MEE5114 Advanced Control for Robotics

Lecture 7: Dynamics of Open Chains

Prof. Wei Zhang

SUSTech Insitute of Robotics
Department of Mechanical and Energy Engineering
Southern University of Science and Technology, Shenzhen, China

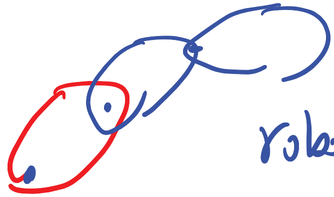
Outline

- Introduction
- Inverse Dynamics: Recursive Newton-Euler Algorithm (RNEA)
- Analytical Form of the Dynamics Model
- Forward Dynamics Algorithms

From Single Rigid Body to Open Chains

- Recall Newton-Euler Equation for a single rigid body:

$$\mathcal{F} = \frac{d}{dt}h = \mathcal{I}\mathcal{A} + \mathcal{V} \times^* \mathcal{I}\mathcal{V}$$



robot : URDF

- Open chains consist of multiple rigid links connected through joints

- Dynamics of adjacent links are coupled.

- We are concerned with modeling multi-body dynamics subject to constraints.

Preview of Open-Chain Dynamics

- Equations of Motion are a set of 2nd-order differential equations:

$$\tau = M(\theta)\ddot{\theta} + \tilde{c}(\theta, \dot{\theta})$$

  - $\theta \in \mathbb{R}^n$: vector of joint variables; $\tau \in \mathbb{R}^n$: vector of joint forces/torques
  - $M(\theta) \in \mathbb{R}^{n \times n}$: mass matrix
  - $\tilde{c}(\theta, \dot{\theta}) \in \mathbb{R}^n$: forces that lump together centripetal, Coriolis, gravity, friction terms, and torques induced by external forces. These terms depend on θ and/or $\dot{\theta}$

- **Forward dynamics:** Determine acceleration $\ddot{\theta}$ given the state $(\theta, \dot{\theta})$ and the joint forces/torques:

$$\ddot{\theta} \leftarrow \mathsf{FD}(\tau, \theta, \dot{\theta}, \mathcal{F}_{ext})$$

- **Inverse dynamics:** Finding torques/forces given state $(\theta, \dot{\theta})$ and desired acceleration $\ddot{\theta}$

$$\tau \leftarrow \mathsf{ID}(\theta, \dot{\theta}, \ddot{\theta}, \mathcal{F}_{ext})$$

Lagrangian vs. Newton-Euler Methods

- There are typically two ways to derive the equation of motion for an open-chain robot: Lagrangian method and Newton-Euler method

Lagrangian Formulation

- Energy-based method
- Dynamic equations in closed form
- Often used for study of dynamic properties and analysis of control methods

Newton-Euler Formulation

- Balance of forces/torques
- Dynamic equations in numeric/recursive form
- Often used for numerical solution of forward/inverse dynamics

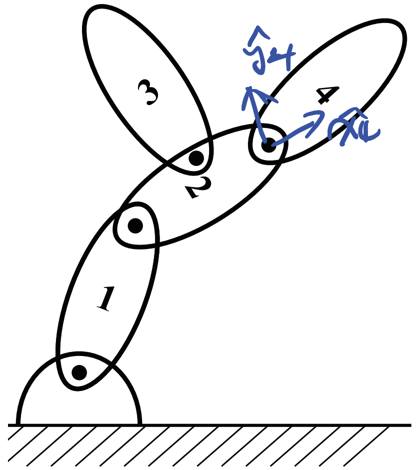
- We focus on Newton-Euler Formulation

Outline

- Introduction
- Inverse Dynamics: Recursive Newton-Euler Algorithm (RNEA)
- Analytical Form of the Dynamics Model
- Forward Dynamics Algorithms

RNEA: Notations

- Number bodies (link): 1 to N
  - Parent: $p(i)$ e.g: $p(4) = 2$, $p(3) = 2$
  - Children: $c(i)$ e.g: $c(2) = \{3, 4\}$, $c(1) = 2$
- Joint i connects $p(i)$ to i



- Frame {i} attached to body i
- $\mathcal{S}_i$: Spatial velocity (screw axis) of joint i : ${}^4\mathcal{S}_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$
- $\mathcal{V}_i$ and $\mathcal{A}_i$: spatial velocity and acceleration of body i
- $\mathcal{F}_i$: force (wrench) onto body i from body $p(i)$, e.g. $\mathcal{F}_2$: force from body 1 to body 2
- Note: By default, all vectors $(\mathcal{S}_i, \mathcal{V}_i, \mathcal{F}_i)$ are expressed in local frame {i}

RNEA: Velocity and Accel. Propagation (Forward Pass)

Goal: Given joint velocity $\dot{\theta}$ and acceleration $\ddot{\theta}$, compute the body spatial velocity $\mathcal{V}_i$ and spatial acceleration $\mathcal{A}_i$

we need to know acceleration bodies as they are related to force

$$\begin{cases} \text{Velocity Propagation:} & {}^i\mathcal{V}_i = \left({}^iX_{p(i)}\right)\ {}^{p(i)}\mathcal{V}_{p(i)} + {}^i\mathcal{S}_i\,\dot{\theta}_i \\ \text{Accel Propagation:} & {}^i\mathcal{A}_i = \left({}^iX_{p(i)}\right)\ {}^{p(i)}\mathcal{A}_{p(i)} + {}^i\mathcal{V}_i \times {}^i\mathcal{S}_i\dot{\theta}_i + {}^i\mathcal{S}_i\ddot{\theta}_i \end{cases}$$

velocity: $\mathcal{V}_1 = S_1\dot{\theta}_1$, $\quad \mathcal{V}_2 = \mathcal{V}_1 + \mathcal{V}_{2/1} = S_1\dot{\theta}_1 + S_2\dot{\theta}_2$

In coordinate: ${}^2\mathcal{V}_2 = {}^2X_1\,{}^1S_1\,\dot{\theta}_1 + {}^2S_2\,\dot{\theta}_2$

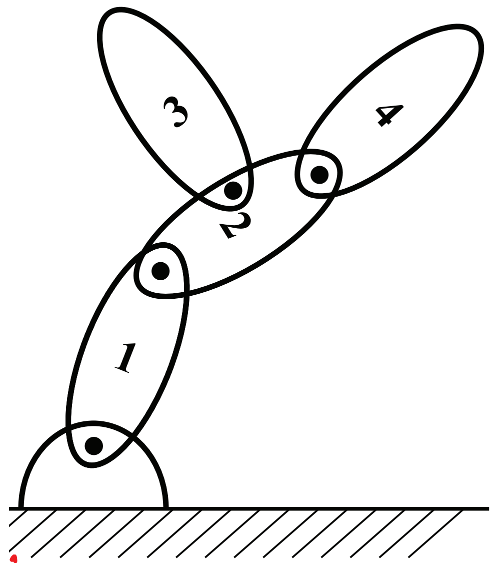
In general: ${}^i\mathcal{V}_i = {}^iX_{p(i)}\,{}^{p(i)}\mathcal{V}_{p(i)} + {}^iS_i\,\dot{\theta}_i$

Accel: $A_2 = A_1 + A_{2/1}$

In coordinate: ${}^2A_2 = {}^2X_1\,{}^1A_1 + {}^2\left[\frac{d}{dt}\Big(\underbrace{S_2\dot{\theta}_2}_{\text{coordinate-free}}\Big)\right]$

$${}^2\left[\frac{d}{dt}\left(S_2\dot{\theta}_2\right)\right] = \underbrace{\left({}^2S_2\ddot{\theta}_2\right)}_{\text{apparent derivative}} + {}^2\mathcal{V}_2 \times \left({}^2S_2\dot{\theta}_2\right)$$

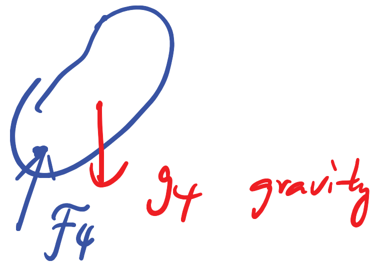
$$\Rightarrow {}^2A_2 = {}^2X_1\,{}^1A_1 + {}^2\mathcal{V}_2 \times \left({}^2S_2\dot{\theta}_2\right) + {}^2S_2\ddot{\theta}_2$$



RNEA: Force Propagation (Backward Pass)

Goal: Given body spatial velocity $\mathcal{V}_i$ and spatial acceleration $\mathcal{A}_i$, compute the joint wrench $\mathcal{F}_i$ and the corresponding torque $\tau_i = \mathcal{S}_i^T \mathcal{F}_i$

$$\begin{cases} \mathcal{F}_i &= \mathcal{I}_i \mathcal{A}_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i + \sum_{j \in c(i)} \mathcal{F}_j \\ \tau_i &= \mathcal{S}_i^T \mathcal{F}_i \end{cases}$$



Body 4: $\mathcal{F}_4 + g_4 = \mathcal{I}_4 \mathcal{A}_4 + \mathcal{V}_4 \times^* \mathcal{I}_4 \mathcal{V}_4$

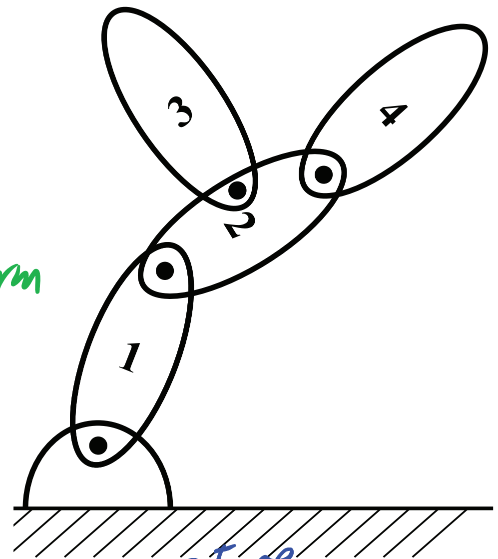
$$\mathcal{F}_4 = \mathcal{I}_4 \mathcal{A}_4 + \mathcal{V}_4 \times^* \mathcal{I}_4 \mathcal{V}_4 - \underline{g_4}$$

note: $g_4 = \mathcal{I}_4 \cdot {}^4\mathcal{A}_g = \mathcal{I}_4 \, {}^4X_0 \, {}^0\mathcal{A}_g$

$$\mathcal{F}_4 = \mathcal{I}_4 \mathcal{A}_4 + \mathcal{V}_4 \times^* \mathcal{I}_4 \mathcal{V}_4$$

← assume include gravity term

$$\tau_4 = \underline{S_4^T} \, \underline{\mathcal{F}_4}$$



Body 2: $\mathcal{F}_2 - \mathcal{F}_3 - \mathcal{F}_4 + g_2 = \mathcal{I}_2 \mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2 \mathcal{V}_2$

$\Rightarrow \mathcal{F}_2 = \mathcal{I}_2 \mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2 \mathcal{V}_2 + \mathcal{F}_3 + \mathcal{F}_4 - g_2 \Rightarrow \tau_2 = S_2^T \mathcal{F}_2$

Recursive Newton-Euler Algorithm

ID

$$\tau \leftarrow \text{RNEA}(\theta, \dot{\theta}, \ddot{\theta}, \mathcal{F}_{ext}; \text{Model}) \in O(N)$$

initialize: $\mathcal{V}_0 = 0$, $\mathcal{A}_0 = -\mathcal{A}_g$

e.g. If $\mathcal{F}_{ext}$ is the force body i exerts on the environment

- Forward pass:

For $i = 1$ to N

$\mathcal{V}_i = {}^{i}X_{p(i)} \mathcal{V}_{p(i)} + S_i \dot{\theta}_i$

$\mathcal{A}_i = {}^{i}X_{p(i)} \mathcal{A}_{p(i)} + S_i \ddot{\theta}_i + \mathcal{V}_i \times S_i \dot{\theta}_i$

$\mathcal{F}_i = \mathcal{I}_i \mathcal{A}_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i$

at N step $\mathcal{F}_i = \mathcal{I}_i \mathcal{A}_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i + \mathcal{F}_{ext}$

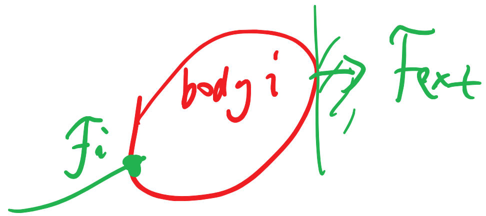
- Backward pass:

For $i = N$ to 1

$\underline{\tau_i = S_i^T \mathcal{F}_i}$

$\mathcal{F}_{p(i)} = \mathcal{F}_{p(i)} + {}^{p(i)}X_i^* \mathcal{F}_i$

END



Without $\mathcal{A}_0 = -\mathcal{A}_g$ trick: modify: $\mathcal{F}_i = \mathcal{I}_i \mathcal{A}_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i - \mathcal{I}_i \, {}^{i}X_0 \mathcal{A}_g$

Outline

- Introduction
- Inverse Dynamics: Recursive Newton-Euler Algorithm (RNEA)
- Analytical Form of the Dynamics Model
- Forward Dynamics Algorithms

Structures in Dynamic Equation (1/3)

- Jacobian of each link (body): $J_1, \dots, J_4$

J_i : denote the Jacobian of body i, i.e. $\mathcal{V}_i = J_i \dot{\theta}$

$$= \begin{bmatrix} J_{i,1} & J_{i,2}, & J_{i,3}, & J_{i,4} \end{bmatrix} \begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \\ \dot\theta_4 \end{bmatrix}$$

e.g.: $\mathcal{V}_1 = J_1 \begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \\ \dot\theta_4 \end{bmatrix} = S_1 \dot\theta_1 = \begin{bmatrix} \delta_{11} S_1 & \delta_{12} S_2 & \delta_{13} S_3 & \delta_{14} S_4 \end{bmatrix} \begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \\ \dot\theta_4 \end{bmatrix}$

$$\delta_{ij} = \begin{cases} 1, & \text{if joint } j \text{ support body } i \\ 0, & \text{otherwise} \end{cases}$$

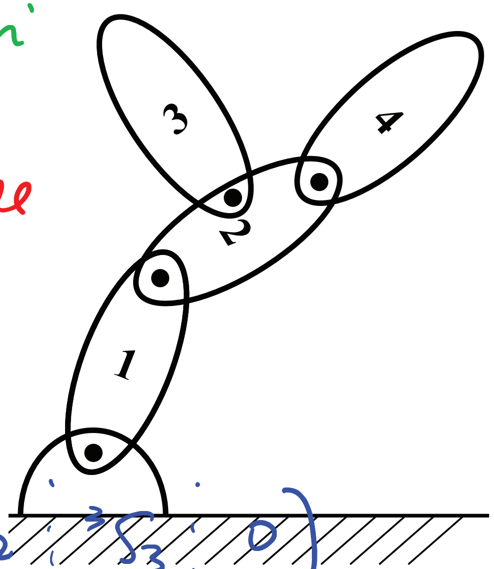
$\mathcal{V}_2 = \begin{bmatrix} \underbrace{\delta_{21}}_{1} S_1 & \underbrace{\delta_{22}}_{1} S_2 & \underbrace{\cancel{\delta_{23}}}_{0} S_3 & \underbrace{\cancel{\delta_{24}}}_{0} S_4 \end{bmatrix} \dot\theta$ ← coordinate free

If $\mathcal{V}_2 = {}^2\mathcal{V}_2$ (using local coordinate)

$${}^2\mathcal{V}_2 = \underbrace{\begin{bmatrix} {}^2X_1 {}^1S_1 & {}^2S_2 & 0 & 0 \end{bmatrix}}_{{}^2J_2} \dot\theta$$

$${}^3J_3 = \begin{bmatrix} {}^3X_1 {}^1S_1 & {}^3X_2 {}^2S_2 & {}^3S_3 & 0 \end{bmatrix}$$

$${}^4J_4 = \begin{bmatrix} {}^4X_1 {}^1S_1 & {}^4X_2 {}^2S_2 & 0 & {}^4S_4 \end{bmatrix}$$



Structures in Dynamic Equation (2/3)

- Torque required to generate a "force" $\mathcal{F}_4$ to body 4 with velocity $\mathcal{V}_4$

For simplicity, let's not worry about gravity for now

Work done to body 4 over $[0, T]$: $\int_0^T \mathcal{V}_4^T \mathcal{F}_4 \, dt = \int \tau^T \dot{\theta} \, dt$

$$= \tau_1 \dot{\theta}_1 + \tau_2 \dot{\theta}_2 + \tau_4 \dot{\theta}_4$$

$$\mathcal{V}_4 = J_4 \dot{\theta}$$

$$\Rightarrow \int_0^T \dot{\theta}^T J_4^T \mathcal{F}_4 \, dt = \int_0^T \dot{\theta}^T \tau \, dt$$

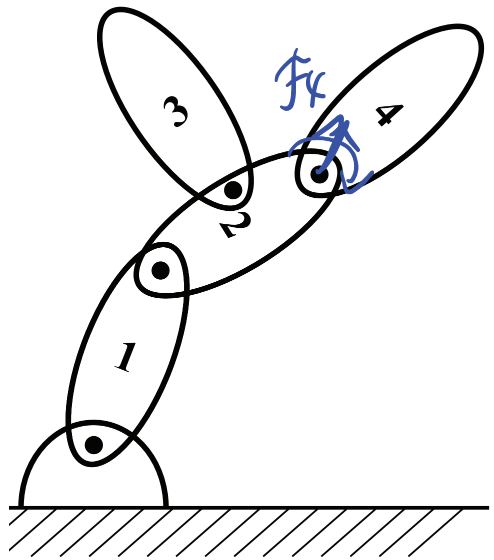
$$\Rightarrow \boxed{\tau = J_4^T \mathcal{F}_4} \rightarrow [6\times 1]$$

$$\begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix} \quad \begin{bmatrix} {}^1S_1^T \, {}^4X_1^T \\ {}^2S_2^T \, {}^4X_2^T \\ 0 \\ S_4^T \end{bmatrix}$$

$$\Rightarrow \tau_1 = J_{4,1}^T \mathcal{F}_4$$

$$= ({}^4X_1 \, {}^1S_1)^T \mathcal{F}_4$$

$$\tau_2 = (J_{4,2})^T \mathcal{F}_4$$



Structures in Dynamic Equation (3/3)

- Overall torque expression:

See example below: Let's use RNEA:

① Forward pass: $\mathcal{V}_1 = S_1\dot{\theta}_1$, $\mathcal{V}_2 = {}^2X_1 S_1\dot{\theta}_1 + S_2\dot{\theta}_2 = \underbrace{[{}^2X_1 S_1 \quad S_2]}_{J_2}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$

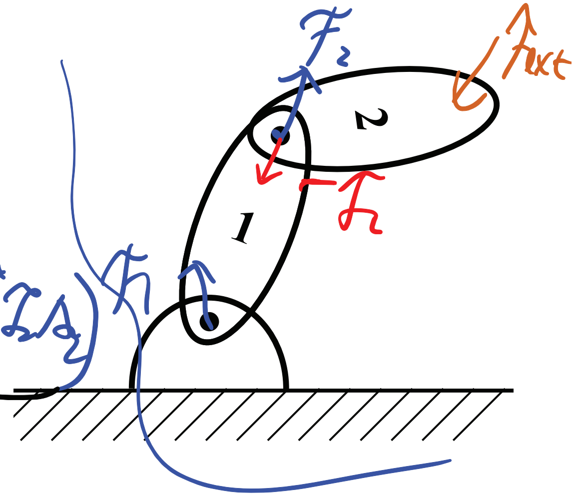
$\mathcal{A}_1 \cdots$ $\quad \mathcal{A}_2 \cdots$

② Backward pass: $\mathcal{F}_2 = \underline{\mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{A}_2}$

$\mathcal{F}_1 = \mathcal{I}_1\mathcal{A}_1 + \mathcal{V}_1 \times^* \mathcal{I}_1\mathcal{A}_1 + {}^1X_2^* \mathcal{F}_2$

(if consider gravity $\mathcal{A}_2 \Rightarrow \mathcal{A}_2 - {}^2X_0\, {}^0\mathcal{A}_g$)

$\tau_2 = S_2^T \mathcal{F}_2 = S_2^T(\mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{A}_2)$



$\tau_1 = S_1^T \mathcal{F}_1 = \underbrace{S_1^T(\mathcal{I}_1\mathcal{A}_1 + \mathcal{V}_1 \times^* \mathcal{I}_1\mathcal{A}_1)}_{\text{torque @ Joint 1 due to motion of body 1}} + \underbrace{S_1^T\, {}^2X_1^T(\mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{A}_2)}_{\text{torque @ Joint 1 due to motion of body 2}}$

If conside $\mathcal{F}_{ext}$ $\quad \tau_1 = S_1^T(\mathcal{I}_1 \cdots) + S_1^T\, {}^2X_1^T(\mathcal{I}_2\mathcal{A}_2 + \cdots) + S_1^T({}^1X_2^* \mathcal{F}_{ext})$

Derivation of Overall Dynamics Equation

- $\tau = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} S_1^T (\mathcal{I}_1 \mathcal{A}_1 + \cdots) + S_1^T \, {}^2X_1^T (\mathcal{I}_2 \mathcal{A}_2 + \cdots) \\ 0 \, (\mathcal{I}_1 \mathcal{A}_1 + \cdots) + S_2^T (\mathcal{I}_2 \mathcal{A}_2 + \cdots) \end{bmatrix}$

$J_1^T \rightarrow {}^1J_1^T \qquad J_2^T - {}^2J_2^T$

$$\tau = \sum_{i=1}^{2} J_i^T (\mathcal{I}_i \mathcal{A}_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i)$$

In general, suppose we have N body chain,

$$\tau = \sum_{i=1}^{N} J_i^T (\underbrace{\mathcal{I}_i \mathcal{A}_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i}) + \text{some external force induced torque}$$

$\rightarrow - \mathcal{I}_i \, {}^iX_0 \, {}^0\mathcal{A}_g$

Note: $\mathcal{V}_i = J_i \dot{\theta}$ → body i Jacobian

$$\tau = M(\theta)\ddot{\theta} + c(\theta, \dot{\theta})\dot{\theta} + \tau_g + J^T(\theta)\mathcal{F}_{ext} \tag{1}$$

$$\mathcal{A}_i = (\mathcal{V}_i)' = J_i \ddot{\theta} + \left(\frac{d}{dt} J_i\right) \dot{\theta} = J_i \ddot{\theta} + (\dot{J}_i \dot{\theta} + \mathcal{V}_i \times J_i \dot{\theta})$$

$$\Rightarrow \tau = \sum_{i=1}^{N} J_i^T \mathcal{I}_i J_i \ddot{\theta} + J_i^T \mathcal{I}_i \dot{J}_i \dot{\theta} + J_i^T \mathcal{I}_i V_i \times J_i \dot{\theta} + J_i^T V_i \times^* \mathcal{I}_i \underbrace{V_i}_{J_i \dot{\theta}}$$

$$= \underbrace{\left(\sum_{i=1}^{N} J_i^T \mathcal{I}_i J_i \right)}_{\triangleq M(\theta)} \ddot{\theta} + \underbrace{\sum_{i=1}^{N} J_i^T \left(\mathcal{I}_i \dot{J}_i + \mathcal{I}_i V_i \times J_i + V_i \times^* \mathcal{I}_i J_i \right) \dot{\theta}}_{\triangleq C(\theta, \dot{\theta})}$$

$N \times N$

mass matrix

coriolis term

If consider gravity

$$+ \sum_{i=1}^{N} J_i^T \mathcal{I}_i \, {}^{i}X_0 \left(-A_g \right)$$

generalized gravity

Properties of Dynamics Model of Multi-body Systems

- $-$ $M(\theta)$: mass matrix, $M(\theta) = M(\theta)^T$, $M(\theta)$ positive definite if $\mathcal{I}_i > 0$

$-$ There are many other definitions of $C(\theta, \dot\theta)$, i.e., all of them give the same product $C(\theta,\dot\theta)\dot\theta$

e.g.: $$C(\theta,\dot\theta)\dot\theta = \begin{bmatrix} -2\dot\theta_2\dot\theta_1 \\ \dot\theta_1^2 \end{bmatrix} = \overbrace{\begin{bmatrix} -2\dot\theta_2 & 0 \\ \dot\theta_1 & 0 \end{bmatrix}}^{C(\theta,\dot\theta)} \begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -2\dot\theta_1 \\ \dot\theta_1 & 0 \end{bmatrix} \begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \end{bmatrix}$$

$-$ Typical expression for C: $$[C]_{ij} = \sum_k \frac{1}{2}\underbrace{\left(\frac{\partial M_{ij}}{\partial \theta_k} + \frac{\partial M_{ik}}{\partial \theta_j} - \frac{\partial M_{jk}}{\partial \theta_i}\right)}_{\Gamma_{ijk}\text{ : christoffel}}\dot\theta_k$$

$-$ $C(\theta,\dot\theta)$ defined above satisfies, $[M - 2C]$ is skew symmetric

Outline

If S is skew symmetric $\Rightarrow$ $\underline{S = -S^T}$

$$\Rightarrow \underbrace{x^T S x} = x^T S^T x = -x^T S x = 0$$

$$S = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad x^T S x = x_1 x_2 - x_1 x_2 = 0$$

- Introduction
- Inverse Dynamics: Recursive Newton-Euler Algorithm (RNEA)

– $M(\theta)$, $c(\theta, \dot\theta)$, τ_g all depend on $\mathcal{I}_i$ linearly.

- Analytical Form of the Dynamics Model
- **Forward Dynamics Algorithms**

Forward Dynamics Problem

$$\tau = M(\theta)\ddot{\theta} + c(\theta,\dot{\theta})\dot{\theta} + \tau_g + J^T(\theta)\mathcal{F}_{ext} \tag{2}$$

- Inverse dynamics: $\tau \leftarrow \text{RNEA}(\theta, \dot{\theta}, \ddot{\theta}, \mathcal{F}_{ext})$ $O(N)$ complexity
  - RNEA can work directly with a given URDF model (kinematic tree + joint model + dynamic parameters). It does not require explicit formula for $M(\theta), \tilde{c}(\theta, \dot{\theta})$

- **Forward dynamics:** Given $(\theta, \dot{\theta})$, τ, $\mathcal{F}_{ext}$, find $\ddot{\theta}$
  1. Calculate $\tilde{c}(\theta, \dot{\theta}) \triangleq c(\theta,\dot{\theta})\dot{\theta} + \tau_g + J^T(\theta) \mathcal{F}_{ext}$
  2. Calculate mass matrix $M(\theta)$
  3. Solve $M\ddot{\theta} = \tau - \tilde{c}$ $\Rightarrow$ $\underbrace{\ddot{\theta} = M^{-1}}_{\text{not efficient}}(\tau - \tilde{c})$

Dynamics (2) is nonlinear ODE

State space form:

$x = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$, $\dot{x} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = f(x_1, x_2, \tau)$

$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ $\begin{matrix} \in \mathbb{R}^n \\ \in \mathbb{R}^n \end{matrix}$

Calculations of $\tilde{c}$ and M

- Denote our inverse dynamics algorithm: $\tau = \text{RNEA}(\theta, \dot{\theta}, \ddot{\theta}, \mathcal{F}_{ext}) = M\ddot{\theta} + \tilde{c}$

- **Calculation of $\tilde{c}$**: obviously, $\tau = \tilde{c}(\theta, \dot{\theta})$ if $\ddot{\theta} = 0$. Therefore, $\tilde{c}$ can be computed via:
$$\tilde{c}(\theta, \dot{\theta}) = \text{RNEA}(\theta, \dot{\theta}, 0, \mathcal{F}_{ext})$$

- **Calculation of M:** Note that $\tilde{c}(\theta, \dot{\theta}) = c(\theta, \dot{\theta})\dot{\theta} + \tau_g + J^T(\theta)\mathcal{F}_{ext}$.
  - Set $\mathrm{g} = 0$, $\mathcal{F}_{ext} = 0$, and $\dot{\theta} = 0$, then $\tilde{c}(\theta, \dot{\theta}) = 0 \Rightarrow \tau = M(\theta)\ddot{\theta}$, if $\ddot{\theta} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$
    $\Rightarrow \tau = M_1(\theta)$
  - We can compute the jth column of $M(\theta)$ by calling the inverse algorithm
$$M_{:,j}(\theta) = \text{RNEA}(\theta, 0, \ddot{\theta}_j^0, 0)$$
$\ddot{\theta}_1^0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, $\ddot{\theta}_2^0 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$

    where $\ddot{\theta}_j^0$ is a vector with all zeros except for a 1 at the jth entry.

- A more efficient algorithm for computing M is the *Composite-Rigid-Body Algorithm (CRBA)*. Details can be found in Featherstone's book.

Forward Dynamics Algorithm

- Now assume we have $\theta, \dot{\theta}, \tau, M(\theta), \tilde{c}(\theta, \dot{\theta})$, then we can immediately compute $\ddot{\theta}$ as $\ddot{\theta} = M^{-1}(\theta)\left[\tau - \tilde{c}(\theta, \dot{\theta})\right]$

- This provides a 2nd-order differential equation in $\mathbb{R}^n$, we can easily simulate the joint trajectory over any time period (under given ICs θ^o and $\dot{\theta}^o$)

- Computational Complexity:
  - RNEA: $O(N)$
  - $\tilde{c} = RNEA(\theta, \dot{\theta}, 0, \mathcal{F}_{ext})$: $O(N)$
  - $M(\theta)$: $O(N^2)$
  - $M^{-1}(\theta)$: $O(N^3)$
  - Most efficient forward dynamics algorithm:
    Articulated-Body Algorithm (ABA): $O(N)$

More Discussions

-

More Discussions

-