MEE5114 Advanced Control for Robotics

Lecture 12: Robot Motion Control

Prof. Wei Zhang

CLEAR Lab
Department of Mechanical and Energy Engineering
Southern University of Science and Technology, Shenzhen, China
https://www.wzhanglab.site/

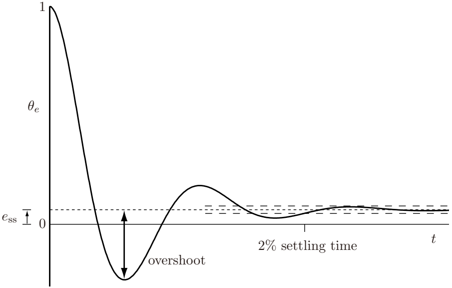
Outline

- Basic Linear Control Design
- Motion Control Problems
- Motion Control with Velocity/Acceleration as Input
- Motion Control with Torque as Input and Task Space Inverse Dynamics

Outline

- Basic Linear Control Design
- Motion Control Problems
- Motion Control with Velocity/Acceleration as Input
- Motion Control with Torque as Input and Task Space Inverse Dynamics

Error Response



- Steady-state error:

- Percent overshoot:

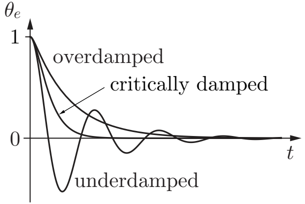
- Rise time/Peak time:

- Settling time:

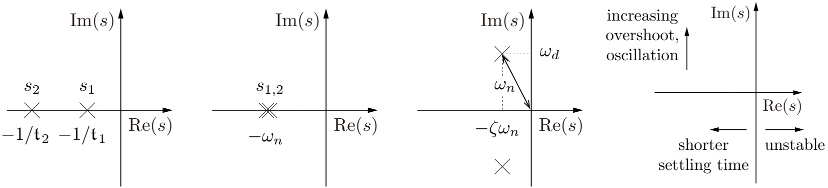
Standard Second-Order Systems

$$\ddot{\theta}_e(t) + 2\xi\omega_n\dot{\theta}_e(t) + \omega_n^2\theta_e(t) = 0 \quad \leftrightarrow \quad s^2 + 2\xi\omega_n s + \omega_n^2 = 0$$

- ξ: damping ration, ω_n: natural frequency
- Underdamped:
- Critically damped:
- Overdamped:



Second-Order Response Characteristics



- Settling time:

- Peak time:

- Percent overshoot:

State-Space Controller Design (1/2)

- Linear Control Systems: $\dot{x} = Ax + Bu$, $y = Cx + Du$
- Linear Control Law: $u = -Kx$
- Closed-loop Dynamics:
- Solution of CL-Dynamics:
- Closed-loop Stability condition:

State-Space Controller Design (2/2)

- Eigenvalue assignment:

- Solvability:

- How to choose desired eigs?:

- LQR

Robot Motion Control Problems (1/1)

- Dynamic equation of fully-actuated robot (without external force):

$$\begin{cases} \tau &= M(\theta)\ddot{q} + c(q,\dot{q})\dot{q} + g(q) \\ y &= h(q) \end{cases} \tag{1}$$

- $q \in \mathbb{R}^n$: joint positions (generalized coordinate)

- $\tau \in \mathbb{R}^n$: joint torque (generalized input)

- y: output (variable to be controlled)

- **Motion Control Problems**: Let y track given reference y_d

Variations in Robot Motion Control

- Joint-space vs. Task-space control:
  - Joint-space: $y(t) = q(t)$, i.e., want $q(t)$ to track a given $q_d(t)$ joint reference
  - Task-space: $y(t) = T(q(t))$ denotes end-effector pose/configuration, we want $y(t)$ to track $y_d(t)$
- Actuation models:
  - Velocity source: $u = \dot{q}$
  - Acceleration sources: $u = \ddot{q}$
  - Torque sources: $u = \tau$

Outline

- Basic Linear Control Design
- Motion Control Problems
- Motion Control with Velocity/Acceleration as Input
- Motion Control with Torque as Input and Task Space Inverse Dynamics

Velocity-Resolved Control

- Each joints' velocity $\dot{q}_i$ can be directly controlled

- Good approximation for hydraulic actuators

- Common approximation of the outer-loop control for the Inner/outer loop control setup

Velocity-Resolved Joint Space Control

- Joint-space "dynamics": single integrator

$$\dot{q} = u$$

- Joint-space tracking becomes standard linear tracking control problem:

$$u = \dot{q}_d + K_0 \tilde{q} \quad \Rightarrow \dot{\tilde{q}} + K_0 \tilde{q} = 0$$

where $\tilde{q} = q_d - q$ is the joint position error.

- The error dynamic is stable if $-K_0$ is Hurwitz

Velocity-Resolved Task-Space Control (1/3)

- For task space control, $y = T(q)$ needs to track y_d
  - y can be any function of q, in particular, it can represents position and/or the end-effector frame
- Taking derivatives of y, and letting $u = \dot{q}$, we have

$$\dot{y} = J_a(q)u \tag{2}$$

  - Note that q is function of y through inverse kinematics.
  - So the above dynamics can be written in terms of y and u only. The detailed form can be quite complex in general

Velocity-Resolved Task-Space Control (2/3)

- System (2) is nonlinear system, a common way is to break it into inner-outer loop, where the outer loop directly control velocity of y, and the inner loop tries to find u to generate desired task space velocity

- **Outer loop**: $\dot{y} = v_y$, where control $v_y = \dot{y}_d + K_0\tilde{y}$, resulting in task-space closed-loop error dynamics:

$$\dot{\tilde{y}} + K_0\tilde{y} = 0$$

- Above task space tracking relies on a fictitious control v_y, i.e., it assumes $\dot{y}$ can be arbitrarily controlled by selecting appropriate $u = \dot{q}$, which is true if J_a is full-row rank.

Velocity-Resolved Task-Space Control (3/3)

- **Inner loop**: Given v_y from the outer loop, find the joint velocity control by solving

$$\begin{cases} \min_u \|v_y - J_a(q)u\|^2 + \text{regularization term} \\ \text{subj. to:} \quad \text{Constraints on } u \end{cases} \tag{3}$$

- Inner-loop is essentially a differential IK controller

- One can also use the pseudo-inverse control $u = J_a^{\dagger} v_y$

Acceleration-Resolved Control in Joint Space

- Joint acceleration can be directly controlled, resulting in double-integrator dynamics

$$\ddot{q} = u$$

- Joint-space tracking becomes standard linear tracking control problem for double-integrator system:

$$u = \ddot{q}_d + K_1\dot{\tilde{q}} + K_0\tilde{q} \quad \Rightarrow \ddot{\tilde{q}} + K_1\dot{\tilde{q}} + K_0\tilde{q} = 0$$

  where $\tilde{q} = q_d - q$ is the joint position error.

- Stability condition:

Acceleration-Resolved Control in Task Space (1/2)

- For task space control, $y = T(q)$ needs to track y_d
- Note: $\dot{y} = J_a(q)\dot{q}$ and $\ddot{y} = \dot{J}_a(q)\dot{q} + J_a(q)\ddot{q}$
- Following the same inner-outer loop strategy discussed before
- **Outer-loop** dynamics: $\ddot{y} = a_y$, with a_y being the outer-loop control input

$$a_y = \ddot{y}_d + K_1\dot{\tilde{y}} + K_0\tilde{y} \quad \Rightarrow \quad \ddot{\tilde{y}} + K_1\dot{\tilde{y}} + K_0\tilde{y} = 0$$

Acceleration-Resolved Control in Task Space (2/2)

- **Inner-loop:** Given a_y from outer loop, find the "best" joint acceleration:

$$\begin{cases} \min_u \|a_y - \dot{J}_a(q)\dot{q} - J_a(q)u\|^2 + \text{regularization term} \\ \text{subj. to:} \quad \text{Constraints on } u \end{cases} \tag{4}$$

- Mathematically, the above problem is the same as the Differential IK problem

- At any given time, $q, \dot{q}$ can be measured, and then y and $\dot{y}$ can be computed, which allows us to compute outer loop control a_y and inter loop control u

Outline

- Basic Linear Control Design
- Motion Control Problems
- Motion Control with Velocity/Acceleration as Input
- Motion Control with Torque as Input and Task Space Inverse Dynamics

Recall Properties of Robot Dynamics

For fully actuated robot:

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) \tag{5}$$

- $M(q) \in \mathbb{R}^{n\times n} \succ 0$
- There are many valid definitions of $C(q,\dot{q})$, typical choice for C include:

$$C_{ij} = \sum_k \frac{1}{2}\left[\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i}\right]$$

- For the above defined C, we have $\dot{M} - 2C$ is skew symmetric
- For all valid C, we have $\dot{q}^T\left[\dot{M} - 2C\right]\dot{q} = 0$
- These properties play important role in designing motion controller

Computed Torque Control (1/2)

- For fully-actuated robot, we have $M(q) \succ 0$ and $\ddot{q}$ can be arbitrarily specified through torque control $u = \tau$

$$\ddot{q} = M^{-1}(q)\left[u - C(q,\dot{q})\dot{q} - g(q)\right]$$

- Thus, for fully-actuated robot, torque controlled case can be reduced to the acceleration-resolved case

- Outer loop: $\ddot{q} = a_q$ with joint acceleration as control input

$$a_q = \ddot{q}_d + K_1\dot{\tilde{q}} + K_0\tilde{q} \quad \Rightarrow \ddot{\tilde{q}} + K_1\dot{\tilde{q}} + K_0\tilde{q} = 0$$

- Inner loop: since $M(q)$ is square and nonsingular, inner loop control u can be found analytically:

$$u = M(q)\left(\ddot{q}_d + K_1\dot{\tilde{q}} + K_0\tilde{q}\right) + C(q,\dot{q})\dot{q} + g(q) \tag{6}$$

Computed Torque Control (2/2)

- The control law (6) is a function of $q, \dot{q}$ and the reference q_d. It is called *computed-torque control*.
- The control law also relies on system model M, C, g, if these model information are not accurate, the control will not perform well.
- Idea easily extends to task space: $\dot{y} = J_a(q)\dot{q}$ and $\ddot{y} = \dot{J}_a(q)\dot{q} + J_a(q)\ddot{q}$
- Outer loop: $\ddot{y} = a_y$, and $a_y = \ddot{y}_d + K_1\dot{\tilde{y}} + K_0\tilde{y}$
- Inner loop: select torque control $u = \tau$ by

$$\begin{cases} & \min_u \|a_y - \dot{J}_a\dot{q} - J_a M^{-1}(u - C\dot{q} - g)\|^2 \\ \text{subj. to:} & \text{constraints} \end{cases} \tag{7}$$

- If J_a is invertible and we don't impose additional torque constraints, analytical control law can be easily obtained.

Inverse Dynamics Control (1/2)

- The computed-torque controller in (6) is also called *inverse dynamics control*

- Forward dynamics: given τ to compute $\ddot{q}$

- Inverse dynamics: given desired acceleration a_q, we inverted it to find the required control by $u = Ma_q + C\dot{q} + g$

- Task space case can be viewed as inverting the task space dynamics

- With recent advances in optimization, it is often preferred to do ID with quadratic program

Inverse Dynamics Control (2/2)

- For example, Eq (7) can be viewed as task-space ID. We can incorporate torque contraints explicitly as follows:

$$\begin{cases} & \min_u \|a_y - \dot{J}_a \dot{q} - J_a M^{-1}(u - C\dot{q} - g)\|^2 \\ \text{subj. to:} & u_- \le u \le u_+ \end{cases} \tag{8}$$

- This is equivalent to the following more popular form:

$$\begin{cases} \min\limits_{u,\ddot{q}} & \|a_y - \dot{J}_a \dot{q} - J_a \ddot{q}\|^2 \\ \text{subj. to:} & M\ddot{q} + C\dot{q} + g = u \\ & u_- \le u \le u_+ \end{cases} \tag{9}$$

More Discussions

-

More Discussions

-

More Discussions

•