Lecture 04
Dynamics & Control

Song Chaoyang

Assistant Professor

Department of Mechanical and Energy Engineering

songcy@sustc.edu.cn

SUSTech
Southern University of Science and Technology

Dynamics

Physical laws governing the motions of bodies and aggregates of bodies.

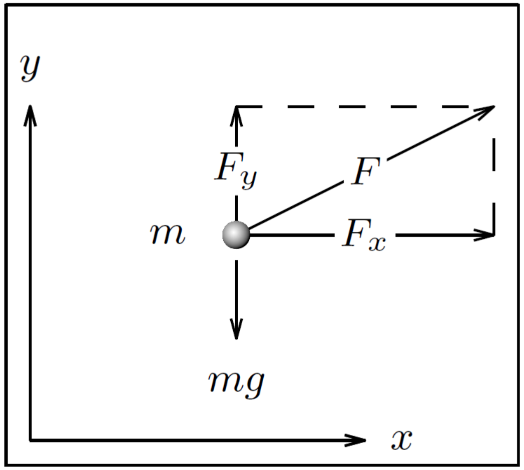
Newton's Equation:

$$\boxed{\begin{aligned} m\ddot{x} &= F_x \\ m\ddot{y} &= F_y - mg \end{aligned}}$$

Momentum: $P_x = m\dot{x}$

$$P_y = m\dot{y}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}P_x = F_x, \frac{\mathrm{d}}{\mathrm{d}t}P_y = F_y - mg$$



AncoraSIR.com

Lagrangian Equation:

$$\boxed{\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} &= F_x \\ \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{y}} - \frac{\partial L}{\partial y} &= F_y \end{aligned}}$$

Lagrangian function:

$$L = T - V, P_x = \frac{\partial L}{\partial \dot{x}}, P_y = \frac{\partial L}{\partial \dot{y}}$$

Kinetic energy:

$$T = \frac{1}{2}m(\dot{x}^2 + \dot{y}^2)$$

Potential energy:

$$V = mgy$$

Generalization to multibody systems

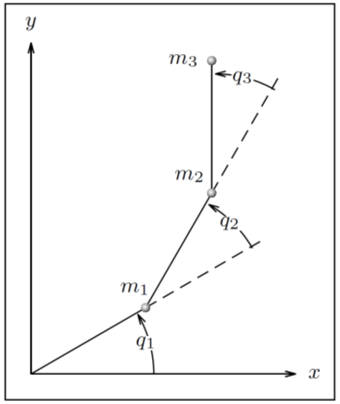


Figure 4.2

$q_i, i = 1, \ldots, n$: generalized coordinates

Kinetic energy:

$$T = T(q, \dot{q})$$

Potential energy:

$$V = V(q)$$

Lagrangian:

$$L(q, \dot{q}) = T(q, \dot{q}) - V(q)$$

$\tau_i, i = 1, \ldots, n$: external force on q_i

Lagrangian Equation:

$$\boxed{\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i, i = 1, \ldots, n}$$

SUSTech Southern University of Science and Technology

Manipulator Dynamics

Considered the forces required to cause motion

- 1^{st} Problem
  - We are given a trajectory point, Θ, $\dot{\Theta}$ and $\ddot{\Theta}$, and we wish to find the required vector of joint torques, τ.
  - Useful for the problem of controlling the manipulator

- 2^{nd} Problem
  - Given a torque vector, τ, calculate the resulting motion of the manipulator, Θ, $\dot{\Theta}$ and $\ddot{\Theta}$.
  - This is to calculate how the mechanism will move under application of a set of joint torques.
  - Useful for simulating the manipulator.

Mass Distribution

Inertia Tensor is a function of the location and orientation of the reference frame

$$
{}^{A}I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}
$$

mass moments of inertia

$$
I_{xx} = \iiint_V (y^2 + z^2)\rho dv,
$$

$$
I_{yy} = \iiint_V (x^2 + z^2)\rho dv,
$$

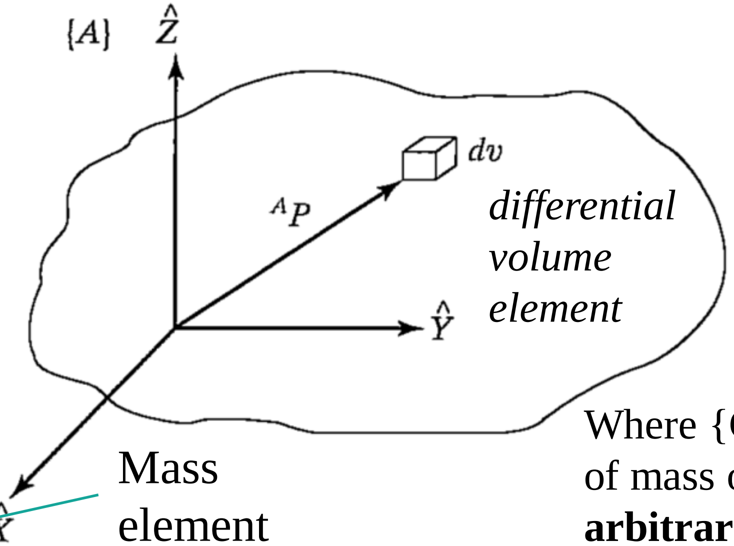
$$
I_{zz} = \iiint_V (x^2 + y^2)\boxed{\rho dv},
$$

mass products of inertia

$$
I_{xy} = \iiint_V xy\rho dv,
$$

$$
I_{xz} = \iiint_V xz\rho dv,
$$

$$
I_{yz} = \iiint_V yz\rho dv,
$$



Mass element

Material density

In practice, to measure rather than to calculate

Parallel-axis Theorem

- Compute how the inertia tensor changes under translations of the reference coordinate system

Where {C} is located at the **center** of mass of the body, and {A} is an **arbitrarily** translated frame, the theorem can be stated as

$$
{}^{A}I_{zz} = {}^{C}I_{zz} + m(x_c^2 + y_c^2),
$$

$$
{}^{A}I_{xy} = {}^{C}I_{xy} - m x_c y_c,
$$

$$
P_c = [x_c, y_c, z_c]^T
$$

$$
{}^{A}I = {}^{C}I + m[P_c^T P_c I_3 - P_c P_c^T],
$$

Principal axes are the axes of the reference frame aligned in a way that cause the **products** of inertia to be **zero**.

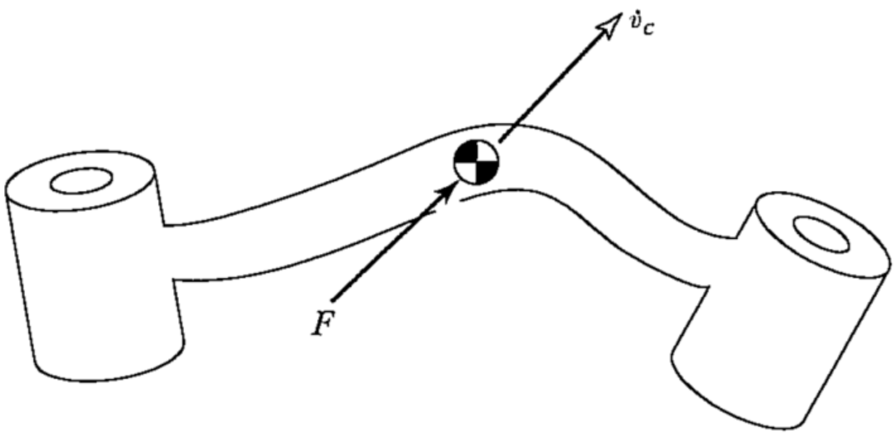
- Greatly simplifies the calculation
- The corresponding mass moments are the **principal** moments of inertia.

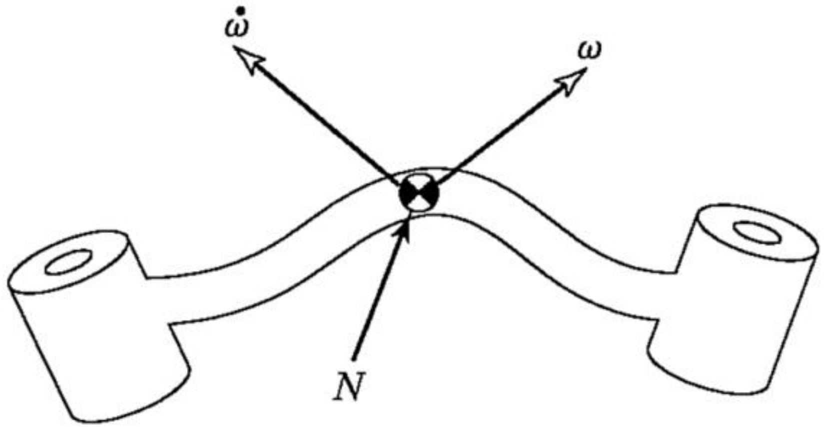
AncoraSIR.com

Newton's Equation, Euler's Equation

We will consider each link of a manipulator as a rigid body

- Newton’s equation, along with its rotational analog, Euler's equation, describes how forces, inertias, and accelerations relate.





$$F = m\dot{v}_C$$

$$N = {}^{C}I\dot{\omega} + \omega \times {}^{C}I\omega$$

Iterative Newton-Euler Dynamic Formulation

Computing the torques that correspond to a given trajectory of a manipulator.

- To calculate the joint torques required to cause motion.
  - **Known variables**: the position, velocity, and acceleration of the joints, (Θ, $\dot{\Theta}$ and $\ddot{\Theta}$).
  - **Known parameters**: the kinematics, and mass-distribution information of the robot.
- Outward iterations to compute velocities and accelerations

$$^{i+1}\omega_{i+1} = {}^{i+1}_{i}R\, {}^{i}\omega_i + \dot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1}$$

transforming angular / linear acceleration from one link to the next

Link 1 simplest form in free motion as zero

(for joint $i + 1$ rotational)

$$^{i+1}\dot{\omega}_{i+1} = {}^{i+1}_{i}R\, {}^{i}\dot{\omega}_i + {}^{i+1}_{i}R\, {}^{i}\omega_i \times \dot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1} + \ddot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1}$$

$$^{i+1}\dot{v}_{i+1} = {}^{i+1}_{i}R[{}^{i}\omega_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i]$$

(for joint $i + 1$ prismatic)

$$^{i+1}\dot{\omega}_{i+1} = {}^{i+1}_{i}R\, {}^{i}\omega_i$$

$$^{i+1}\dot{v}_{i+1} = {}^{i+1}_{i}R({}^{i}\dot{\omega}_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i) + 2\, {}^{i+1}\omega_{i+1} \times \dot{d}_{i+1}\, {}^{i+1}\hat{Z}_{i+1} + \ddot{d}_{i+1}\, {}^{i+1}\hat{Z}_{i+1}.$$

The linear acceleration of the center of mass of each link

$$^{i}\dot{v}_{C_i} = {}^{i}\dot{\omega}_i \times {}^{i}P_{C_i} + {}^{i}\omega_i \times ({}^{i}\omega_i + {}^{i}P_{C_i}) + {}^{i}\dot{v}_i$$

Apply the Newton-Euler equations for forces and torques acting on each link

$$F_i = m\dot{v}_{C_i},$$

$$N_i = {}^{C_i}I\dot{\omega}_i + \omega_i \times {}^{C_i}I\omega_i,$$

SUSTech
Southern University of Science and Technology

AncoraSIR.com

Iterative Newton-Euler Dynamic Formulation

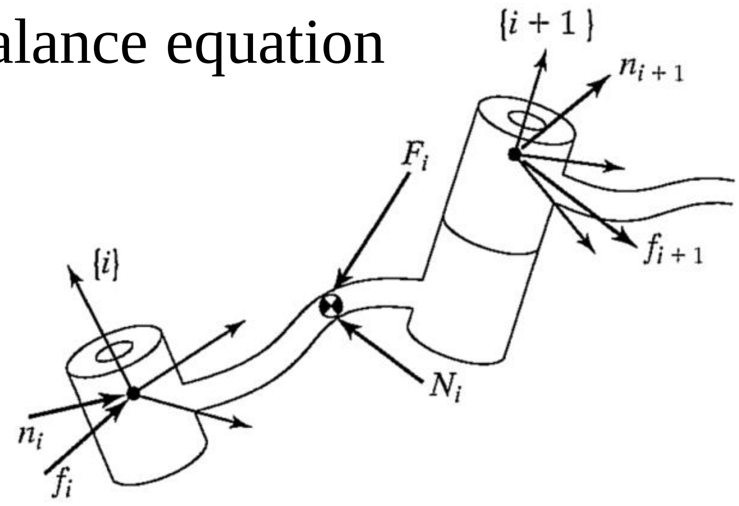
Calculate the joint torques that will result in net forces and torques being applied to each link

- Inward iterations to compute forces and torques
  - Write a force-balance and moment-balance equation

$$^{i}F_i = {}^{i}f_i - {}^{i}_{i+1}R^{i+1}f_{i+1}$$

$$^{i}N_i = {}^{i}n_i - {}^{i}n_{i+1} + (-{}^{i}P_{C_i}) \times {}^{i}f_i - ({}^{i}P_{i+1} - {}^{i}P_{C_i}) \times {}^{i}f_{i+1}$$

Rearrange so that they appear as **iterative** relationships **from higher** numbered neighbor **to lower** numbered neighbor



$$^{i}f_i = {}^{i}_{i+1}R\,{}^{i+1}f_{i+1} + {}^{i}F_i,$$

$$^{i}n_i = {}^{i}N_i + {}^{i}_{i+1}R\,{}^{i+1}n_{i+1} + {}^{i}P_{C_i} \times {}^{i}F_i + {}^{i}P_{i+1} \times {}^{i}_{i+1}R\,{}^{i+1}f_{i+1}.$$

f_i = force exerted on link i by link $i-1$,
n_i = torque exerted on link i by link $i-1$.

The required joint torques

$\tau_i = {}^{i}n_i^{T}\,{}^{i}\hat{Z}_i$ Rotational

$\tau_i = {}^{i}f_i^{T}\,{}^{i}\hat{Z}_i$ Prismatic

Link n simplest form in free motion as zero

AncoraSIR.com

The Iterative Newton-Euler Dynamics Algorithm

Velocities and Accelerations from link 1 to *n*, then Force and Torque from link *n* to 1

Outward iterations: $i : 0 \rightarrow 5$

$$^{i+1}\omega_{i+1} = {}^{i+1}_{i}R\, {}^{i}\omega_i + \dot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1},$$

$$^{i+1}\dot{\omega}_{i+1} = {}^{i+1}_{i}R\, {}^{i}\dot{\omega}_i + {}^{i+1}_{i}R\, {}^{i}\omega_i \times \dot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1} + \ddot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1},$$

$$^{i+1}\dot{v}_{i+1} = {}^{i+1}_{i}R({}^{i}\dot{\omega}_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i),$$

$$^{i+1}\dot{v}_{C_{i+1}} = {}^{i+1}\dot{\omega}_{i+1} \times {}^{i+1}P_{C_{i+1}} + {}^{i+1}\omega_{i+1} \times ({}^{i+1}\omega_{i+1} \times {}^{i+1}P_{C_{i+1}}) + {}^{i+1}\dot{v}_{i+1},$$

$$^{i+1}F_{i+1} = m_{i+1}\, {}^{i+1}\dot{v}_{C_{i+1}},$$

$$^{i+1}N_{i+1} = {}^{C_{i+1}}I_{i+1}\, {}^{i+1}\dot{\omega}_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{C_{i+1}}I_{i+1}\, {}^{i+1}\omega_{i+1}.$$

Inward iterations: $i : 6 \rightarrow 1$

$$^{i}f_i = {}^{i}_{i+1}R\, {}^{i+1}f_{i+1} + {}^{i}F_i,$$

$$^{i}n_i = {}^{i}N_i + {}^{i}_{i+1}R\, {}^{i+1}n_{i+1} + {}^{i}P_{C_i} \times {}^{i}F_i + {}^{i}P_{i+1} \times {}^{i}_{i+1}R\, {}^{i+1}f_{i+1},$$

$$\tau_i = {}^{i}n_i^T\, {}^{i}\hat{Z}_i.$$

- Inclusion of gravity forces in the dynamics algorithm

$$^{0}\dot{v}_0 = G$$

  - G has the magnitude of the gravity vector but points in the opposite direction
  - Equivalent to saying that the base of the robot is accelerating upward with 1 g acceleration

- Usage
  - as a **numerical** computational algorithm,
    - to compute the joint torques corresponding to any motion
  - as an algorithm used **analytically** to develop symbolic equations
    - obtaining better insight into the structure of the equations

AncoraSIR.com

The Structure Of A Manipulator's Dynamic Equations

Express the dynamic equations of a manipulator in a single equation

- The state-space equation
  - The Newton-Euler equations are evaluated symbolically for any manipulator
  - The term $V(\Theta, \dot{\Theta})$ has both position and velocity dependence

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

n x *n* mass matrix of the manipulator

n x 1 vector of centrifugal and Coriolis terms

n x 1 vector of gravity terms

- The configuration-space equation
  - The matrices are functions only of manipulator position

$$[\dot{\theta}_1^2 \ \dot{\theta}_2^2 \ldots \dot{\theta}_n^2]^T$$

$$\tau = M(\Theta)\ddot{\Theta} + B(\Theta)[\dot{\Theta}\dot{\Theta}] + C(\Theta)[\dot{\Theta}^2] + G(\Theta)$$

n x n(n -1)/2 matrix of Coriolis coefficients

n(n - 1)/2 x 1 vector of joint velocity products

n x n matrix of centrifugal coefficients

$$[\dot{\Theta}\dot{\Theta}] = [\dot{\theta}_1\dot{\theta}_2 \ \dot{\theta}_1\dot{\theta}_3 \ \ldots \ \dot{\theta}_{n-1}\dot{\theta}_n]^T$$

Lagrangian Formulation Of Manipulator Dynamics

An "energy-based" approach to dynamics, rather than a "force-based" one

- Kinetic Energy
  - Start with the kinetic energy, k_i, of the ith link
  - The total kinetic energy of the manipulator

$$k_i = \frac{1}{2} m_i v_{C_i}^T v_{C_i} + \frac{1}{2} {}^i\omega_i^T \, {}^{C_i}I_i \, {}^i\omega_i$$

linear velocity

angular velocity

$$k = \sum_{i=1}^{n} k_i.$$

$$k(\Theta, \dot{\Theta}) = \frac{1}{2} \dot{\Theta}^T M(\Theta) \dot{\Theta},$$

quadratic form

- Potential Energy
  - Next, the potential energy, u_i, of the ith link
  - The total potential energy of the manipulator

$$u_i = -m_i \, {}^0g^T \, {}^0P_{C_i} + u_{ref_i}$$

the 3 x 1 gravity vector

the vector locating the center of mass of the ith link

a constant chosen so that the minimum value of u_i is zero

$$u = \sum_{i=1}^{n} u_i$$

- Formulate the Lagrangian of a manipulator

$$\mathcal{L}(\Theta, \dot{\Theta}) = k(\Theta, \dot{\Theta}) - u(\Theta)$$

$$\frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{\Theta}} - \frac{\partial \mathcal{L}}{\partial \Theta} = \tau,$$

the n x 1 vector of actuator torques

$$\frac{d}{dt} \frac{\partial k}{\partial \dot{\Theta}} - \frac{\partial k}{\partial \Theta} + \frac{\partial u}{\partial \Theta} = \tau$$

AncoraSIR.com

SUSTech
Southern University of Science and Technology

Formulating Manipulator Dynamics In Cartesian Space

Manipulator Dynamics in *Joint Space* **at the End-Effector**

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

$$\tau = J^T(\Theta)\mathcal{F}$$

the Jacobian, usually in the Tool Frame $\{T\}$

- **The Cartesian State-Space Equation**

Cartesian velocity vector

A force-torque vector acting on the end-effector

$$\mathcal{F} = M_x(\Theta)\ddot{\chi} + V_x(\Theta, \dot{\Theta}) + G_x(\Theta)$$

Cartesian mass matrix

An appropriate Cartesian vector representing position and orientation of the end-effector

A vector of gravity terms in Cartesian space

$$\mathcal{F} = J^{-T}\tau$$

$$= J^{-T}M(\Theta)\ddot{\Theta} + J^{-T}V(\Theta, \dot{\Theta}) + J^{-T}G(\Theta),$$

$$\dot{\chi} = J\dot{\Theta},$$ the definition of the Jacobian

$$= J^{-T}M(\Theta)J^{-1}\ddot{\chi} - J^{-T}M(\Theta)J^{-1}\dot{J}\dot{\Theta}$$
$$+ J^{-T}V(\Theta, \dot{\Theta}) + J^{-T}G(\Theta),$$

$$M_x(\Theta) = J^{-T}(\Theta)M(\Theta)J^{-1}(\Theta),$$
$$V_x(\Theta, \dot{\Theta}) = J^{-T}(\Theta)(V(\Theta, \dot{\Theta}) - M(\Theta)J^{-1}(\Theta)\dot{J}(\Theta)\dot{\Theta}),$$
$$G_x(\Theta) = J^{-T}(\Theta)G(\Theta).$$

The Cartesian Configuration Space Torque Equation

$$\tau = J^T(\Theta)(M_x(\Theta)\ddot{\chi} + V_x(\Theta, \dot{\Theta}) + G_x(\Theta)).$$
$$= J^T(\Theta)M_x(\Theta)\ddot{\chi} +$$
$$B_x(\Theta)[\dot{\Theta}\dot{\Theta}] + C_x(\Theta)[\dot{\Theta}^2] + G(\Theta),$$

Coriolis coefficients

centrifugal coefficients

$$[\dot{\Theta}\dot{\Theta}] = [\dot{\theta}_1\dot{\theta}_2 \ \ \dot{\theta}_1 \ \dot{\theta}_3 \ \ldots \dot{\theta}_{n-1}\dot{\theta}_n]^T, \qquad [\dot{\theta}_1^2 \ \ \dot{\theta}_2^2 \ \ldots \dot{\theta}_n^2]^T$$

AncoraSIR.com

SUSTech Southern University of Science and Technology

Inclusion Of Nonrigid Body Effects

The most important source of forces that are not included is ***friction***

- In present-day manipulators, in which significant gearing is typical, the forces due to friction can actually be quite large
  - Perhaps equaling **25%** of the torque required to move the manipulator in typical situations.

- A simple model: *viscous friction* $\tau_{friction} = v\dot{\theta}$ $\tau_{friction} = c\, sgn(\dot{\theta}) + v\dot{\theta}.$
  - The torque due to friction is proportional to the velocity of joint motion.
- Another simple model: *Coulomb friction* $\tau_{friction} = c\, sgn(\dot{\theta}),$
  - Constant except for a sign dependence on the joint velocity
- In many manipulator joints, friction also displays a *dependence on the joint position*

$$\tau_{friction} = f(\theta, \dot{\theta})$$

- **A more complete dynamic model with friction**

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

  - *Usually very difficult to model, like bending effects (which give rise to resonances)*

AncoraSIR.com

Dynamic Simulation

simulation requires solving the dynamic equation for acceleration

$$\ddot{\Theta} = M^{-1}(\Theta)[\tau - V(\Theta, \dot{\Theta}) - G(\Theta) - F(\Theta, \dot{\Theta})].$$

- We can then apply any of several known numerical integration techniques to integrate the acceleration to compute future positions and velocities
- Given initial conditions on the motion of the manipulator

$$\Theta(0) = \Theta_0,$$
$$\dot{\Theta}(0) = 0,$$

- We integrate forward in time numerically by steps of size Δt
- ***Euler integration***: starting with Δt = 0, iteratively compute

$$\dot{\Theta}(t + \Delta t) = \dot{\Theta}(t) + \ddot{\Theta}(t)\Delta t,$$
$$\Theta(t + \Delta t) = \Theta(t) + \dot{\Theta}(t)\Delta t + \frac{1}{2}\ddot{\Theta}(t)\Delta t^2,$$

for each iteration

- In this way, the position, velocity, and acceleration of the manipulator caused by a certain input torque function can be computed numerically.
  - Simple but more advanced options are can be used for accurate and efficient simulation
  - Other issues such as the select of step size Δt

AncoraSIR.com

Computational Considerations

- **A historical note concerning efficiency**
  - Counting the number of multiplications and additions
  - When taking into consideration the simple first outward computation and simple last inward computation
- **Efficiency of closed form vs. that of iterative form**
  - If manipulators are designed to be *simple* in the kinematic and dynamic sense, they will have dynamic equations that are simple.
    - Kinematically Simple and Dynamically Simple
- **Efficient dynamics for simulation**
  - address both the computation of the dynamic equations studied in this chapter and efficient schemes for solving equations (for joint accelerations) and performing numerical integration
- **Memorization schemes**
  - The size of the memory required is large

Newton-Euler Iterative approach

$126n - 99$ multiplications,

$106n - 92$ additions,

Lagrangian approach

$32n^4 + 86n^3 + 171n^2 + 53n - 128$ multiplications,

$25n^4 + 66n^3 + 129n^2 + 42n - 96$ additions.

AncoraSIR.com

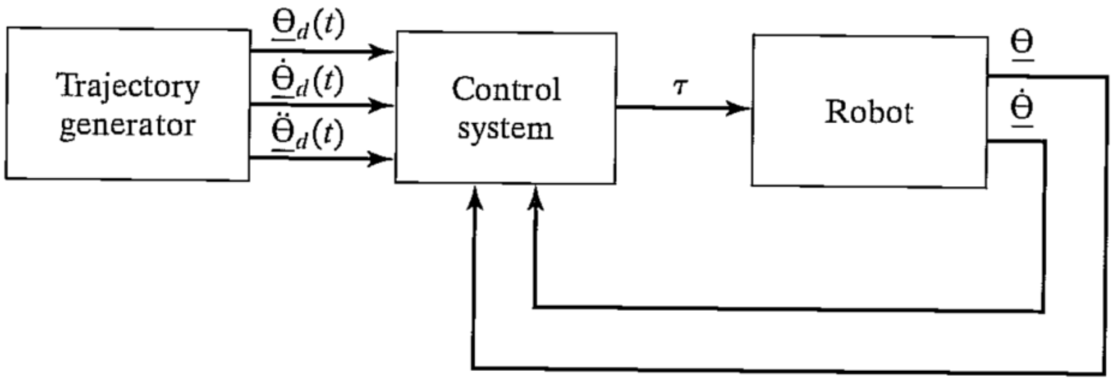
Linear & Nonlinear Control of Manipulators

Linear methods must essentially be viewed as approximate methods of nonlinear system

- Linear control techniques
  - Only valid when the system being studied can be modeled mathematically by *linear* differential equations
- Nonlinear control techniques
  - However, the dynamics of a manipulator are more properly represented by a *nonlinear* differential equation
- Method of Approximation
  - It is often reasonable to make such approximations,
  - It also is the case that these linear methods are the ones most often used in current industrial practice.
  - Consideration of the linear approach will serve as a basis for the more complex treatment of nonlinear control systems

Feedback and Closed-Loop Control

Model a manipulator as a mechanism that is instrumented with sensors at each joint to measure the joint angle and that has an actuator at each joint to apply a torque on the neighboring (next higher) link.



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

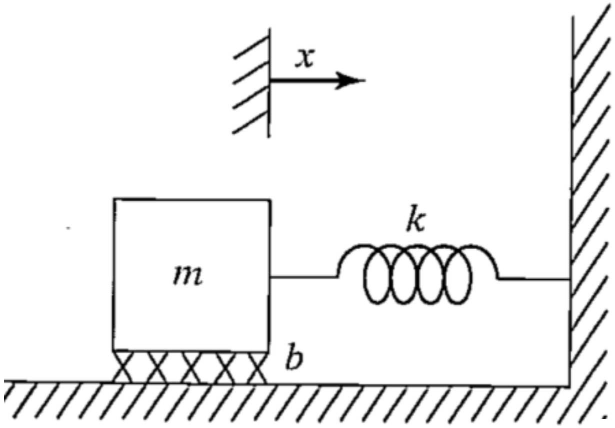
$$E = \Theta_d - \Theta,$$
$$\dot{E} = \dot{\Theta}_d - \dot{\Theta}.$$

- The basic goal of robot feedback control
  - Stable motion
  - Satisfactory performance

AncoraSIR.com

Second-Order Linear System

General Concept Review

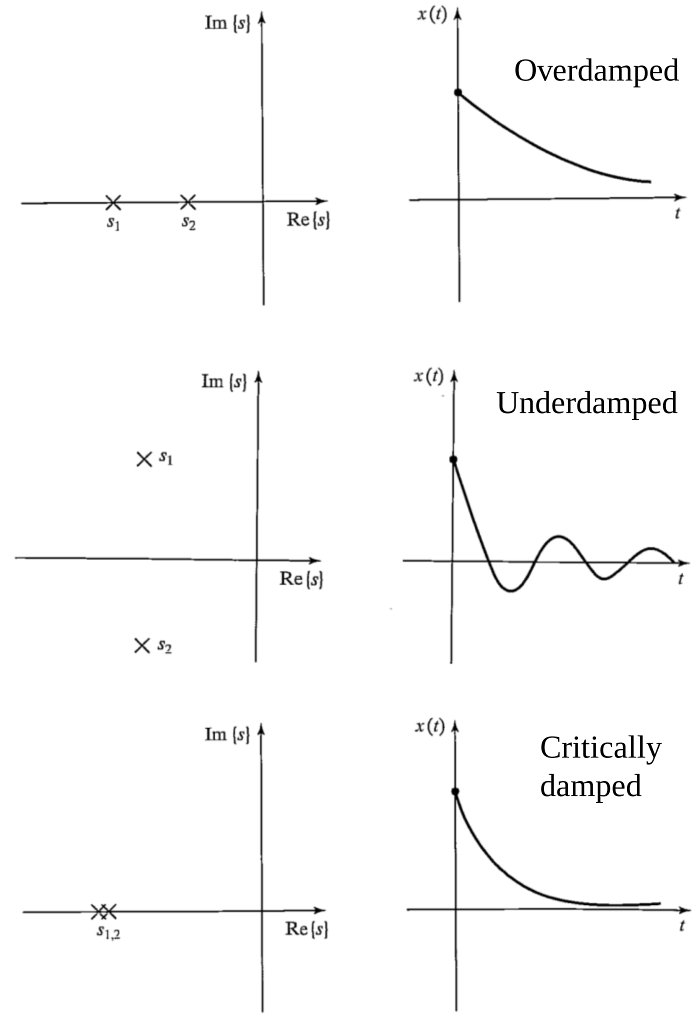


$$m\ddot{x} + b\dot{x} + kx = 0.$$

$$ms^2 + bs + k = 0.$$

$$s_1 = -\frac{b}{2m} + \frac{\sqrt{b^2 - 4mk}}{2m},$$

$$s_2 = -\frac{b}{2m} - \frac{\sqrt{b^2 - 4mk}}{2m}.$$



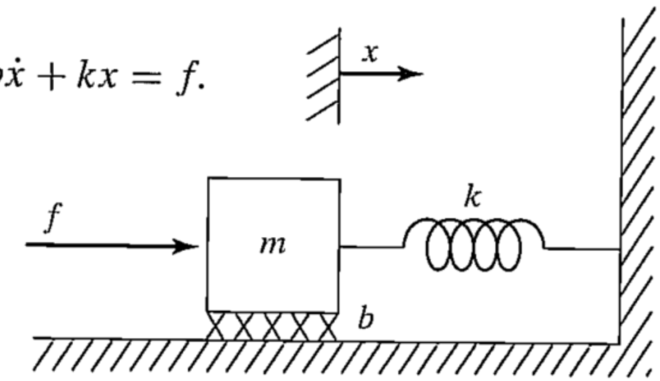
1. **Real and Unequal Roots**. This is the case when $b^2 > 4\,mk$; that is, friction dominates, and sluggish behavior results. This response is called **overdamped**.
2. **Complex Roots**. This is the case when $b^2 < 4\,mk$; that is, stiffness dominates, and oscillatory behavior results. This response is called **underdamped**.
3. **Real and Equal Roots**. This is the special case when $b^2 = 4\,mk$; that is, friction and stiffness are "balanced," yielding the fastest possible nonoscillatory response. This response is called **critically damped**.

AncoraSIR.com

Control of Second-Order Systems

Through the use of sensors, an actuator, and a control system, we can modify the system's behavior as desired

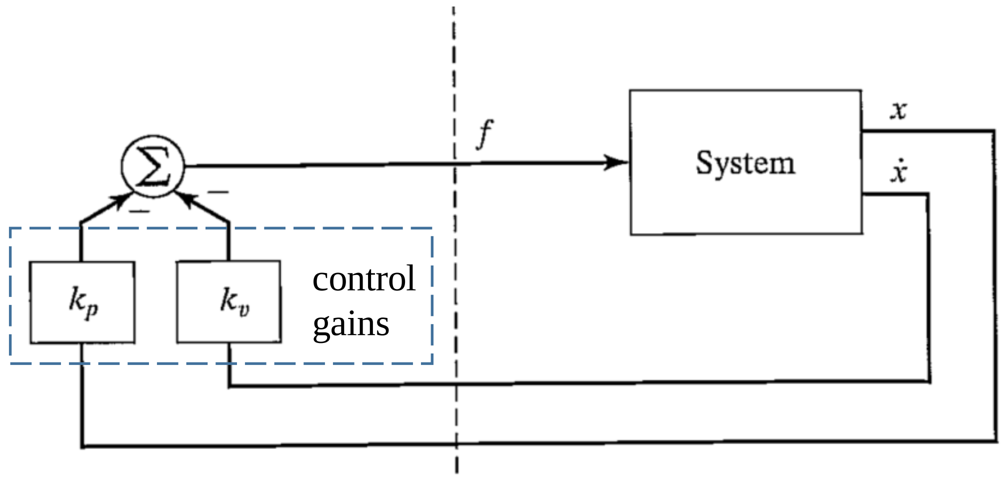
$$m\ddot{x} + b\dot{x} + kx = f.$$



a damped spring-mass system with the addition of an actuator with which it is possible to apply a force f to the block.

$$f = -k_p x - k_v \dot{x}.$$

a **control law** which computes the force that should be applied by the actuator as a function of the sensed feedback



a **position-regulation** system

- it simply attempts to maintain the position of the block in one fixed place regardless of disturbance forces applied to the block

$$m\ddot{x} + (b + k_v)\dot{x} + (k + k_p)x = 0,$$

$$m\ddot{x} + b'\dot{x} + k'x = 0,$$

by setting the control gains, we can cause the closed-loop system to appear to have any second system behavior that we wish

- Critical damping is often natural choice

AncoraSIR.com

Control-Law Partitioning

Decompose the controller into two parts: Servo & Model based portions

- Model-based portion
  - Make use of supposed knowledge of *m*, *b*, and *k*
  - *Reduces the system so that it appears to be a unit mass*

$$m\ddot{x} + b\dot{x} + kx = f. \longrightarrow f = \alpha f' + \beta,$$

$$\alpha = m,$$

$$\beta = b\dot{x} + kx.$$

$$m\ddot{x} + b\dot{x} + kx = \alpha f' + \beta.$$

$$\ddot{x} = f'.$$

If f' is taken as the *new input* to the system, *the system appears to be a unit mass*

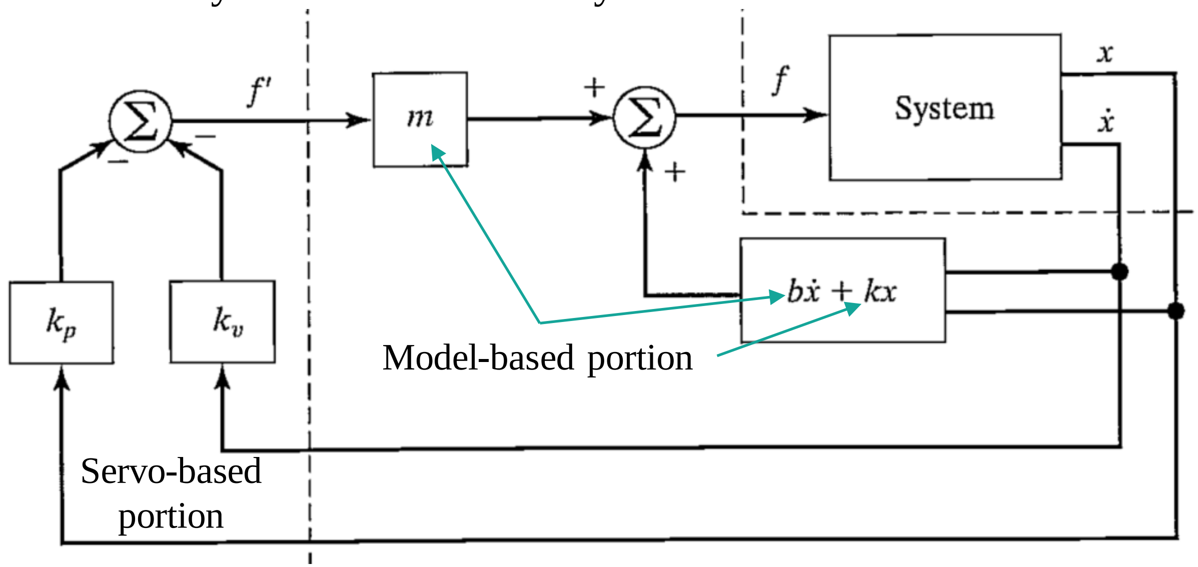
- Servo-based portion
  - Makes use of feedback to modify the behavior of the system
  - If critically damped

$$\ddot{x} + k_v\dot{x} + k_p x = 0. \longleftarrow f' = -k_v\dot{x} - k_p x.$$

$$k_v = 2\sqrt{k_p}$$

Position Control

- Maintaining at a desired location



AncoraSIR.com

Trajectory-following Control

The trajectory is given by a function of time, $x_d(t)$, that specifies the desired position of the block

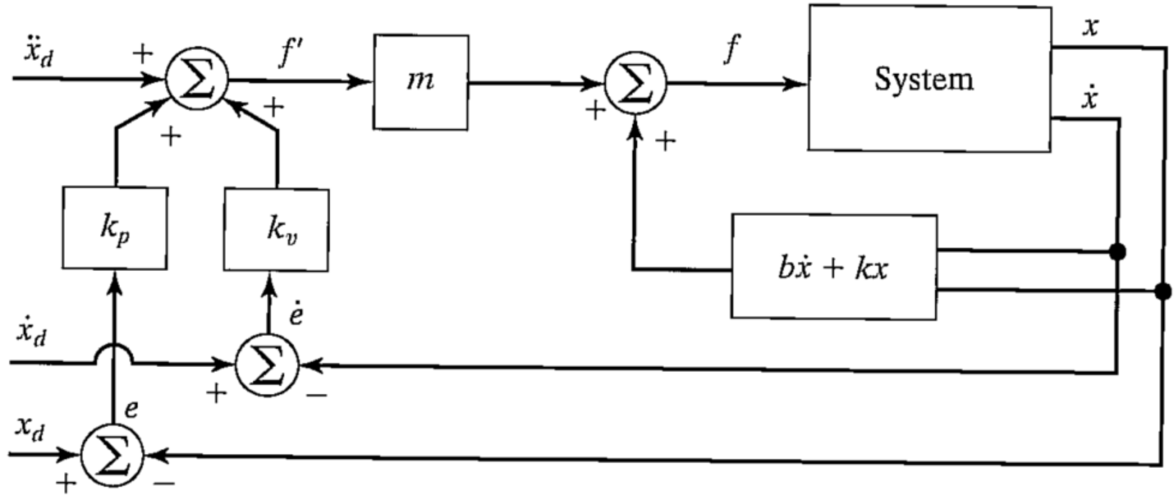
- We define the servo error between the desired and actual trajectory $e = x_d - x$
- A servo-control law that will cause trajectory following $f' = \ddot{x}_d + k_v\dot{e} + k_p e.$
  - If our model is perfect (i.e., our knowledge of *m*, *b*, and *k*), and if there is no noise and no initial error, the block will follow the desired trajectory exactly.
  - If there is an initial error, it will be suppressed according to the **error space** motion , and thereafter the system will follow the trajectory exactly.

$\ddot{x} = f'.$

a unit mass motion

$$\ddot{x} = \ddot{x}_d + k_v\dot{e} + k_p e,$$

$$\ddot{e} + k_v\dot{e} + k_p e = 0.$$



AncoraSIR.com

Disturbance Rejection

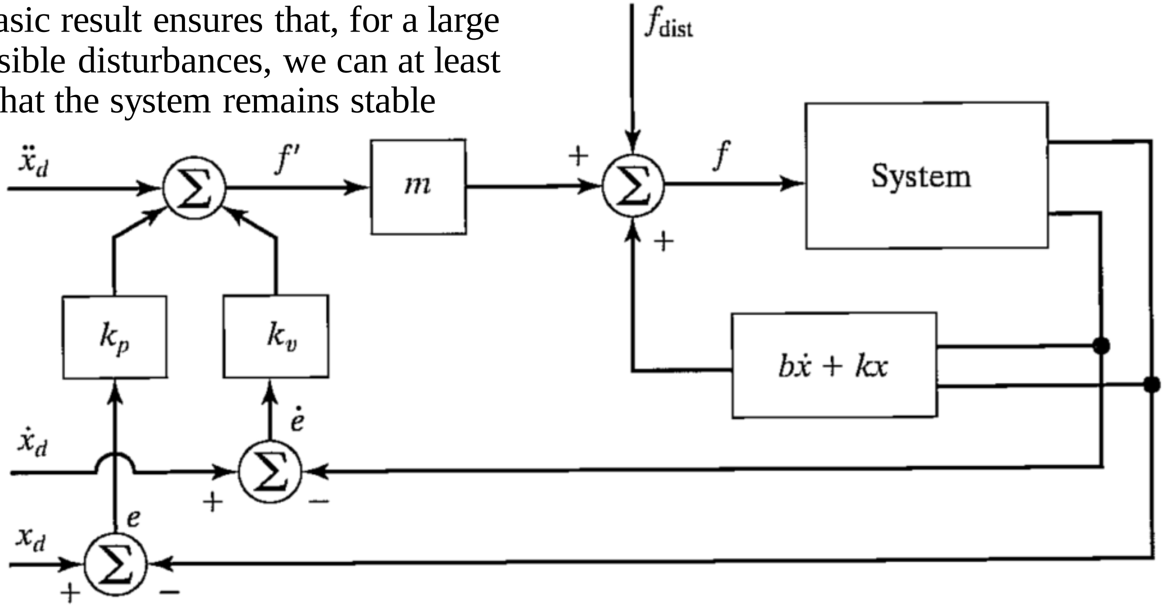
to maintain good performance (i.e., minimize errors) even in the presence of some external disturbances or noise.

$$\ddot{e} + k_v\dot{e} + k_p e = f_{\text{dist}}/m$$

- If it is known that f_{dist} is bounded—that is, that a constant a exists such that

$$\max_t f_{\text{dist}}(t) < a,$$

- Then the solution of the differential equation, *e(t)*, is also bounded
  - This result is due to a property of stable linear systems known as **bounded-input, bounded-output or BIBO** stability
  - This very basic result ensures that, for a large class of possible disturbances, we can at least be assured that the system remains stable



AncoraSIR.com

Disturbance Rejection

Steady-State Analysis & PID Control Law

$$\ddot{e} + k_v \dot{e} + k_p e = f_{\text{dist}}/m$$

Steady-state Analysis

- analyzing the system at rest (i.e., the derivatives of all system variables are zero)

$$k_p e = f_{\text{dist}}/m$$

$$e = f_{\text{dist}}/k_p m$$

steady-state error

In order to eliminate steady-state error, a modified control law is sometimes used by **adding an integral term**

$$\ddot{x} = f' = \ddot{x}_d + k_v \dot{e} + k_p e + k_i \int e dt,$$

PID Control Law, or "proportional, integral, derivative" control law

$$\ddot{e} + k_v \dot{e} + k_p e + k_i \int e dt = f_{\text{dist}}/m$$

If $e(t) = 0$ for $t < 0$, then for $t > 0$

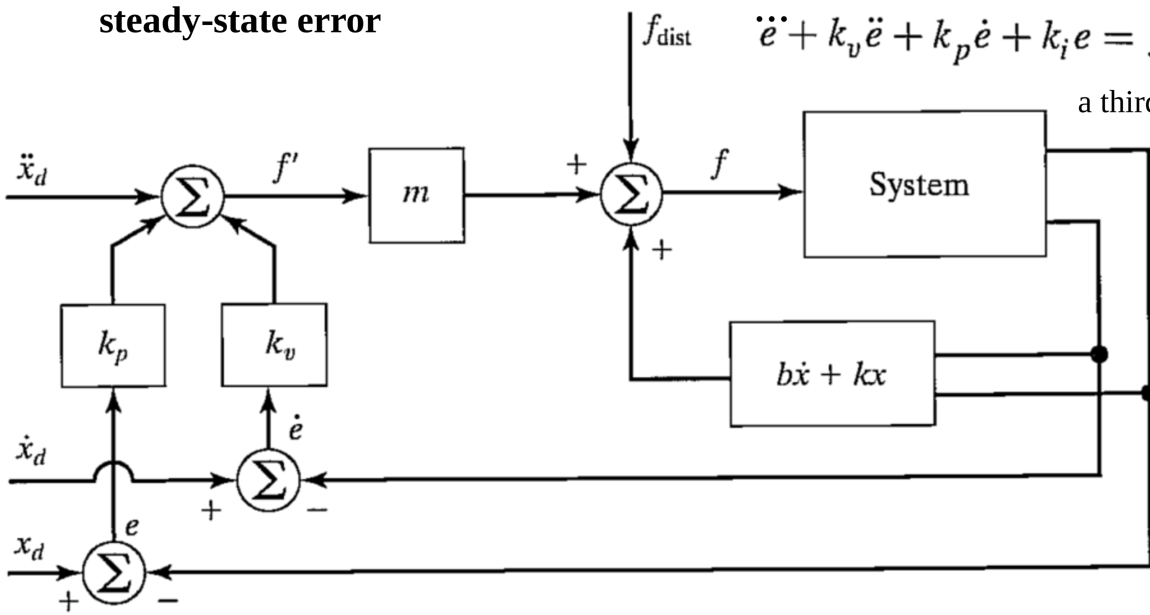
$$\dddot{e} + k_v \ddot{e} + k_p \dot{e} + k_i e = \dot{f}_{\text{dist}}/m$$

a third-order system

Steady-state Analysis

$$k_i e = 0,$$

$$e = 0.$$



Continuous Vs. Discrete Time Control

- **Servo rate**
- Tracking reference inputs
- Disturbance rejection
- Antialiasing
- Structural resonances

AncoraSIR.com

SUSTech Southern University of Science and Technology

Modeling and Control of A Single Joint

Getting Started

- Three major assumptions for analysis
  1. The motor inductance l_a can be neglected. *We can command toques directly*
  2. Taking into account high gearing, we model the effective inertia as a constant equal to $I_{max} + \eta^2 I_m$. *Inertia actually varies with the configuration and load*
  3. Structural flexibilities are neglected, except that the lowest structural resonance ω_{res} is used in setting the servo gains. *Unmodeled flexibility with a simpler dynamic model*
- With these assumptions, a single joint of a manipulator can be controlled with the partitioned controller given by

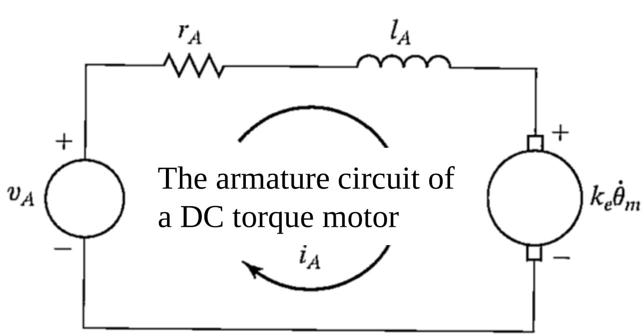
$$\alpha = I_{max} + \eta^2 I_m,$$

$$\beta = (b + \eta^2 b_m)\dot{\theta},$$

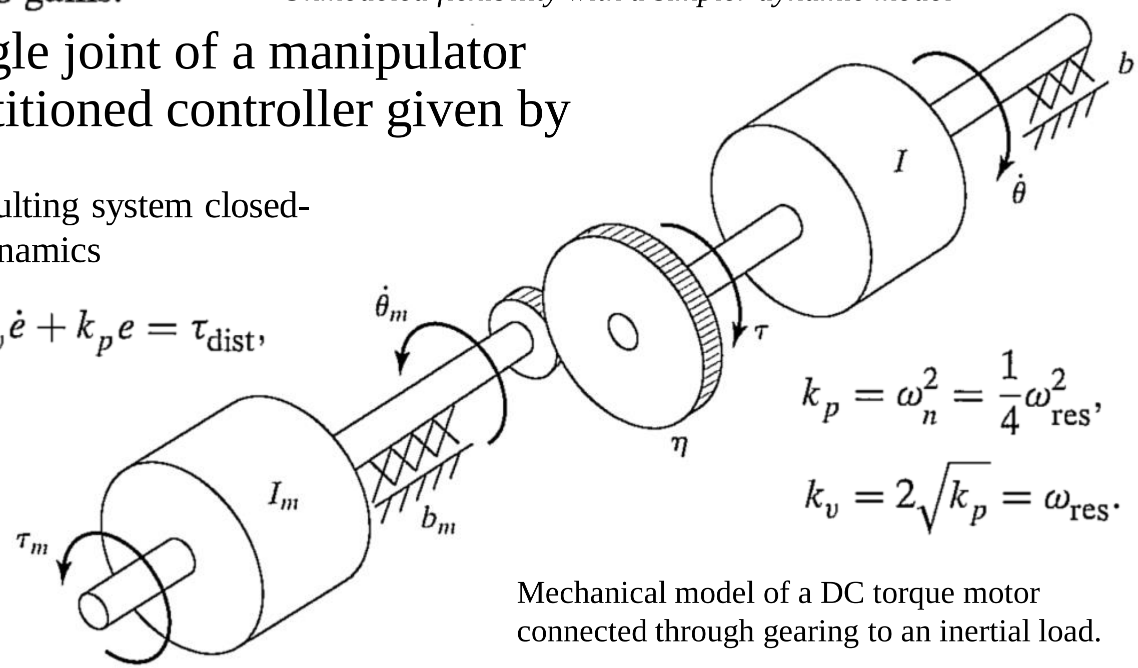
$$\tau' = \ddot{\theta}_d + k_v \dot{e} + k_p e.$$

The resulting system closed-loop dynamics

$$\ddot{e} + k_v \dot{e} + k_p e = \tau_{dist},$$



The armature circuit of a DC torque motor



$$k_p = \omega_n^2 = \frac{1}{4}\omega_{res}^2,$$

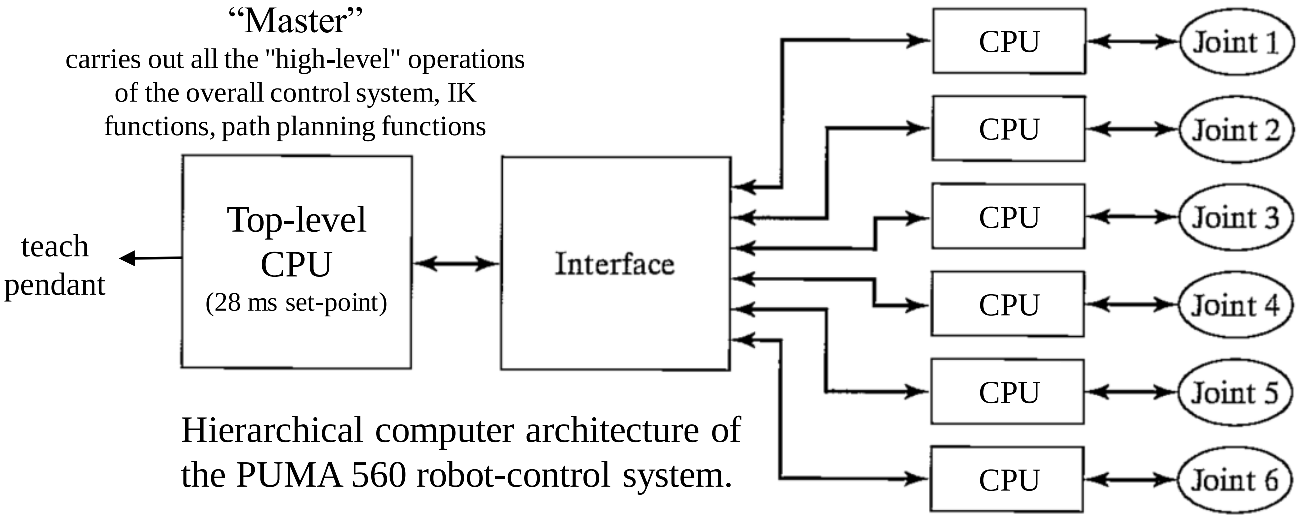
$$k_v = 2\sqrt{k_p} = \omega_{res}.$$

Mechanical model of a DC torque motor connected through gearing to an inertial load.

Architecture of An Industrial-Robot Controller

Example of the Unimation PUMA 560 industrial robot

"Master"
carries out all the "high-level" operations of the overall control system, IK functions, path planning functions

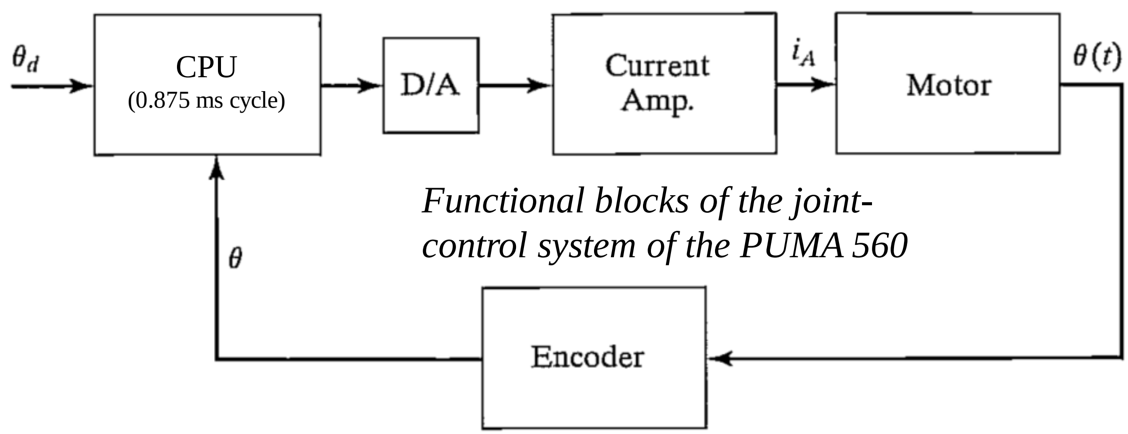


Hierarchical computer architecture of the PUMA 560 robot-control system.

Typically equipped with one optical encoder per joint for position feedback

Velocity is usually estimated by numerical differencing in the local joint controller

Lower-level Controller



Functional blocks of the joint-control system of the PUMA 560

Each low-level CPU is interfaced to a digital-to-analog (DAC) convertor so that motor current can be commanded to the current-driver circuits.

The current flowing through the motor is controlled in analog circuitry by adjusting the voltage across the armature as needed to maintain the desired armature current.

AncoraSIR.com

Nonlinear and Time-Varying Systems

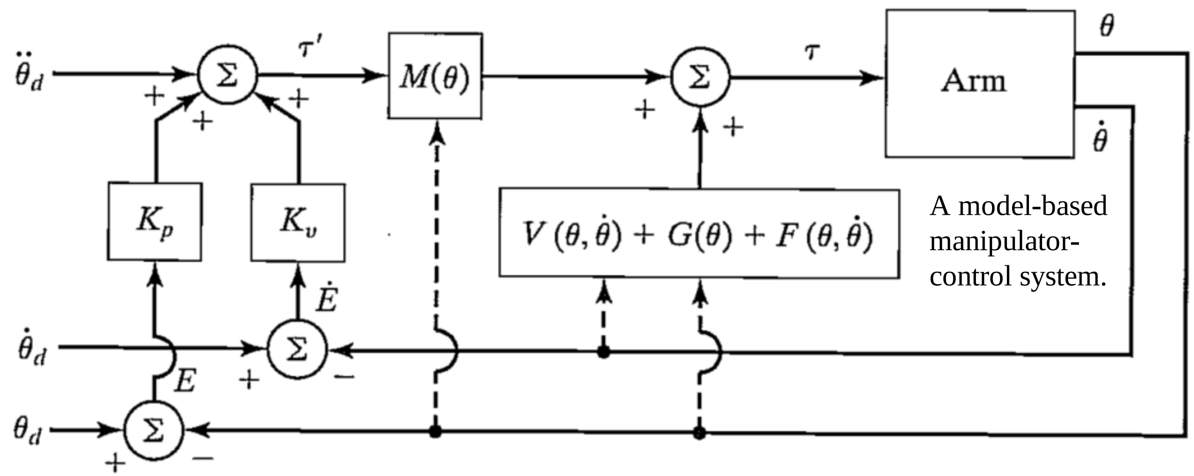
Manipulators constantly move among regions of their workspaces so widely separated that no linearization valid for all regions can be found

- Common methods to simplify the problem
  1. Compute a nonlinear model-based control law that "cancels" the nonlinearities of the system to be controlled.
  2. Reduce the system to a linear system that can be controlled with the simple linear servo law developed for the unit mass.

- Model system dynamics with Friction $\tau = M(\Theta)\ddot{\Theta} + V(\Theta, \dot{\Theta}) + G(\Theta) + F(\Theta, \dot{\Theta}).$
- Reduce to partitioned controllers $\tau = \alpha\tau' + \beta, \quad \alpha = M(\Theta),\ \beta = V(\Theta, \dot{\Theta}) + G(\Theta) + F(\Theta, \dot{\Theta}),$
- Incorporate with the servo law $\tau' = \ddot{\Theta}_d + K_v\dot{E} + K_pE, \quad E = \Theta_d - \Theta.$
- Now, our closed-loop system is characterized by the error equation $\ddot{E} + K_v\dot{E} + K_pE = 0.$

Problems makes this *perfect* model *impractical* to solve

- The discrete nature of a digital-computer implementation
- Inaccuracy in the manipulator model



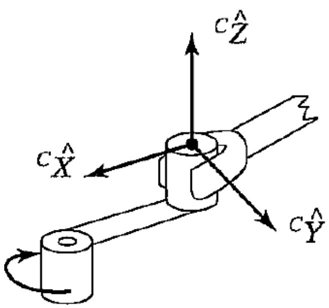
A model-based manipulator-control system.

Force Control of Manipulators

Making the *first* contact (using hybrid position/force controller)

- *Constraints* for *Subtasks* w.r.t. a *Generalized Surface*
    - **Generalized Surface**: can be defined with position constraints along the normals to this surface and force constraints along the tangents
    - **Natural Constraints**: a set of constraints that result from the particular mechanical and geometric characteristics of task configuration.
    - **Artificial Constraints**: Additional constraints introduced in accordance with the natural constraints to specify desired motions or force application
- **Assembly strategy**
    - A sequence of planned artificial constraints that in a desirable manner.
    - How should the manipulator moves for the task.

(a) Turning crank



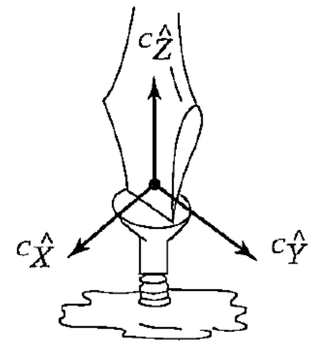
Natural constraints

$v_x = 0$ $\quad f_y = 0$
$v_z = 0$ $\quad n_z = 0$
$\omega_x = 0$
$\omega_y = 0$

Artificial constraints

$v_y = r\alpha_1$ $\quad f_x = 0$
$\omega_z = \alpha_1$ $\quad f_z = 0$
$n_x = 0$
$n_y = 0$

(b) Turning screwdriver



Natural constraints

$v_x = 0$ $\quad f_y = 0$
$\omega_x = 0$ $\quad n_z = 0$
$\omega_y = 0$
$v_z = 0$

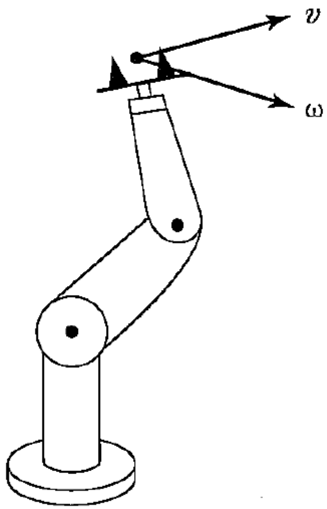
Artificial constraints

$v_y = 0$ $\quad f_x = 0$
$\omega_z = \alpha_2$ $\quad n_x = 0$
$n_y = 0$
$f_z = \alpha_3$

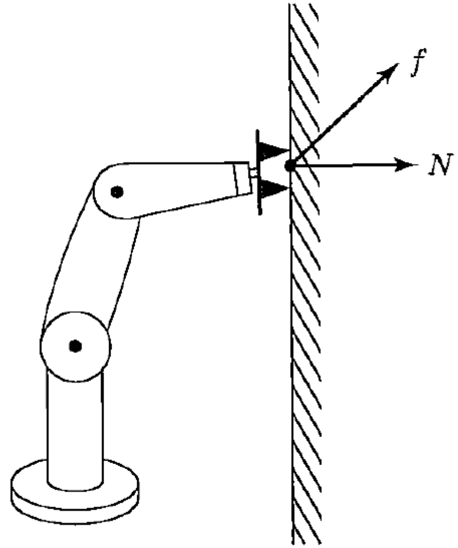
AncoraSIR.com

Hybrid Position/Force Control Problem

The natural constraints are all force constraints - there is nothing to react against, so all forces are constrained to be zero



moving in free space where no reaction surface exits



glued to the wall so that no free motion is possible

The manipulator is subject to six natural position constraints, because it is not free to be repositioned.

Three problems to solve:

1. Position control of a manipulator along directions in which a natural force constraint exists.
2. Force control of a manipulator along directions in which a natural position constraint exists.
3. A scheme to implement the arbitrary mixing of these modes along orthogonal degrees of freedom of an arbitrary frame, {C}.

Force Control Of A Mass-Spring System

$$f_e = k_e x.$$

The variable we wish to control is the force acting on the environment

Physical system

$$f = m\ddot{x} + k_e x + f_{\text{dist}},$$

an unknown disturbance force

$$f = mk_e^{-1}\ddot{f}_e + f_e + f_{\text{dist}}.$$

Using the partitioned-controller concept

$$\alpha = mk_e^{-1}$$

$$\beta = f_e + f_{\text{dist}}$$

$$f = mk_e^{-1}\left[\ddot{f}_d + k_{vf}\dot{e}_f + k_{pf}e_f\right] + \boxed{f_e + f_{\text{dist}}},$$

$$e_f = f_d - f_e$$

$$f_d$$

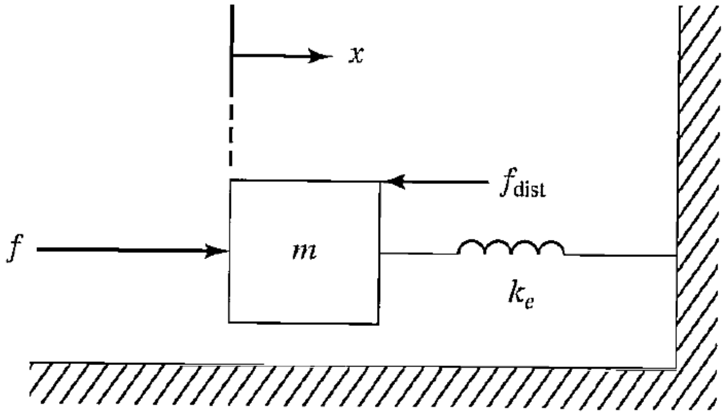
$$\ddot{e}_f + k_{vf}\dot{e}_f + k_{pf}e_f = 0.$$

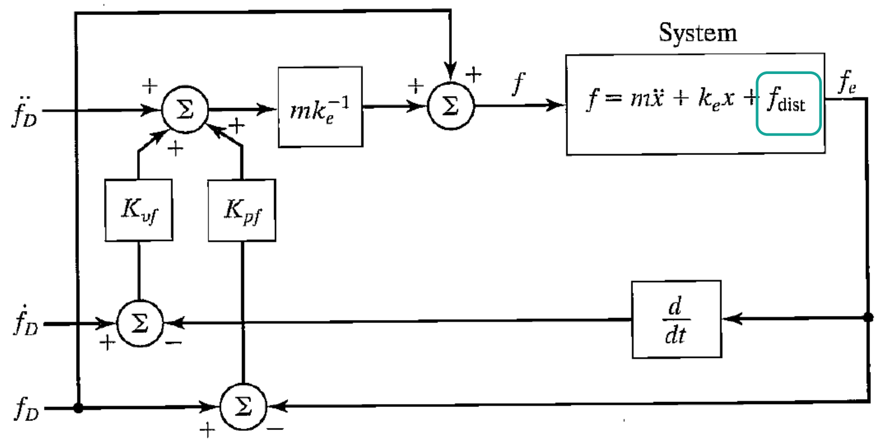
Steady-state analysis by setting all time derivatives to zeros

$$e_f = \frac{f_{\text{dist}}}{1+\alpha}.$$

steady-state error

$$f = mk_e^{-1}\left[\ddot{f}_d + k_{vf}\dot{e}_f + k_{pf}e_f\right] + f_d.$$





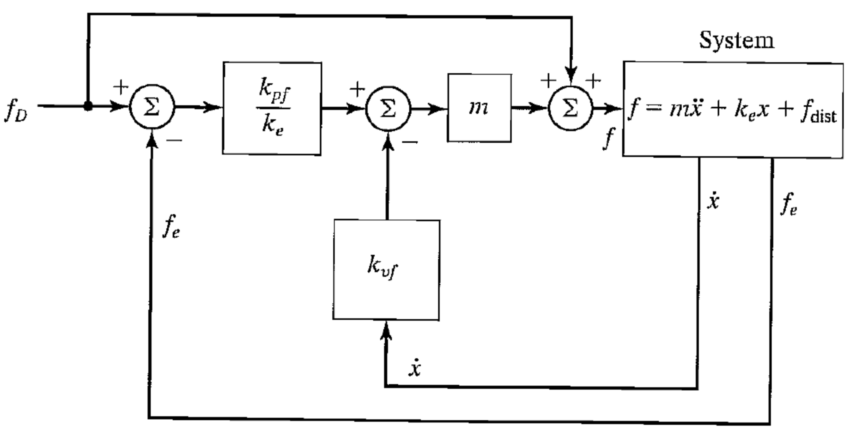
Practical Considerations

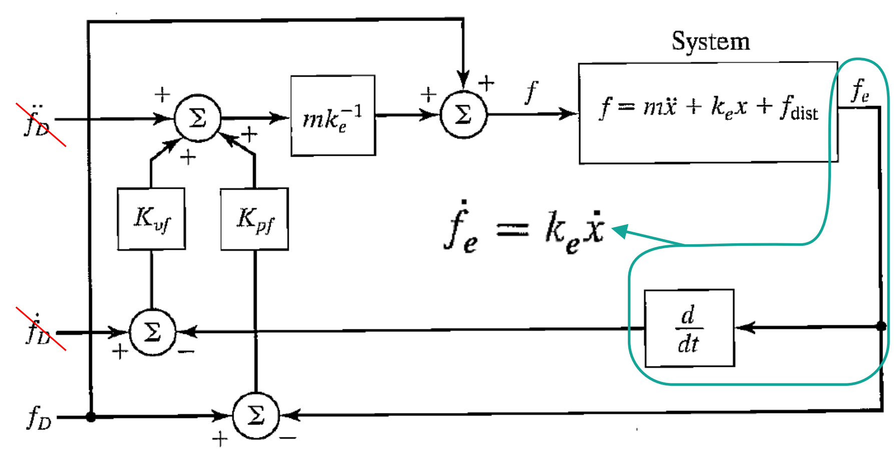
A practical force-control system for the spring-mass system.

- Force trajectories are usually constants
  - We are usually interested in controlling the contact force to be at some constant level.
- Sensed forces are quite "noisy," and numerical differentiation to compute is ill-advised.

$$f = m\left[k_{pf}k_e^{-1}e_f - k_{vf}\dot{x}\right] + f_d,$$

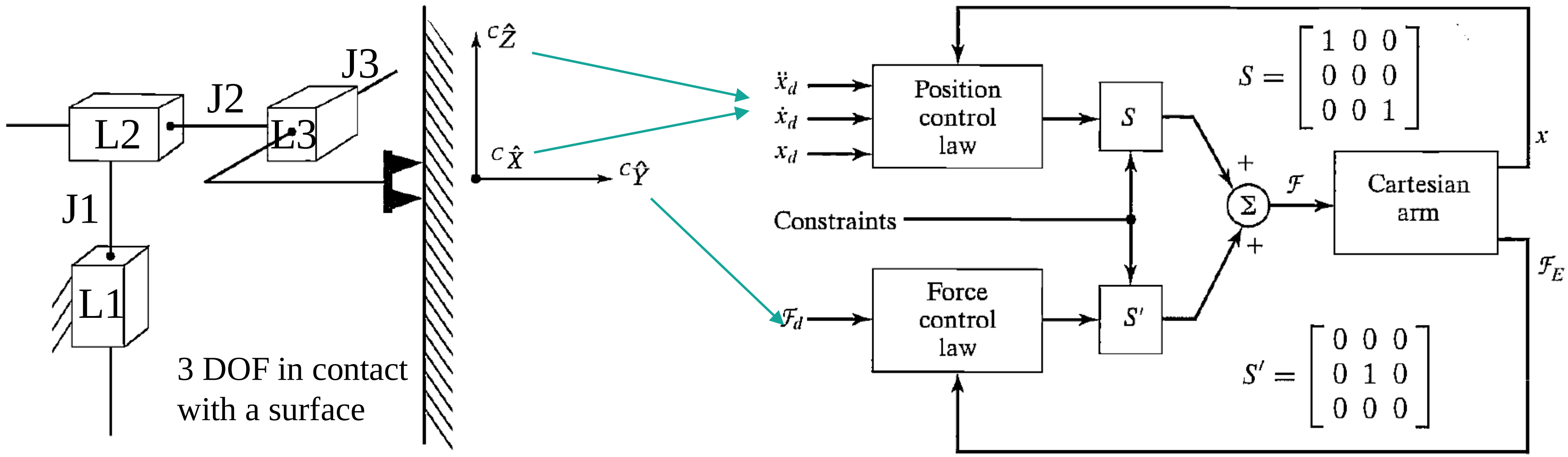
$$f = mk_e^{-1}\left[\ddot{f}_d + k_{vf}\dot{e}_f + k_{pf}e_f\right] + f_d.$$

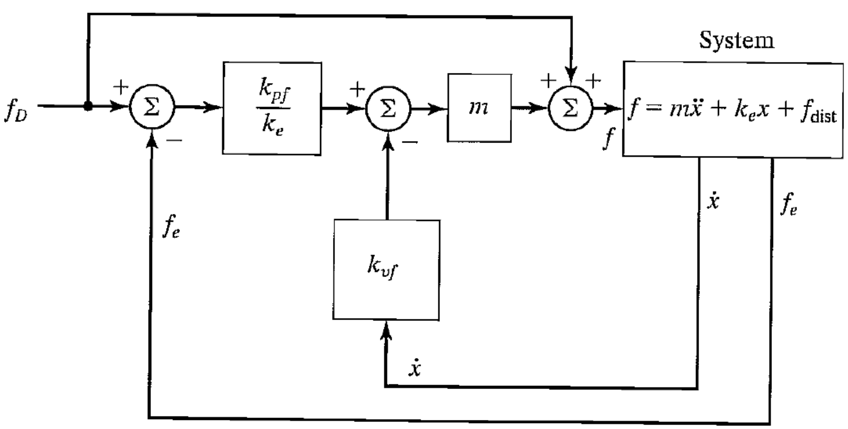


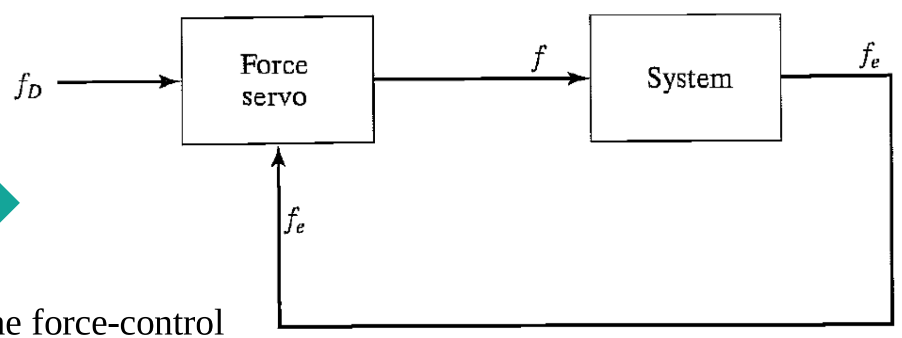


Hybrid Position/Force Control Scheme

A Cartesian manipulator aligned with {C}





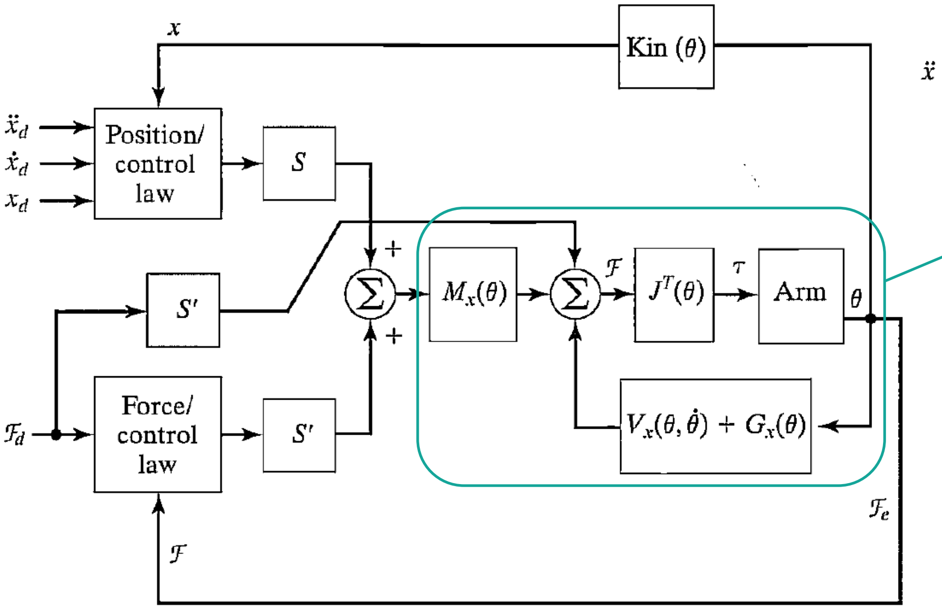


The force-control servo as a black box

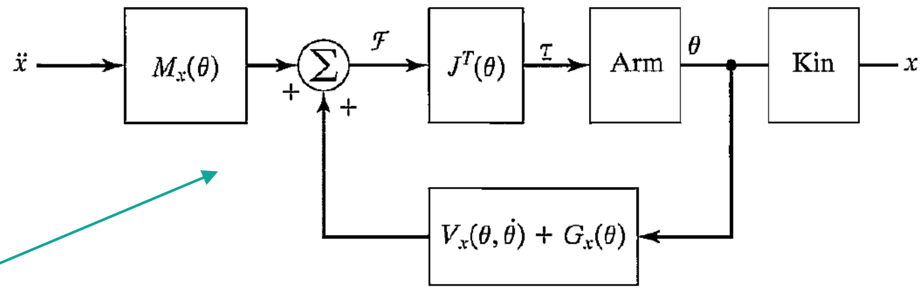
A General Manipulator

An ideal position servo is infinitely stiff and rejects all force disturbances acting on the system.

In contrast, an ideal force servo exhibits zero stiffness and maintains the desired force application regardless of position disturbances.



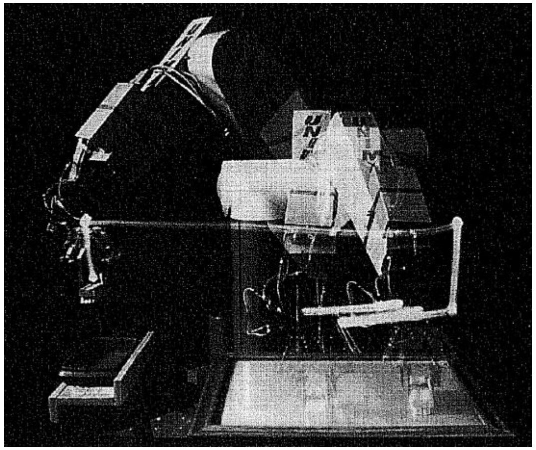
The hybrid position/force controller for a general manipulator



The Cartesian decoupling scheme

Current Industrial Robot Control Schemes

- Passive compliance
- Compliance through softening position gains
- Force sensing



Adding variable stiffness

- Control the end-effector to exhibit stiffnesses other than zero or infinite.
- Control the mechanical impedance of the end-effector

AncoraSIR.com

Thank you!

Prof. Song Chaoyang

- Dr. Wan Fang (sophie.fwan@hotmail.com)

AncoraSIR.com