Introduction to Robotics

Lecture 10

Shuang Li, Jianwei Zhang
[sli, zhang]@informatik.uni-hamburg.de

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Technical Aspects of Multimodal Systems

June 18, 2021

Outline

Introduction

Spatial Description and Transformations

Forward Kinematics

Robot Description

Inverse Kinematics for Manipulators

Instantaneous Kinematics

Trajectory Generation 1

Trajectory Generation 2

Principles of Walking

Path Planning

Task/Manipulation Planning

Dynamics

Robot Control

- Introduction

Outline (cont.)

Dynamics – Recapitulation

- **Forward dynamics:**
  - *Input*: joint forces / torques;
  - *Output*: kinematics;
  - *Application*: Simulation of a robot model.
- **Inverse Dynamics**:
  - *Input*: desired trajectory of a manipulator;
  - *Output*: required joint forces / torques;
  - *Application*: model-based control of a robot.

$$\tau(t) \rightarrow \text{direct dynamics} \rightarrow \mathbf{q}(t), (\dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t))$$

$$\mathbf{q}(t) \rightarrow \text{inverse dynamics} \rightarrow \tau(t)$$

Dynamics – Recapitulation

General inverse dynamic equations of a manipulator:

$$\tau = M(\Theta)\ddot{\Theta} + V(\Theta, \dot{\Theta}) + G(\Theta)$$

Forward dynamic equations of a manipulator:

$$\begin{aligned}
\ddot{\Theta} &= M^{-1}(\Theta)(\tau - V(\Theta, \dot{\Theta}) - G(\Theta)) \\
\dot{\Theta} &= \int \ddot{\Theta} dt \\
\Theta &= \int \dot{\Theta} dt
\end{aligned}$$

Unlike kinematics, the inverse dynamics is easier to solve than forward dynamics.

Dynamics – Recapitulation

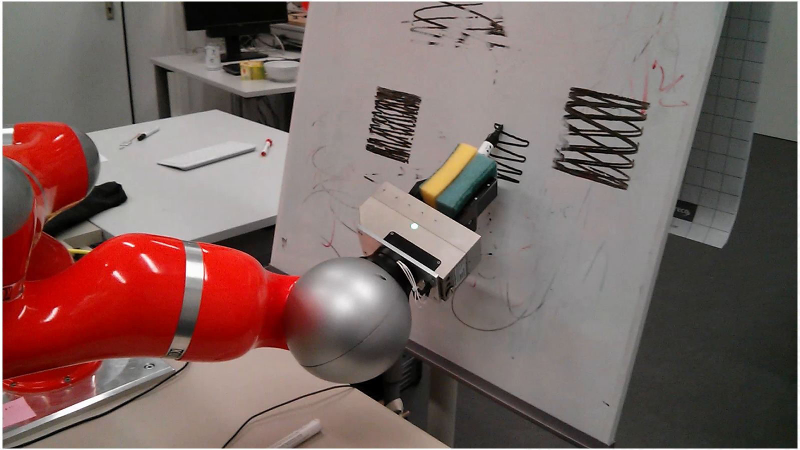
Two methods for calculation:

- Analytical methods
  - based on Lagrangian equations

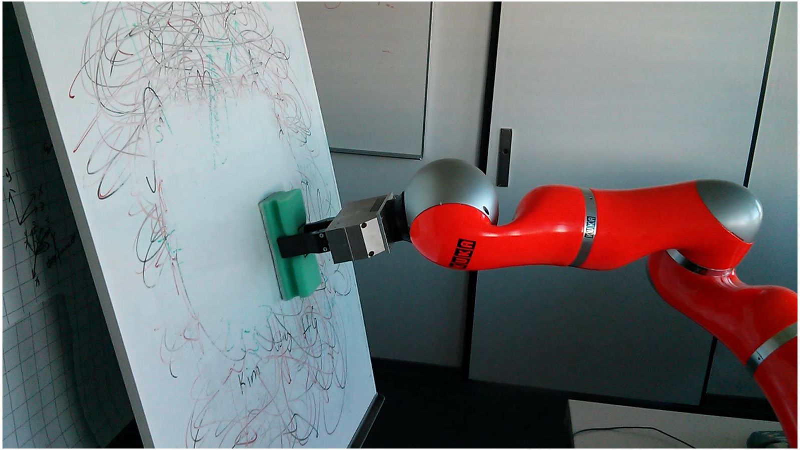
$$L(q_i, \dot{q}_i) = K(q_i, \dot{q}_i) - P(q_i)$$

- Synthetic methods:
  - based on the Newton-Euler equations

Drawing task



Wiping task



Definition

Controller

- Influences one or more physical variables
  - meet a control variable
  - reduce disturbances
- Compares actual value to reference value
  - minimize control deviation

Development of Control Engineering - Timeline

1788	J. Watt: engine speed governor
1877	J. Routh: differential equation for the description of control processes
1885	A. Hurwitz: stability studies
1932	A. Nyquist: frequency response analysis
1940	W. Oppelt: frequency response analysis, Control Engineering becomes an independent discipline
1945	H. Bode: discipline new methods for frequency response analysis
1950	N. Wiener: statistical methods
1956	L. Pontrjagin: optimal control theory, maximum principle
1957	R. Bellmann: dynamic programming
1960	direct digital control
1965	L. Zadeh: Fuzzy-Logic
1972	Microcomputer use
1975	Control systems for automation
1980	Digital device technology
1985	Fuzzy-controller for industrial use
1995	Artificial neuronal networks for industrial use

Control Problem

Given: dynamic system (to be controlled)

- Model describing dynamic system (e.g. Jacobian)
- Input variables – control variables
  - measured values (sensor data)
- Output variables – controlled variables
  - system input (force/torque data)

Problem

- Keep control variable values constant and / or
- Follow a reference value and / or
- Minimize the influence of disturbances

Control Problem (cont.)

Sought: controller (for dynamic system)

- Implement hardware or software controller
- Alter controlled-variables (output)
- Based on control variables (input)
- Solve the problem

Example: Cruise Control

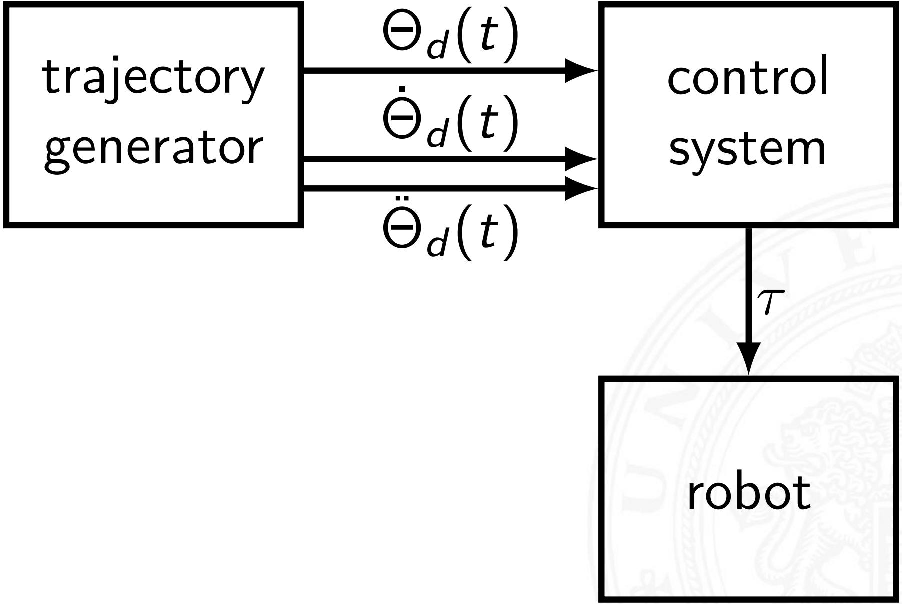
Input

- Speed over ground
- Relative speed to traffic
- Distance to car in front
- Distance to car behind
- Weather conditions
- Relative position in road lane
- ...

Output

- Throttle
- Brakes
- Steering

Control System of a Robot



trajectory generator

$\Theta_d(t)$

$\dot{\Theta}_d(t)$

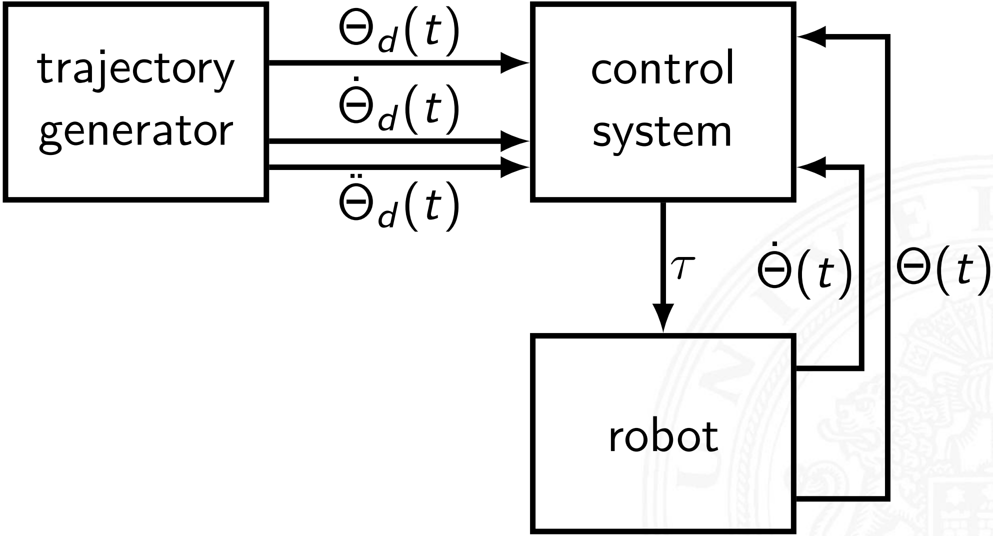
$\ddot{\Theta}_d(t)$

control system

τ

robot

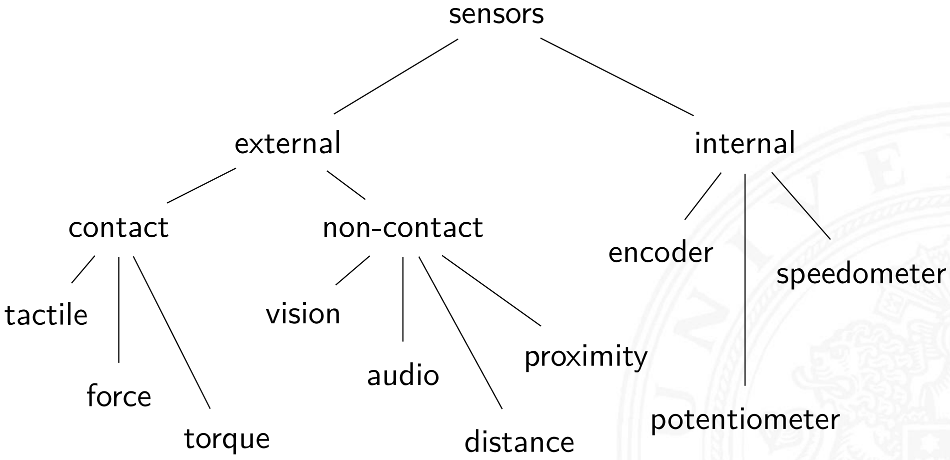
Control System of a Robot (cont.)



Control System of a Robot (cont.)

- Target values
    - $\Theta_d(t)$
    - $\dot{\Theta}_d(t)$
    - $\ddot{\Theta}_d(t)$
- Magnitude of error
    - $E = \Theta_d - \Theta, \dot{E} = \dot{\Theta}_d - \dot{\Theta}$
- Output (Control) value
    - $\Theta(t)$
    - $\dot{\Theta}(t)$
- Controlled value
    - τ

Sensor Classification Hierarchy



sensors

external

internal

contact

non-contact

tactile

force

torque

vision

audio

distance

proximity

encoder

potentiometer

speedometer

Internal Sensors of Robots

- Placed inside the robot
- Monitor the internal state of the robot
  - e.g. position and velocity of a joint

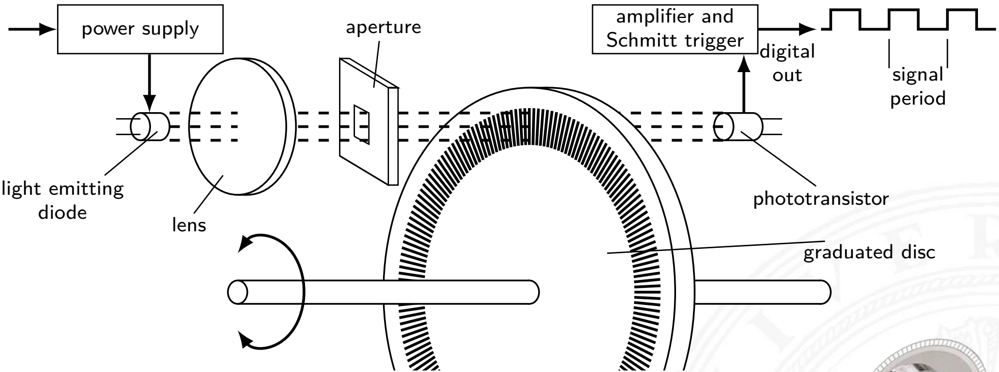
Position measurement systems

- Potentiometer
- Incremental/absolute encoder
- Resolver

Velocity measurement systems

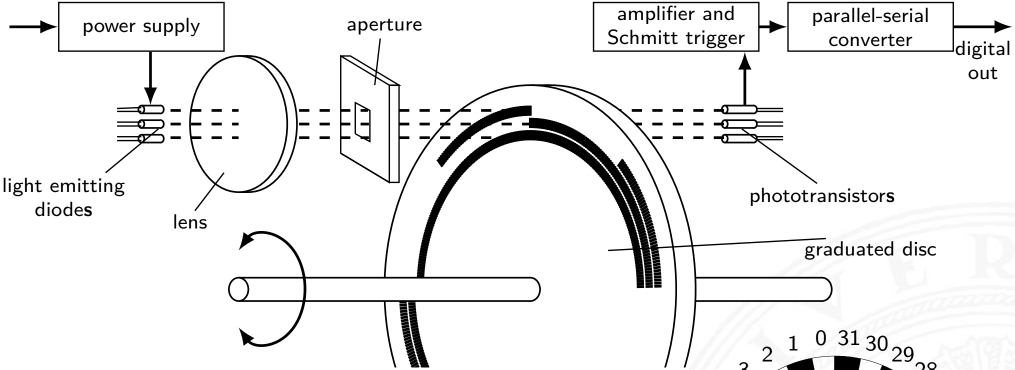
- Speedometers
- Calculate from position change over time

Optical Incremental Encoders

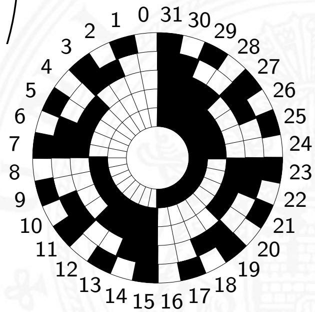


- An optical encoder reads the lines
- The disc is mounted to the shaft of the joint motor
  - PUMA-560: 1:1 ratio; .0001 rad/bit accuracy
- one special line is marked as the “zero-position”

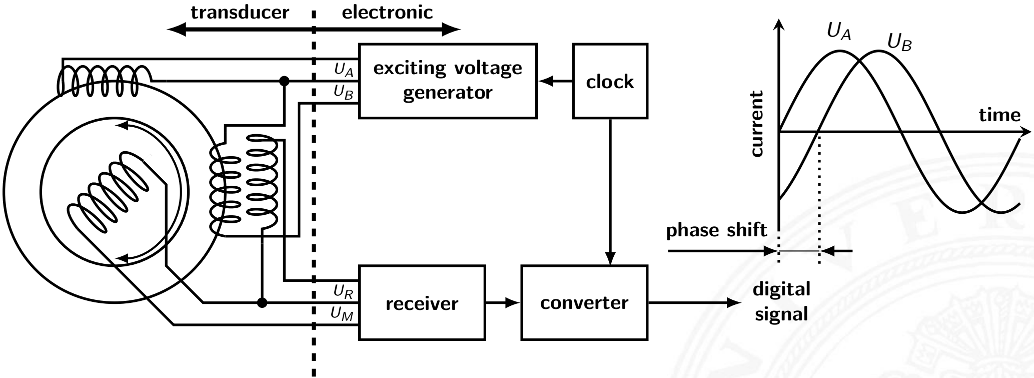
Optical Absolute Encoder



- multiple LEDs and phototransistors
- e.g. 5 bit dual code gives 32 angular positions and 11.25° resolution
- parallel-serial converter required
- absolute positioning and direction encoding



Resolver

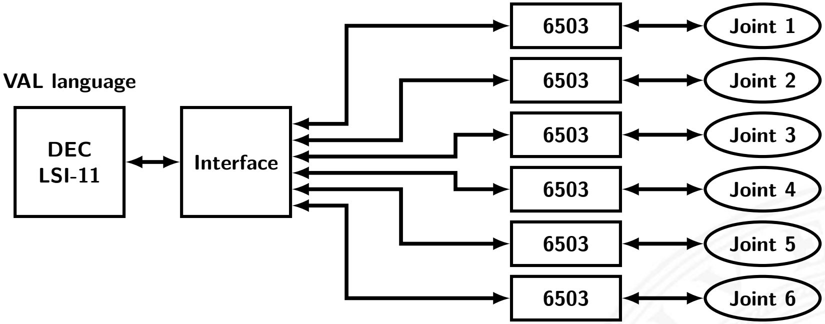


- analog rotation encoding
- phase shift between U_A and U_B determines rotation
- precision depending on digital converter

Encoder vs Resolver

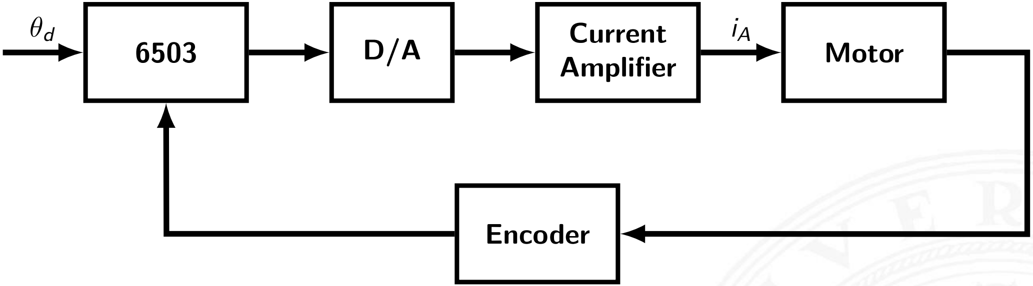
- Encoder:
  - higher accuracy
  - simplicity of integration, and update
  - suitable for applications with high acceleration and deceleration rates
- Resolver:
  - lack of sensitive optics
  - resistant to electrical disturbances
  - complexity of integrating a resolver into a system
  - suitable for extremely harsh applications, such as military and aerospace equipment

Control System Architecture of PUMA-Robot



- two-level hierachical structure of control system
- *DEC LSI-11* sends joint values at 35.7 Hz (28 ms)
    - trajectory
- Distance of actual value to goal value is interpolated
    - using 8,16,**32** or 64 increments

Control System Architecture of PUMA-Robot (cont.)



- The joint control loop operates at 1143 Hz (0.875 ms)
- Encoders are used as position sensors
- No dedicated speedometer
  - velocity is calculated as the difference of joint positions over time

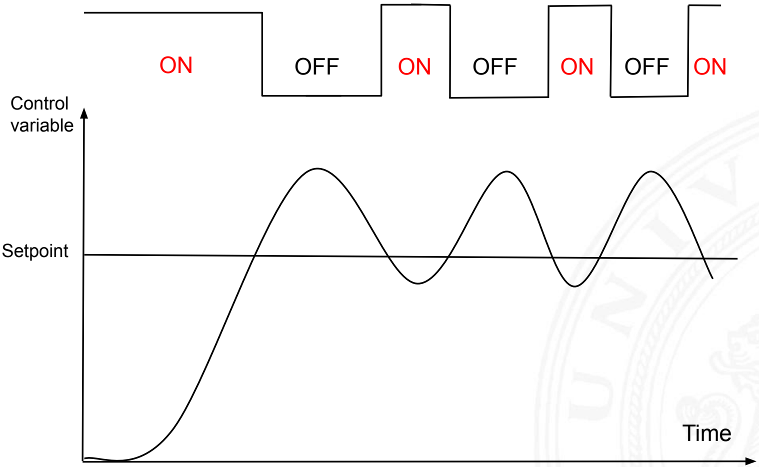
Introduction—PID controller

- more than half of the industrial controllers in use today are PID controllers or modified PID controllers
- many different types of tuning rules have been proposed in the literature
- Using these tuning rules, delicate and fine tuning of PID controllers can be made on-site

P	Proportional controller
I	Integral controller
D	Derivative controller

Bang Bang (On-off) controller

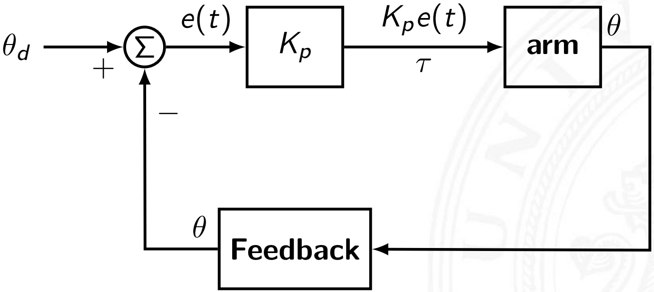
This is the simplest form of control.



Proportional control

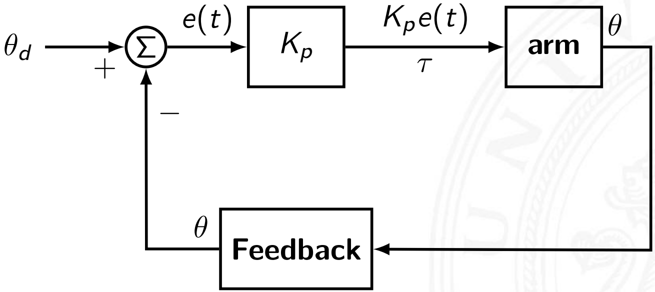
In proportional mode, there is a continuous linear relation between value of the controlled variable and position of the final control element.

- $e(t) = \theta_d - \theta$
- output of proportional controller is $\tau(t) = K_p e(t)$, K_p is proportional gain.



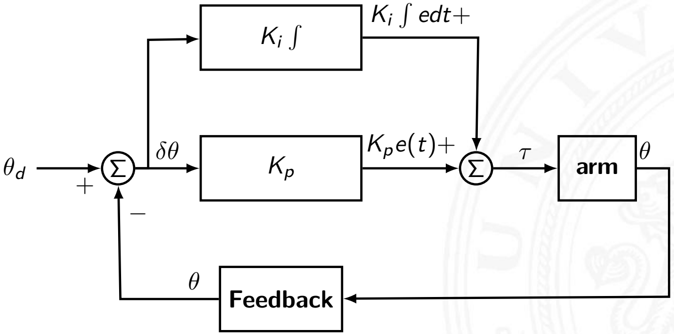
Proportional control (cont.)

- Using P control is simple, but often insufficient:
  - If K_p is small, the sensor reading will approach the setpoint slowly and never reach it
  - As the gain is increased the system responds faster to changes in set-point but becomes progressively underdamped and eventually unstable.
  - If K_p is large, the system may overshoot, oscillate (i.e. become unstable)



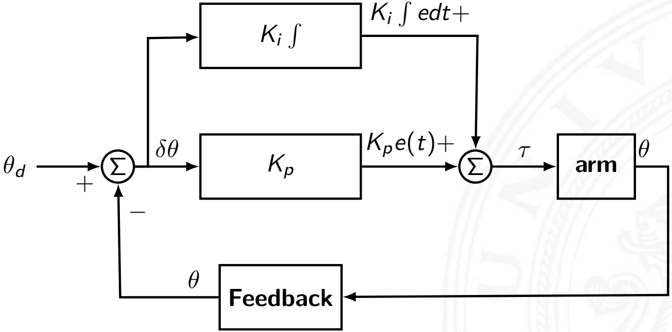
Proportional-Integral (PI) control

- $\tau(t) = K_p e(t) + K_i \int edt$
- The P term will take care of the large movement
- Integral signal is sum of all instantaneous errors
- The I term will take care of any steady-state error



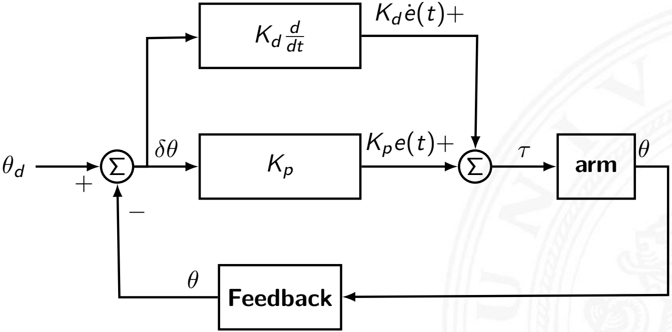
Proportional-Integral (PI) control (cont.)

- It eliminates steady-state error
- It can help with stability of the system, especially if K_p is large
- But, it responds relatively slowly to an error signal



Proportional-Derivative (PD) control

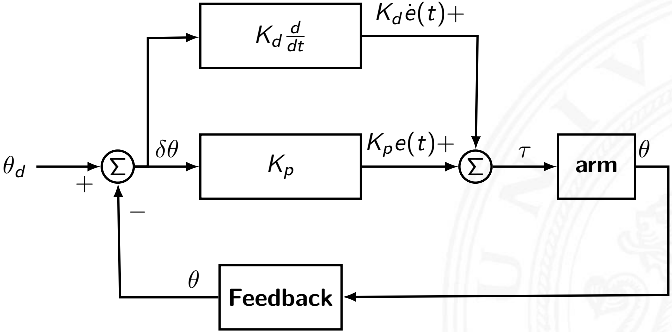
- $\tau(t) = K_p e(t) + K_d \dot{e}(t)$
- Differential term at time $n = K_d(e(n) - e(n-1)/\Delta t)$



Proportional-Derivative (PD) control (cont.)

The main advantages of the PD controllers are:

- The derivative term acts as "brake" to the system
- It can improve the system's tolerance to external disturbances



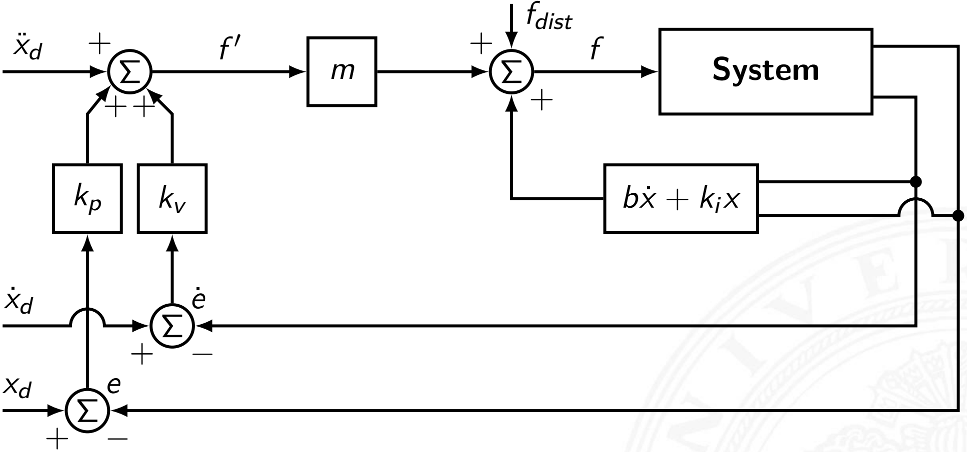
PID-Controller

- P Proportional controller: $\tau(t) = k_p \cdot e(t)$
  The amplification factor k_p defines the sensitivity.
- I Integral controller: $\tau(t) = k_i \cdot \int_{t_0}^{t} e(t')dt'$
  Long term errors will sum up.
- D Derivative controller: $\tau(t) = k_v \cdot \dot{e}(t)$
  This controller is sensitive to changes in the deviation.

Combined ⇒ PID-controller:

$$\tau(t) = k_p \cdot e(t) + k_v \cdot \dot{e}(t) + k_i \int_{t_0}^{t} e(t')dt'$$

Linear Control for Trajectory Tracking



$$f' = \ddot{x}_d + k_v\dot{e} + k_p e + k_i \int edt \qquad (47)$$

is called the principle of PID-control.

Summary: the characteristics of P, I, D controllers

	Rise time	Overshoot	Settling time	S-S error
Kp	decrease	increase	small change	decrease
Ki	decrease	increase	increase	eliminate
Kd	small change	decrease	decrease	small change

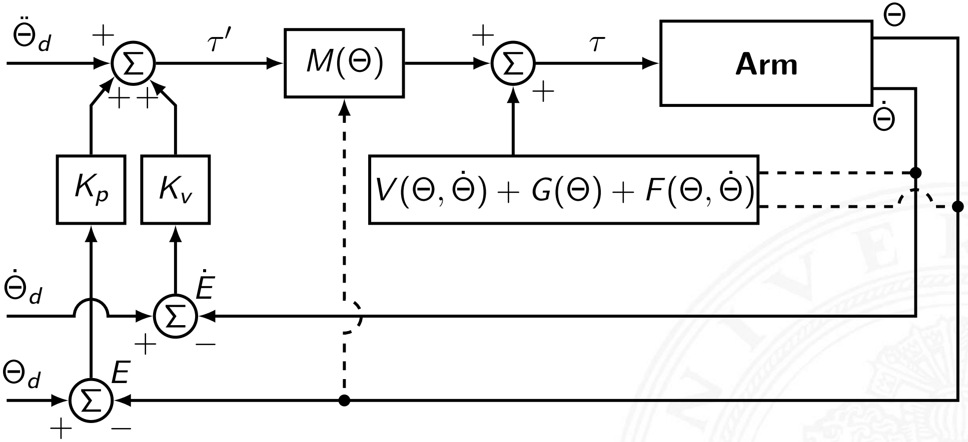
Further Resources

- PID Control with Python (simple-pid)
- PID Control with MATLAB and Simulink

Tips for designing a PID controller

1. Obtain an open-loop response and deternine what needs to be improved
2. Add a P control to improve the rise time
3. Add a D control to improve the overshoot
4. Add a I control to eliminate the steady-state error
5. Adjust each of K_p, K_d, K_i until you obtain a desrired overall response

Model-Based Control for Trajectory Tracking



The dynamic equation: $\tau = M(\Theta)\ddot{\Theta} + V(\Theta,\dot{\Theta}) + G(\Theta)$

where $M(\Theta)$ is the position-dependent $n \times n$-mass matrix of the manipulator, $V(\Theta,\dot{\Theta})$ is a $n \times 1$-vector of centripetal and Coriolis factors, and $G(\Theta)$ is a complex function of Θ, the position of all joints of the manipulator.

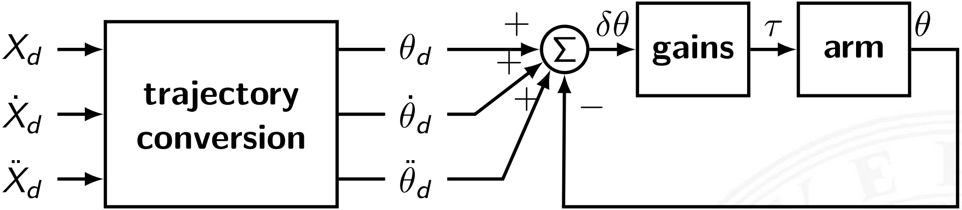
Classification of Robot Arm Controllers

As the problem of trajectory-tracking:

- Joint space: PID, plus model-based
- Cartesian space: joint-based
  - using kinematics or using inverse Jacobian calculation
- Adaptive: model-based adaptive control, self-tuning
  - controller (structure and parameter) adapts to the time-invariant or unknown system-behavior
  - basic control circle is superimposed by an adaptive system
  - process of adaption consists of three phases
    - identification
    - decision-process
    - modification
- Hybrid force and position control is also a popular research topic

Control in Cartesian Space – Method I

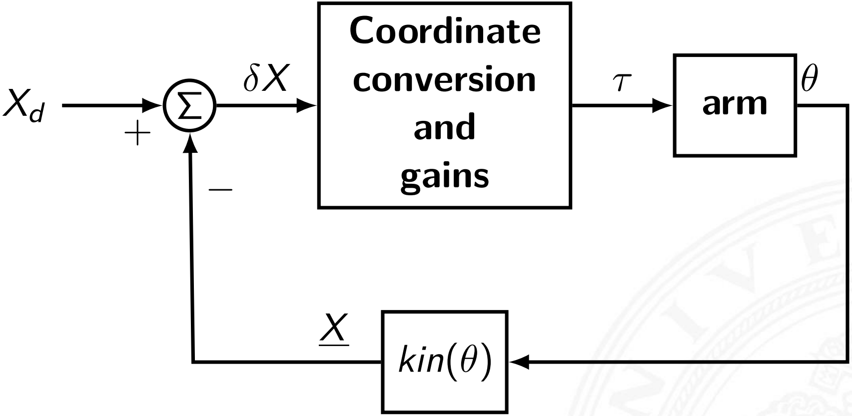
Joint-based control with Cartesian trajectory input



- Cartesian trajectory is converted into joint space first
- joint space trajectory is sent to the controller
- trajectory controller sends joint targets to motor controllers
- motor controller sends torque data to motor
- sensors output joint state

Control in Cartesian Space – Method II

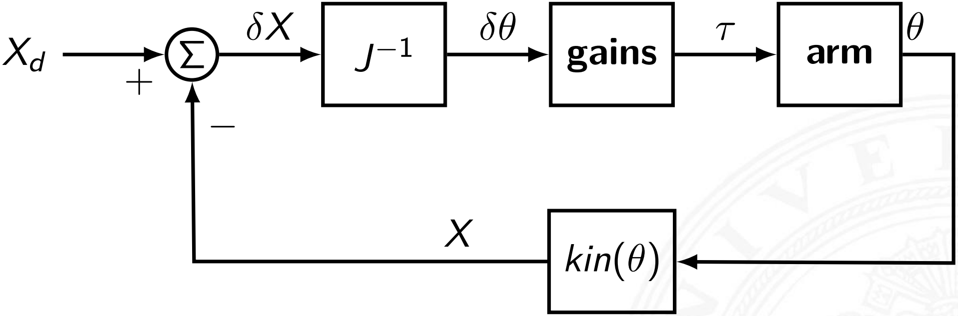
Cartesian control via calculation of kinematics



- controller operates in cartesian space
- joint space conversion within control cycle
- error values in cartesian space using FK

Control in Cartesian Space – Method III

Cartesian control via calculation of inverse Jacobian



- no explicit joint space conversion
- dynamic conversion using inverse Jacobian

Robot Control Improvements

Scientific Research

- model-based control
- adaptive control
- hybrid control

Industrial robotcs

- PID-control system with gravity compensation

$$\tau = \dot{\Theta}_d + K_v \dot{E} + K_p E + K_i \int E dt + \hat{G}(\Theta)$$

Hybrid Control of Force and Position

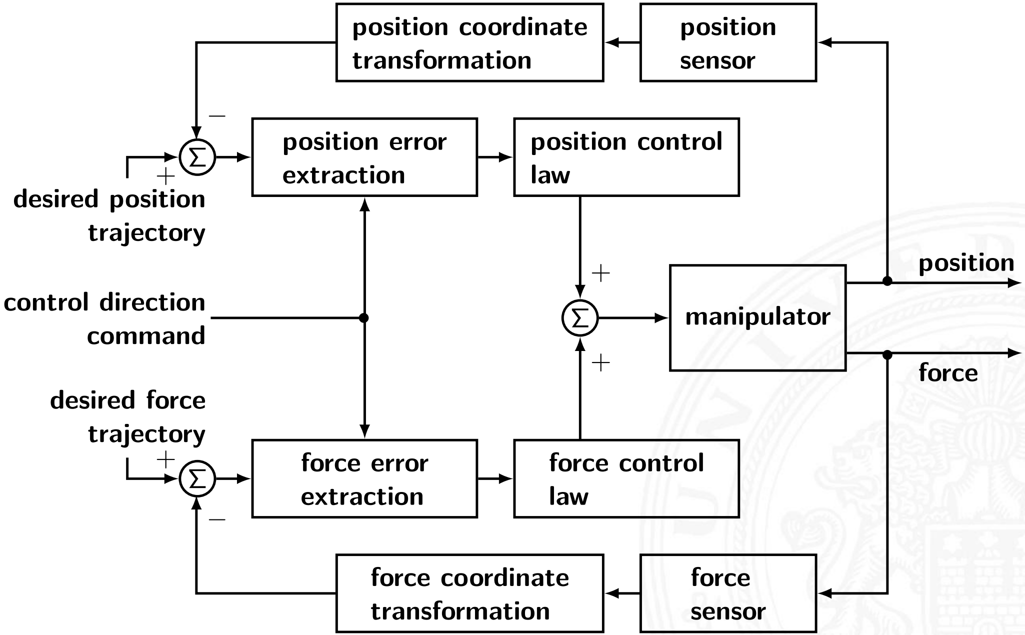
Motivation

Certain tasks require control of both: position and force of the end effector:

- assembly
- grinding
- opening/closing doors
- crank winding
- ...

An example shows two feedback loops for separate control of position and force

Hybrid Control of Force and Position (cont.)



Hybrid Force/Torque Control

Franke Emika Panda

Bibliography

[1] G.-Z. Yang, R. J. Full, N. Jacobstein, P. Fischer, J. Bellingham, H. Choset, H. Christensen, P. Dario, B. J. Nelson, and R. Taylor, “Ten robotics technologies of the year,” 2019.

[2] J. K. Yim, E. K. Wang, and R. S. Fearing, “Drift-free roll and pitch estimation for high-acceleration hopping,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8986–8992, IEEE, 2019.

[3] J. F. Engelberger, *Robotics in service*.
MIT Press, 1989.

[4] K. Fu, R. González, and C. Lee, *Robotics: Control, Sensing, Vision, and Intelligence*.
McGraw-Hill series in CAD/CAM robotics and computer vision, McGraw-Hill, 1987.

[5] R. Paul, *Robot Manipulators: Mathematics, Programming, and Control: the Computer Control of Robot Manipulators*.
Artificial Intelligence Series, MIT Press, 1981.

[6] J. Craig, *Introduction to Robotics: Pearson New International Edition: Mechanics and Control*.
Always learning, Pearson Education, Limited, 2013.

Bibliography (cont.)

[7] T. Flash and N. Hogan, “The coordination of arm movements: an experimentally confirmed mathematical model,” *Journal of neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[8] T. Kröger and F. M. Wahl, “Online trajectory generation: Basic concepts for instantaneous reactions to unforeseen events,” *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 94–111, 2009.

[9] W. Böhm, G. Farin, and J. Kahmann, “A Survey of Curve and Surface Methods in CAGD,” *Comput. Aided Geom. Des.*, vol. 1, pp. 1–60, July 1984.

[10] J. Zhang and A. Knoll, “Constructing Fuzzy Controllers with B-spline Models - Principles and Applications,” *International Journal of Intelligent Systems*, vol. 13, no. 2-3, pp. 257–285, 1998.

[11] M. Eck and H. Hoppe, “Automatic Reconstruction of B-spline Surfaces of Arbitrary Topological Type,” in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’96, (New York, NY, USA), pp. 325–334, ACM, 1996.

Bibliography (cont.)

[12] A. Cowley, W. Marshall, B. Cohen, and C. J. Taylor, “Depth space collision detection for motion planning,” 2013.

[13] Hornung, Armin and Wurm, Kai M. and Bennewitz, Maren and Stachniss, Cyrill and Burgard, Wolfram, “OctoMap: an efficient probabilistic 3D mapping framework based on octrees,” *Autonomous Robots*, vol. 34, pp. 189–206, 2013.

[14] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, “Manipulation planning on constraint manifolds,” in *2009 IEEE International Conference on Robotics and Automation*, pp. 625–632, 2009.

[15] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[16] O. Khatib, “The Potential Field Approach and Operational Space Formulation in Robot Control,” in *Adaptive and Learning Systems*, pp. 367–377, Springer, 1986.

[17] L. E. Kavraki, P. Svestka, J. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

Bibliography (cont.)

[18] J. Kuffner and S. LaValle, “RRT-Connect: An Efficient Approach to Single-Query Path Planning.,” vol. 2, pp. 995–1001, 01 2000.

[19] J. Starek, J. Gómez, E. Schmerling, L. Janson, L. Moreno, and M. Pavone, “An asymptotically-optimal sampling-based algorithm for bi-directional motion planning,” *Proceedings of the ... IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2015, 07 2015.

[20] D. Hsu, J. . Latombe, and R. Motwani, “Path planning in expansive configuration spaces,” in *Proceedings of International Conference on Robotics and Automation*, vol. 3, pp. 2719–2726 vol.3, 1997.

[21] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, “Motion planning networks,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 2118–2124, IEEE, 2019.

[22] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, “Finding locally optimal, collision-free trajectories with sequential convex optimization,” in *in Proc. Robotics: Science and Systems*, 2013.

Bibliography (cont.)

[23] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[24] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017.

[25] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *2011 IEEE International Conference on Robotics and Automation*, pp. 1470–1477, 2011.

[26] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki, "Incremental task and motion planning: A constraint-based approach.," in *Robotics: Science and Systems*, pp. 1–6, 2016.

[27] J. Ferrer-Mestres, G. Francès, and H. Geffner, "Combined task and motion planning as classical ai planning," *arXiv preprint arXiv:1706.06927*, 2017.

[28] M. Görner, R. Haschke, H. Ritter, and J. Zhang, "MoveIt! Task Constructor for Task-Level Motion Planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

Bibliography (cont.)

[29] K. Hauser and J.-C. Latombe, “Multi-modal motion planning in non-expansive spaces,” *The International Journal of Robotics Research*, vol. 29, no. 7, pp. 897–915, 2010.

[30] B. Siciliano and O. Khatib, *Springer handbook of robotics*.
Springer, 2016.

[31] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain, “Time-contrastive networks: Self-supervised learning from video,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1134–1141, IEEE, 2018.

[32] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” *arXiv preprint arXiv:1703.03400*, 2017.

[33] R. Brooks, “A robust layered control system for a mobile robot,” *Robotics and Automation, IEEE Journal of*, vol. 2, pp. 14–23, Mar 1986.

[34] M. J. Mataric, “Interaction and intelligent behavior.,” tech. rep., DTIC Document, 1994.

Bibliography (cont.)

[35] M. P. Georgeff and A. L. Lansky, “Reactive reasoning and planning.,” in *AAAI*, vol. 87, pp. 677–682, 1987.

[36] J. S. Albus, “The nist real-time control system (rcs): an approach to intelligent systems research,” *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 157–174, 1997.

[37] T. Fukuda and T. Shibata, “Hierarchical intelligent control for robotic motion by using fuzzy, artificial intelligence, and neural network,” in *Neural Networks, 1992. IJCNN., International Joint Conference on*, vol. 1, pp. 269–274 vol.1, Jun 1992.

[38] L. Einig, *Hierarchical Plan Generation and Selection for Shortest Plans based on Experienced Execution Duration*.
Master thesis, Universität Hamburg, 2015.

[39] J. Craig, *Introduction to Robotics: Mechanics & Control. Solutions Manual*.
Addison-Wesley Pub. Co., 1986.

Bibliography (cont.)

[40] H. Siegert and S. Bocionek, *Robotik: Programmierung intelligenter Roboter: Programmierung intelligenter Roboter*.
Springer-Lehrbuch, Springer Berlin Heidelberg, 2013.

[41] R. Schilling, *Fundamentals of robotics: analysis and control*.
Prentice Hall, 1990.

[42] T. Yoshikawa, *Foundations of Robotics: Analysis and Control*.
Cambridge, MA, USA: MIT Press, 1990.

[43] M. Spong, *Robot Dynamics And Control*.
Wiley India Pvt. Limited, 2008.