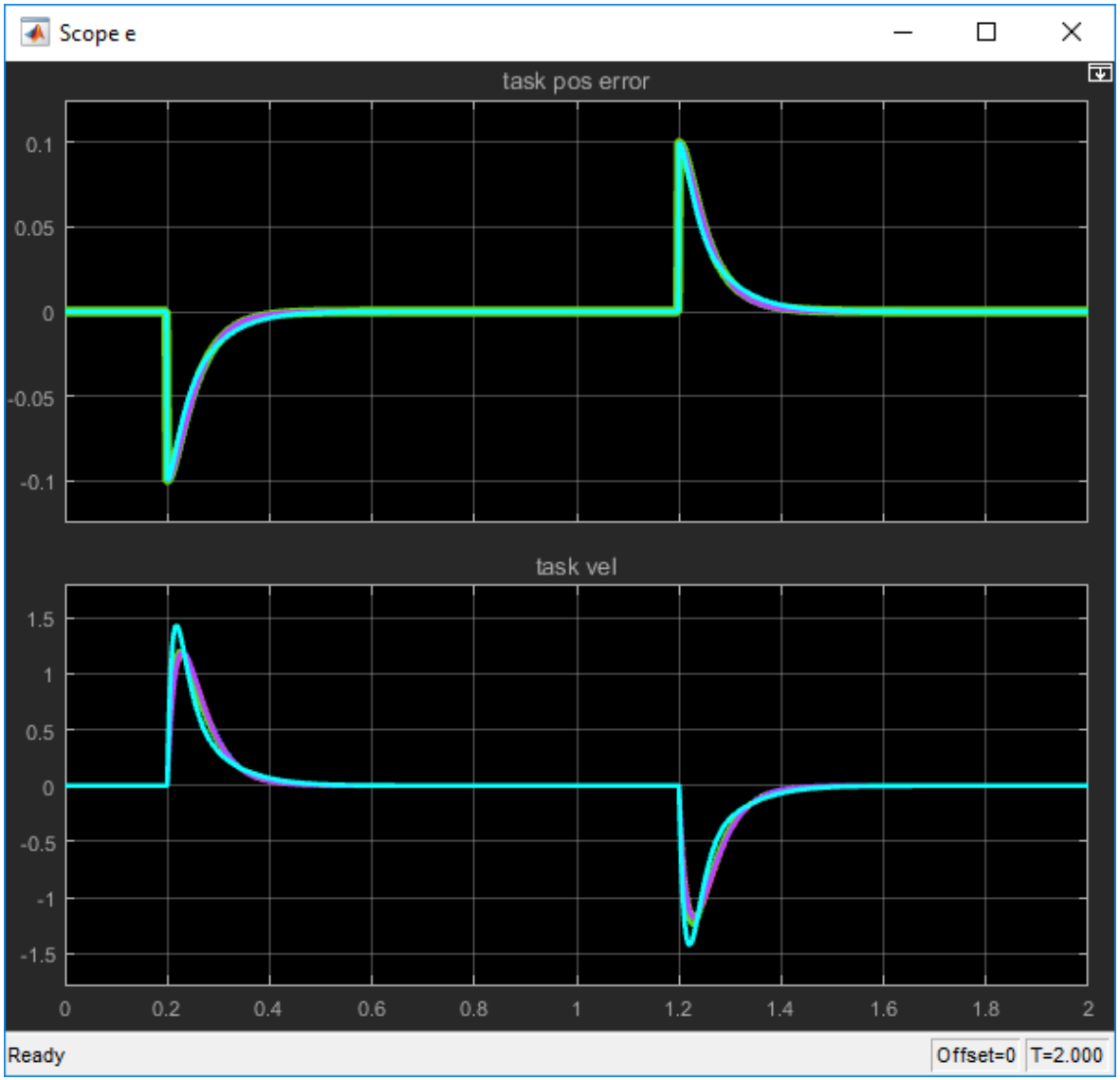
PTP motion in task-space

$$\boldsymbol{x} = [x, y, z, \alpha, \beta, \gamma]^T$$

$$\Delta \boldsymbol{x}_i = 0.2, \; i = 1, \ldots, 6$$

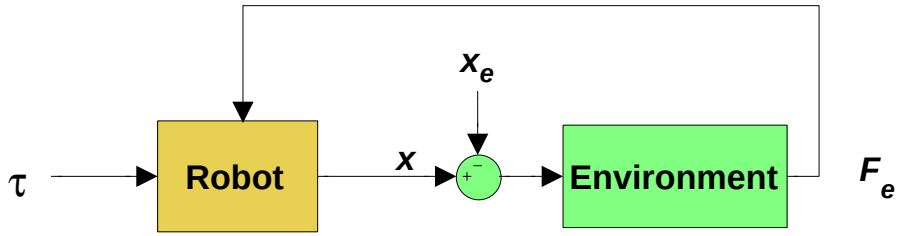
Same K_p and K_d gains for task-space DOFs.





Robot in contact with environment

When the robot is in contact with the environment, its motion is constrained and due to the contacts forces between the robot and the environment arise.



$$\boldsymbol{F}_e = \boldsymbol{G}(\boldsymbol{x} - \boldsymbol{x}_e)$$

Dynamic interaction depends on the contact properties:

- inertial contact (object pushing)
- damping (sliding on surface)
- elastic contact (pushing on soft object)

The best description of the interaction is given by the contact forces.

To control the contact, the system should have a corresponding compliance, which can passive or active.

Summary

Kinematics: geometrical relationships in terms of position/velocity between the joint- and work-space.

Dynamics: relationships between the torques applied to the joints and the consequent movements of the links.

Trajectory planning: planning of the desired motion of the manipulator.

Control: computation of the control actions (joint torques) necessary to execute a desired motion.