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/

机器人：具有类似或超越人的运动能力、计算能力、
感知决策能力 机电系统。

计 感 决 $\Longrightarrow$ 为运动服务

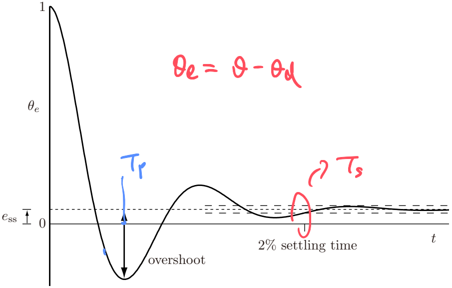
Outline

- 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: $e_{ss} \triangleq \lim_{t\to\infty} \theta_e(t)$
- Percent overshoot: P.O.

work for all systems.

- Rise time/Peak time:
- Settling time: T_s

Standard Second-Order Systems

$H(s) = \frac{\omega_n^2}{s^2 + \cdots}$

$$\underbrace{\ddot{\theta}_e(t) + 2\xi\omega_n\dot{\theta}_e(t) + \omega_n^2\theta_e(t)}_{2^{nd}\text{-ord ODE}} = 0 \quad \leftrightarrow \quad s^2 + 2\xi\omega_n s + \omega_n^2 = 0$$

$s_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$

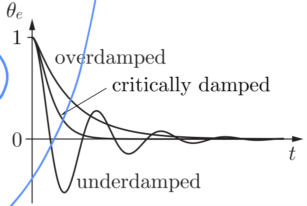
overdamped case

- ξ: damping ~~ration~~ ratio, ω_n: natural frequency
- Underdamped: $\xi < 1$, $\theta_e(t) = e^{-\xi\omega_n t}(c_1 \cos\omega_d t + c_2 \sin\omega_d t)$

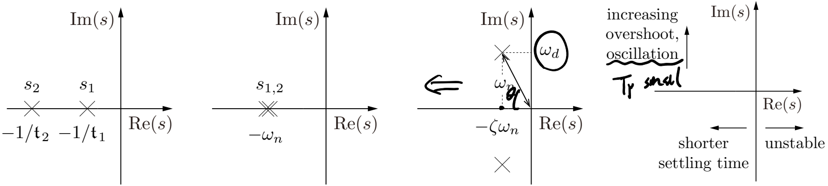
  $\omega_d = \omega_n\sqrt{1-\xi^2}$
- Critically damped:

  $\xi = 1$, repeated poles
- Overdamped:

  $\xi > 1$, $\theta_e(t) = c_1 e^{s_1 t} + c_2 e^{s_2 t}$



Second-Order Response Characteristics ($\zeta < 1$)



$$s_{1,2} = -\zeta\omega_n \pm j\,\omega_n\sqrt{1-\zeta^2}$$

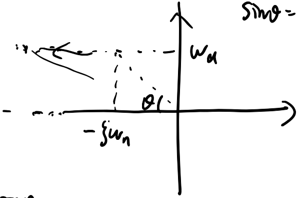
- Settling time: $T_s \approx \frac{4}{\zeta\omega_n}$ (faster T_s : Larger $\zeta\omega_n$

$\cos\theta = \zeta$

$\sin\theta = \sqrt{1-\zeta^2}$

- Peak time: $T_p \approx \frac{\pi}{\omega_n\sqrt{1-\zeta^2}}$ $\leftrightarrow$ ω_d

- Percent overshoot: $P.O. = 100\, e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}}$



$\cos\theta$

State-Space Controller Design (1/2)

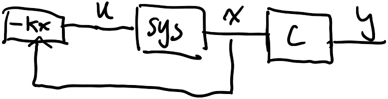
- Linear Control Systems: $\dot{x} = Ax + Bu$, $y = Cx + Du$

· Regulation problem : we want $x(t) \to 0$, $y(t) \to 0$

· Tracking problem : $\tilde{x}(t) = x_d(t) - x(t)$, we want $\tilde{x}(t) \to 0$

- Linear Control Law: $u = \underline{-Kx}$

regulation



- Closed-loop Dynamics:

$$\dot{x} = Ax + B(-Kx) = (A - BK)x$$

A_{CL}

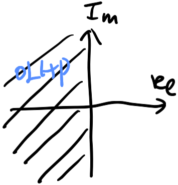
$$\dot{x} = A_{CL} \cdot x$$

- Solution of CL-Dynamics:

$$x(t) = e^{A_{CL} \cdot t} x(0)$$

- Closed-loop Stability condition: we want $\|x(t)\| \to 0$

we need $eig(A_{CL}) \in OLHP$



State-Space Controller Design (2/2)

- Eigenvalue assignment: Find control gain K such that

$$\underline{\text{eig}(A-BK) = \text{eig}_{\text{desired}}}$$

- Solvability: we can always find such K if (A, B) is controllable (rank(M_c) = n)

- How to choose desired eigs?: refer to 2nd-order system

specification "art" $\Longrightarrow$ dominant poles $\Longrightarrow$ eig desire $\Rightarrow$ K

$\begin{cases} \text{P.O} \\ T_p \\ T_s \end{cases}$ $\begin{cases} \text{dominant poles} \\ \text{+ other poles} \end{cases}$

science

- LQR

$\Downarrow$

Robot Motion Control Problems (1/1)

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

$$\begin{cases} \tau &= M(\theta)\ddot{q} + c(q,\dot{q})\dot{q} + g(q) + J^T(q) F_{ext} \\ y &= h(q) \leftarrow \text{output} \end{cases} \tag{1}$$

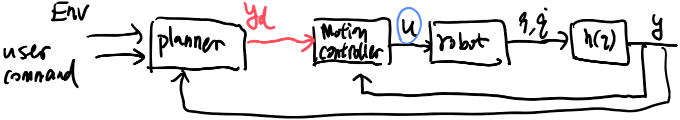
- $q \in \mathbb{R}^n$: joint positions (generalized coordinate)

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

- y: output (variable to be controlled) can be any func of q

  e.g. $y = q$, $y = T(q) \in SE(3)$

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

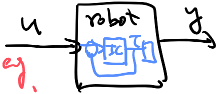


Variations in Robot Motion Control

often times q_d is given by planner represented by polynomials, so that $\dot{q}_d$, $\ddot{q}_d$ can be easily obtained

- 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 $y_d(t)$ $\in SE(3)$

- Actuation models:
  - Velocity source: $u = \dot{q}$ directly control velocity
  - Acceleration sources: $u = \ddot{q}$ directly control acc
  - Torque sources: $u = \tau$ directly control torque



eg $u = \dot{q}$

Actuation model make sense if for any given u, the joint velocity $\dot{q}$ can immediatly reach u

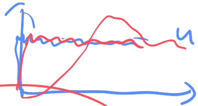
Outline

$u = \dot{q}_c \rightarrow \boxed{C} \xrightarrow{\tau} \boxed{}$

- Basic Linear Control Design
- Motion Control Problems

Linear control

$\dot{q}$



+ feedback linearization

- **Motion Control with Velocity/Acceleration as Input**
- Motion Control with Torque as Input and Task Space Inverse Dynamics

- Motion control problem $\quad u \rightarrow \boxed{\text{robot}} \rightarrow y$

  Design u to let y track desired reference y_d

- Depending on our assumption on u/y (key: Divide & conquer)

output y

- $y \leftrightarrow q \in \mathbb{R}^n$ (joint variable) → **Joint space motion control**
  - $u \leftrightarrow \dot{q} \in \mathbb{R}^n$ → $\boxed{\text{velocity-resolved Joint space control}}$
  - $\boxed{\text{Accelleration resolved joint space control}}$
  - $\boxed{\text{torque}}$
- $y \leftrightarrow T(q) \in SE(3)$ or $y = f(q) \leftarrow$ task space variable e.g. origin of end-effector frame → **Task space motion control**
  - $u \leftrightarrow \dot{q}$ → $\boxed{\text{velocity resolved Task space}}$
  - $u \leftrightarrow \ddot{q}$ → $\boxed{\ldots}$
  - $u \leftrightarrow \tau$ → $\boxed{\ldots}$

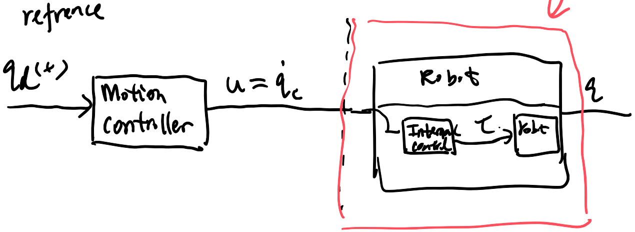
Velocity-Resolved Control

- 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

linear system



$\dot{q} = u$

$\dot{q}_1 = u_1$
$\dot{q}_2 = u_2$
$\vdots$

single integrators

Velocity-Resolved Joint Space Control

- Joint-space "dynamics": single integrator $\dot{y} = u \; \ldots \; ①$

$$\dot{q} = u$$

- 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.

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

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

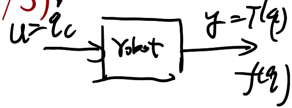
$$\dot{\tilde{q}} = -K_0\tilde{q}$$

common choice:

$$K_0 = \begin{bmatrix} k_{0,1} & & & \\ & k_{0,2} & & \\ & & \ddots & \\ & & & k_{0,n} \end{bmatrix}$$

stable if $eig(-K_0) \in OLHP$

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

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

- Taking derivatives of y, and letting $u = \dot{q}$, we have

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

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

Is this state space model?

$\dot{y} = u$

$\dot{y} = 2u$

  - Note that q is function of y through inverse kinematics.

$q = IK(y)$

  - So the above dynamics can be written in terms of y and u only. The detailed form can be quite complex in general

$\dot{y} = \underbrace{J_a(IK(y))\, u}_{v_y}$ ... ②

v_y ← virtual control

①: $\dot{y} = u$

① ⟺ ② if $J_a(\ )$ invertible

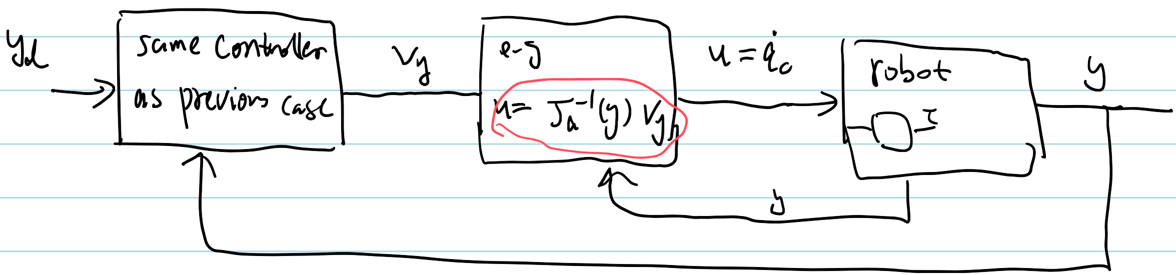
Divide & conquer : $\dot{y} = J_a(IK(y))u$

① Let v_y be virtual control. $\dot{y} = v_y$

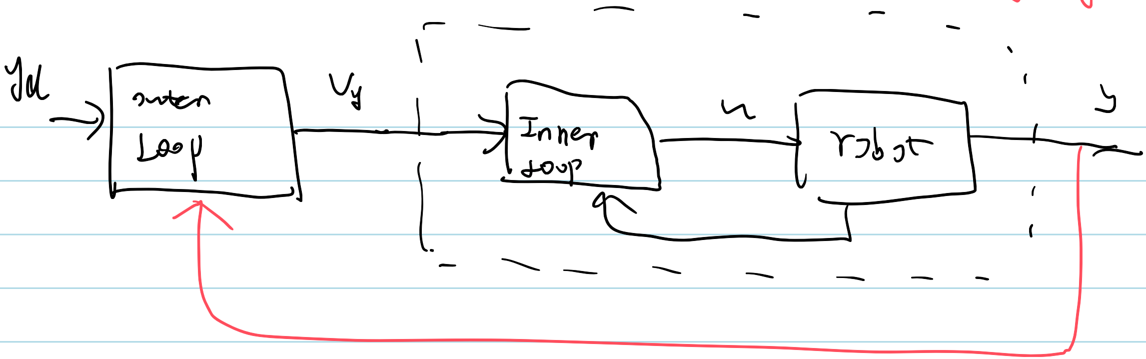
design v_y to track y_d (same as previous case)

② Find actual control u such that $J_a(IK(y))u \approx v_y$

e.g.



$\dot{y} = v_y$



- We can design outer-Loop controller as if we can directly control $\dot{y}$

$$v_y = \dot{y}_d + k(y_d - y) \xrightarrow{\text{plug in } \dot{y} = v_y} \dot{\tilde{y}} = -k\tilde{y}$$

we can select K such that $-K$ is Hurtwiz

objective of inner Loop : determine $u = \dot{q}$ such that $\dot{y} \approx v_y$

$\therefore$ we can let $\dot{y} = J_a(q)u \Rightarrow u = J_a^{+}(q)\, v_y$

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

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

e.g. $\dot{q}_{min} \le u \le \dot{q}_{max}$

$q_{min} \le q + u \cdot \Delta t \le q_{max}$

- 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 (=)

$\ddot q \Rightarrow u \to$ robot $\to y \Rightarrow q$

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

Given q_d reference, we want $q \to q_d$

$$\ddot q = u \quad (4) \qquad \text{(double integrator)}$$

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

(5); (5) into (4) $\Rightarrow$ closed-loop system

$$\underbrace{u = \ddot q_d + \underbrace{K_1 \dot{\tilde q} + K_0 \tilde q}_{\text{PD control}}}_{(5)} \Rightarrow \ddot{\tilde q} + K_1 \dot{\tilde q} + K_0 \underbrace{\tilde q}_{\in \mathbb{R}^n} = 0$$

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

- Stability condition: Let $x = \begin{bmatrix} \tilde q \\ \dot{\tilde q} \end{bmatrix} \in \mathbb{R}^{2n}$,

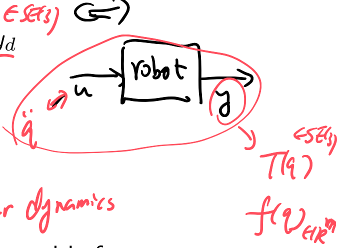
$$\dot x = \underbrace{\overbrace{\begin{bmatrix} 0 & I \\ -K_0 & -K_1 \end{bmatrix}}^{A}}_{2n \times 2n} \underbrace{\begin{bmatrix} \tilde q \\ \dot{\tilde q} \end{bmatrix}}_{x}$$

closed-loop system is stable if $\text{eig}(A) \in$ OLHP or A is Hurwitz

$$\dot x = A x$$

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

- For task space control, $y = T(q)$ needs to track y_d $\in SE(3)$



For $y = f(q)$

- Note: $\dot y = J_a(q)\dot q$ and $\ddot y = \dot J_a(q)\dot q + J_a(q)\ddot q$ u

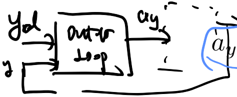
$$\ddot y = \underbrace{\dot J_a(q)\dot q + J_a(q)u}_{a_y} \quad \in \text{nonlinear dynamics}$$

- Following the same inner-outer loop strategy discussed before

– Introduce virtual control a_y such that $\ddot y = a_y$, we can design controller for a_y to let $y \to y_d$

– Inner-Loop, choose u to produce $\ddot y \approx a_y$

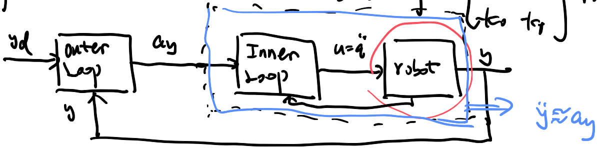
- **Outer-loop** dynamics: $\ddot y = a_y$, with a_y being the outer-loop control ~~input~~



$$a_y = \underbrace{\ddot y_d}_{\text{reference}} + \underbrace{K_1 \dot{\tilde y} + K_0 \tilde y}_{\text{PD control}} \Rightarrow \ddot{\tilde y} + K_1 \dot{\tilde y} + K_0 \tilde y = 0$$

stable if $\begin{bmatrix} 0 & I \\ -K_0 & -K_1 \end{bmatrix}$ Hurwitz.

$\tilde y = y_d - y$



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

- **Inner-loop:** Given a_y from outer loop, find the "best" joint acceleration: $\ddot{q}$

from outer loop

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

optimization variable

known

from the model

Acc. : $\ddot{q}_{min} \leq u \leq \ddot{q}_{max}$

vel : $\dot{q}_{min} \leq \dot{q} + u\Delta t \leq \dot{q}_{max}$

known at current time

- 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:

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

$$M(q) = \sum_i J_i^T \mathcal{I}_i J_i \qquad \underset{6\times 6}{\mathcal{I}_i} > 0$$

- $M(q) \in \mathbb{R}^{n\times n} \succ 0$

$$\dot{q}^T M(q)\dot{q} > 0, \ \forall \dot{q} \neq 0$$

- There are many valid definitions of $C(q, \dot{q})$, typical choice for C include:

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

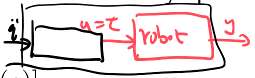
- For the above defined C, we have $\dot{M} - 2C$ is skew symmetric
- For all valid C, we have $\dot{q}^T\left[\dot{M} - 2C\right]\dot{q} = 0$
- These properties play important role in designing motion controller

Computed Torque Control (1/2)

we know how to design. controller
if $u = \ddot{q}$

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

Now :



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

- 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

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

$y = q$

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

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

Computed Torque Control (2/2)

$\tau = M\ddot{q} + c + g$

- 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.

$y = f(q)$

$\ddot{y} = \dot{J}_a(q)\dot{q} + J_a(q)M^{-1}(u - c - g)$

- Idea easily extends to task space: $\dot{y} = J_a(q)\dot{q}$ and $\ddot{y} = \dot{J}_a(q)\dot{q} + J_a(q)\ddot{q}$

$u = \tau$

$\ddot{q} = M^{-1}(u - c - g)$

- 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

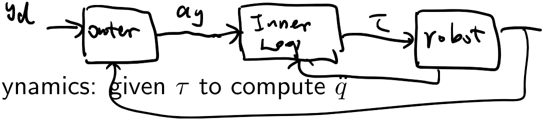
$$\begin{cases} \min_u \|a_y - \underbrace{(\dot{J}_a\dot{q} - J_a M^{-1}(u - C\dot{q} - g))}_{\ddot{y}}\|^2 \\ \text{subj. to: constraints} \end{cases} \tag{7}$$

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

$\Rightarrow u = (J_a M^{-1})^{-1}(a_y - \dot{J}_a\dot{q} \ldots)$

Inverse Dynamics Control (1/2)

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



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

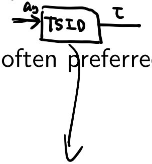
FD: from torque to motion

- Inverse dynamics: eg given desired acceleration a_q $\rightarrow \ddot{q}$, 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

Given a_y, find τ such that $\ddot{y} \approx a_y$

y is task space.



- 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:

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

$\ddot{y}$

optimization variable $u \in \mathbb{R}^n$

TSID

- This is equivalent to the following more popular form:

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

opt variable

$u, \ddot{q} \in \mathbb{R}^n$

$\in \mathbb{R}^n$

More Discussions

-

More Discussions

-

More Discussions

-