MEE5114 Advanced Control for Robotics

Lecture 9: Dynamics of Open Chains

Prof. Wei Zhang

CLEAR Lab

Department of Mechanical and Energy Engineering

Southern University of Science and Technology, Shenzhen, China

https://www.wzhanglab.site/

- Spatial Accelleration: $A \in \mathbb{R}^6$

$A_{body} \triangleq \dot{\mathcal{V}}_{body}$ (coordinate free)

working with inertial/stationary frame

$${}^{0}A_{body} = \frac{d}{dt}({}^{0}\mathcal{V}_{body})$$

$\hookrightarrow {}^{0}\dot{\mathcal{V}}_{body}$

apparent derivative

working with moving frame:

$${}^{B}A_{body} = \frac{d}{dt}({}^{B}\mathcal{V}_{body})$$

$$= \frac{d}{dt}({}^{B}\mathcal{V}_{body}) + {}^{B}\mathcal{V}_{B} \times {}^{B}\mathcal{V}_{body}$$

$[{}^{B}\mathcal{V}_{B}\times]$ 6×6

Outline

${}^B\mathcal{A}_{body} = {}^BX_0 \, {}^0\mathcal{A}_{body}$

Rotation version — $\dot{R}_A = \omega_A \times R_A$, $[R\omega] = R[\omega]R^T$

Twist version — ${}^0\dot{X}_A = \mathcal{V}_A \times {}^0X_A = [\mathcal{V}_A \times]_{6\times 6} \, {}^0X_A$, $[X\mathcal{V}\times] = X[\mathcal{V}\times]X^T$

- Introduction

— Spatial force (wrench)

$${}^B\mathcal{F} = \begin{bmatrix} n_{OB} \\ {}^Bf \end{bmatrix}, \quad {}^A\mathcal{F} = {}^AX_B^* \, {}^B\mathcal{F} \qquad {}^AX_B^* = ({}^BX_A)^T$$

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

$${}^0\dot{X}_A^* = [\mathcal{V}_A \times^*] \, {}^0X_A^*$$

- Analytical Form of the Dynamics Model

Spatial Momentum:

$${}^Ah = \begin{bmatrix} {}^A\phi_{OA} \\ {}^AL \end{bmatrix}, \quad {}^Ah = {}^AX_B^* \, {}^Bh$$

- Forward Dynamics Algorithms

Spatial inertial

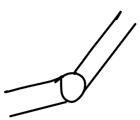
$${}^C\mathcal{I} = \begin{bmatrix} {}^C\bar{I} & 0 \\ 0 & m I_{3\times 3} \end{bmatrix}$$

$${}^A\mathcal{I} = ({}^AX_C^*) \, {}^C\mathcal{I} \, ({}^CX_A)$$

— Joint Torque

$$\tau\dot{\theta} = \mathcal{V}^T\mathcal{F} = S^T\dot{\theta}\mathcal{F}$$

$$\tau = S^T\mathcal{F} = \mathcal{F}^T S$$



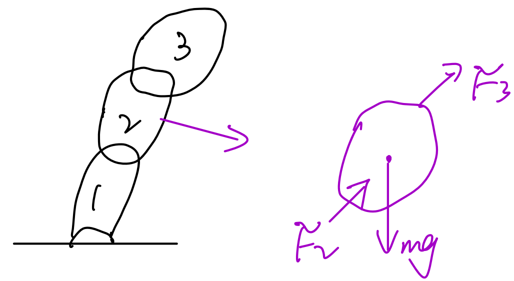
From Single Rigid Body to Open Chains

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

$$\mathcal{F} = \underbrace{\frac{d}{dt}h = \mathcal{I}\mathcal{A} + \mathcal{V} \times^* \mathcal{I}\mathcal{V}}_{\text{coordinate free}}$$

- Open chains consist of multiple rigid links connected through joints

bodies



- Dynamics of adjacent links are coupled.

- This lecture: model multi-body dynamics subject to joint constraints.

Preview of Open-Chain Dynamics

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

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

$$+ C(\theta,\dot{\theta}) + g(\theta) + J^T \mathcal{F}_{ext} + \cdots$$

  - $\theta \in \mathbb{R}^n$: 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:

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

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

Given desired motion $(\theta, \dot{\theta}, \ddot{\theta})$

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

Find the required torque to generate this desired motion

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 $\boxed{\text{Newton-Euler Formulation}}$

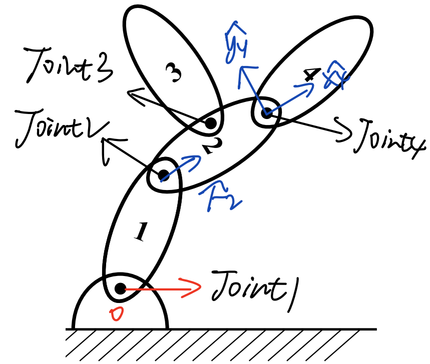
Featherstone's book

Wensing's book

Outline

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

RNEA: Notations



- Number bodies: 1 to N
  - Parent: $p(i)$ e.g. $p(3)=2$, $p(4)=2$
  - 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 at the joint

  frame {4} moves with the body {4}
- $\mathcal{S}_i$: Spatial velocity (screw axis) of joint i e.g. ${}^4S_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ constant
- $\mathcal{V}_i$ and $\mathcal{A}_i$: spatial velocity and acceleration of body i
- $\mathcal{F}_i$: force (wrench) onto body i from body $p(i)$

  F_2 from $p(2)$ 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 $\mathcal{A}_i$

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

$\tau = ID(\theta, \dot\theta, \ddot\theta, F_{ext})$ Velocity: $V_1 = S_1\dot\theta_1$, $V_2 = V_1 + V_{2/1} = S_1\dot\theta_1 + S_2\dot\theta_2$

work with local coordinates ${}^1V_1 = {}^1S_1\dot\theta_1$

${}^2V_2 = {}^2X_1\,{}^1S_1\dot\theta_1 + {}^2S_2\dot\theta_2$

In general ${}^iV_i = {}^iX_{p(i)}\,{}^{p(i)}V_{p(i)} + {}^iS_i\dot\theta_i$

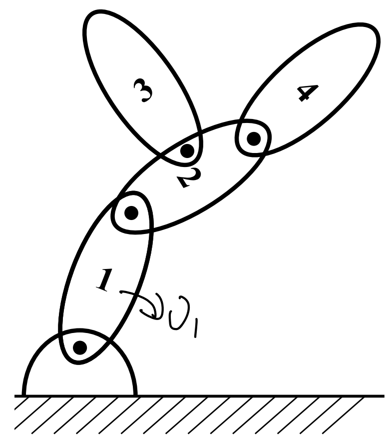
Acceleration: $A_2 = \dot V_2 = \dot V_1 + \dot V_{2/1} = A_1 + A_{2/1} =$

In coordinate ${}^2A_2 = {}^2X_1\,{}^1A_1 + {}^2\left[\frac{d}{dt}(S_2\dot\theta)\right]$

Coordinate free notation

$${}^2\underbrace{\left[\frac{d}{dt}(S_2\dot\theta_2)\right]}_{V_2} = \frac{d}{dt}(\underbrace{{}^2S_2\dot\theta_2}_{{}^2V_2}) + {}^2V_2 \times {}^2S_2\dot\theta_2 = {}^2S_2\ddot\theta_2 + {}^2V_2 \times {}^2S_2\dot\theta_2$$

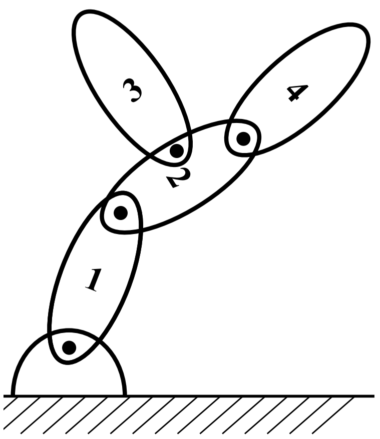
${}^2A_2 = {}^2X_1\,{}^1A_1 + {}^2V_2 \times {}^2S_2\dot\theta_2 + {}^2S_2\ddot\theta_2$



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 $\mathcal{A}_i$

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



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: $F_4 + F_{g4} = I_4 A_4 + V_4 \times^* I_4 V_4$

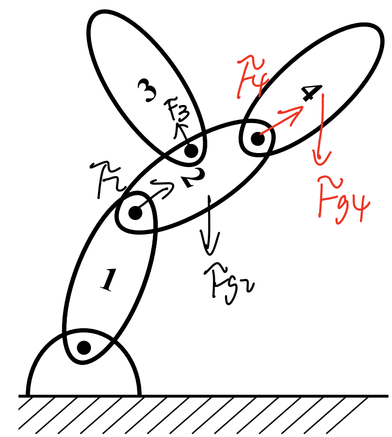
$$F_4 = I_4 A_4 + V_4 \times^* I_4 V_4 - F_{g4}$$

Note: $F_{g4} = I_4 \, {}^4A_g = I_4 \, {}^4X_0 \, {}^0A_g$

$$\tau_4 = S_4^T \tilde{F}_4$$

Body 2: $F_2 = I_2 A_2 + V_2 \times^* I_2 V_2 + (\tilde{F}_3 + \tilde{F}_4 - \underbrace{\tilde{F}_{g2}}_{I_2 \, {}^2X_0 \, {}^0A_g})$

$$\tau_2 = S_2^T F_2$$



Recursive Newton-Euler Algorithm

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

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

Without gravity trick

modify: ① to

$F_i = \mathcal{I}_i A_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i - \mathcal{I}_i \cdot \underbrace{{}^iX_0\, {}^0\mathcal{A}_g}_{{}^i\mathcal{A}_g}$

- Forward pass:

For $i = 1$ to N

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

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

$F_i = \mathcal{I}_i A_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i$

wrench due to i^{th} body

合力

- Backward pass:

For $i = N:-1:1$

$\tau_i = S_i^T F_i$

$F_{p(i)} = F_{p(i)} + {}^{p(i)}X_i^* F_i$

End

扣除前面link力

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): $J_1, \dots, J_4$

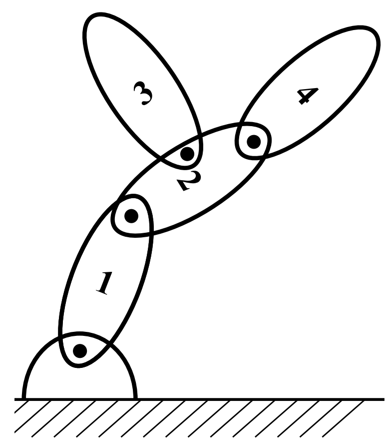
J_i: denote the Jacobian of body (Link) i, i.e. $\mathcal{V}_i = J_i \dot{\theta} = [J_{i1} \ J_{i2} \ \cdots \ J_{i4}] \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}$

e.g. $\mathcal{V}_1 = J_1 \dot{\theta} = [\delta_{11} S_1 \quad \delta_{12} S_2 \quad \delta_{13} S_3 \quad \delta_{14} S_4] \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix} = [S_1, 0, 0, 0] \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_4 \end{bmatrix}$

$$\mathcal{V}_2 = J_2 \dot{\theta} = [S_1 \quad S_2 \quad 0 \quad 0] \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_4 \end{bmatrix}$$

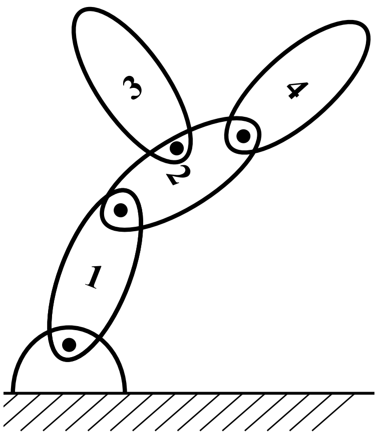
In $\{2\}$: ${}^2\mathcal{V}_2 = \underbrace{[{}^2X_1 S_1 ; \ {}^2S_2 ; \ 0 \quad 0]}_{{}^2J_2} \dot{\theta}$

$${}^4J_4 = [{}^4X_1 \, {}^1S_1 \quad {}^4X_2 \, {}^2S_2 \quad {}^4X_3 \, {}^3S_3 \quad S_4]$$



Structures in Dynamic Equation (2/3)

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



Structures in Dynamic Equation (3/3)

- Overall torque expression:

See the two-body example

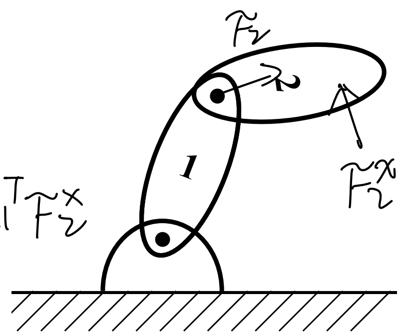
① Forward pass $v_1 = S_1\dot{\theta}_1$, $v_2 = \left[{}^2X_1 S_1 \;\vdots\; S_2\right]\begin{bmatrix}\dot{\theta}_1 \\ \dot{\theta}_2\end{bmatrix}$

$A_1, A_2 = \cdots$

$\tilde{F}_2$ $\tilde{F}_2^x$ 同号

② Backward pass, $\tilde{F}_2 = I_2A_2 + v_2\times^* I_2 v_2 - \tilde{F}_2^x$

$$\begin{aligned}
\tilde{F}_1 &= I_1A_1 + v_1\times^* I_1 v_1 + \tilde{F}_2 \\
&= I_1A_1 + v_1\times^* I_1 v_1 + {}^1X_2^* \tilde{F}_2 \\
&= I_1A_1 + v_1\times^* I_1 v_1 + {}^2X_1^T (I_2 v_2 + v_2\times^* I_2 v_2) - {}^2X_1^T \tilde{F}_2^x
\end{aligned}$$



$$\tau_2 = S_2^T\tilde{F}_2 = S_2^T(I_2A_2 + \cdots) - S_2^T\tilde{F}_2^x$$

$$\tau_1 = S_1^T\tilde{F}_1 = \underbrace{S_1^T(I_1A_1 + \cdots)}_{①} + \underbrace{({}^2X_1S_1)^T(I_2 v_2 + v_2\times^* I_2 v_2)}_{②} \underbrace{\boxed{-({}^2X_1S_1)^T\tilde{F}_2^x}}_{③}$$

Structures in Dynamic Equation (3/3)

- Overall torque expression: ① $S_1^T(I_1A_1+v_1\times^* I_1 v_1)$

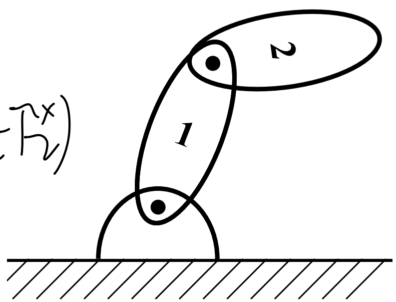
torque @ joint1 due to motion of body 1

② torque at joint1 due to motion of body 2

③ torque ~ ~ ~ external force of body 2

$$\tau=\begin{bmatrix}\tau_1\\ \tau_2\end{bmatrix}=\begin{bmatrix} S_1^T(I_1A_1+\cdots)+({}^2X_1S_1)^T(I_2v_2+\cdots)+({}^2X_1S_1)^T(-F_2^x)\\ 0(I_1A_1+\cdots)+S_2^T(I_2v_2+\cdots)+S_2^T(-F_2^x)\end{bmatrix}$$

$$=\begin{bmatrix}S_1^T\\ 0\end{bmatrix}(I_1A_1+\cdots)+\begin{bmatrix}({}^2X_1S_1)^T\\ S_2^T\end{bmatrix}(I_2A_2+\cdots)+\begin{bmatrix}({}^2X_1S_1)^T\\ S_2^T\end{bmatrix}(-F_2^x)$$



$$={}^1J_1^T(I_1A_1+\cdots)+{}^2J_2^T(I_2A_2+\cdots)+{}^2J_2^T(-F_2^x)$$

Derivation of Overall Dynamics Equation

- Overall: in several with n-Links/Joints

$$\tau = \sum_{i=1}^{N}\left\{ J_i^T(\mathcal{I}_i A_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i) + J_i^T \underbrace{(\text{external force terms})}_{\text{e.g. gravity or other external forces}} \right\}$$

$J_i = {}^iJ_i$

$\mathcal{V}_i = J_i\dot\theta$
↳ body i Jacobian

$$A_i = \dot{\mathcal{V}}_i = \underbrace{\boxed{J_i\ddot\theta + \dot J_i\dot\theta}}_{\text{apparent derivate}} + \underbrace{\mathcal{V}_i \times J_i\dot\theta}$$

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

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

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

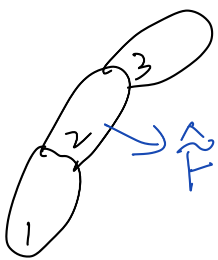
If consider gravity, we need to add $\sum_{i=1}^{N} J_i^T \mathcal{I}_i\, {}^iX_0(-{}^0A_g)$ τ_g

Properties of Dynamics Model of Multi-body Systems

- J_i: body/link i Jacobian $\underset{6\times 1}{\mathcal{V}_i} = \underset{6\times n}{J_i}\,\dot{\theta}_i \rightarrow \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{bmatrix}$

- $\tau = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} \in \mathbb{R}^n$, τ plays two major roles

$$\begin{cases} \text{① to generate motion} \\ \text{② to generate force/torque} \end{cases}$$



$\hat{\mathcal{F}}$ (wrech from body 2 to the environment)

- Only consider body 2's effect

$$\tau = J_2^T(\mathcal{I}_2 \mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2 \mathcal{V}_2) + J_2^T \hat{\mathcal{F}}$$

If consider gravity, we also add terms like $J_2^T(-\mathcal{I}_2\, {}^2X_0\, {}^0\mathcal{A}_g)$

Properties of Dynamics Model of Multi-body Systems

- − If consider all the bodies

  τ = all motions + all forces

  $$= \sum_{i=1}^{n} J_i^T (I_i A_i + V_i \times^* I_i V_i) + J_i^T (-I_i\, {}^i X_0\, {}^0 A_g)$$

Outline

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

$$\underbrace{\tau}_{N\times 1} \quad \left(\underbrace{M}_{N\times N} \; \underbrace{\ddot{\theta}}_{N\times 1}\right) \left(\underbrace{C}_{N\times N} \; \underbrace{\dot{\theta}}_{N\times 1}\right) \; \underbrace{\tau_g}_{N\times 1} \quad \left(\underbrace{J^T(\theta)}_{N} \; F_{ext}\right)$$

Forward Dynamics Problem

$\curvearrowright \tilde{c}(\theta, \dot{\theta})$

applied from body to the environment

given ?

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

- Inverse dynamics: $\tau \leftarrow$ 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 $\tilde{c}(\theta, \dot{\theta}) = c(\theta, \dot{\theta})\dot{\theta} + \tau_g + J^T(\theta) \mathcal{F}_{ext}$
  2. Calculate mass matrix $M(\theta)$
  3. Solve $M\ddot{\theta} = \tau - \tilde{c}$ $\Rightarrow \ddot{\theta} = M^{-1}(\tau - \tilde{c})$

     this is not the most efficient way to find $\ddot{\theta}$

Calculations of $\tilde{c}$ and M

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

  Input

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

  $\tau = \tilde{c}$, when $\ddot{\theta} = 0$

$$\tilde{c}(\theta, \dot{\theta}) = \text{RNEA}(\theta, \dot{\theta}, 0, \mathcal{F}_{ext}) = C(\theta, \dot{\theta})\dot{\theta} + \tau_g + J^T F_{ext}$$

- **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 $\underbrace{g = 0}_{\tau_g = 0}$, $\underbrace{\mathcal{F}_{ext} = 0}_{J^T F_{ext} = 0}$, and $\underbrace{\dot{\theta} = 0}_{C(\theta, \dot{\theta})\dot{\theta} = 0}$, then $\tilde{c}(\theta, \dot{\theta}) = 0 \Rightarrow \boxed{\tau = M(\theta)\ddot{\theta}} + 0$

    if $\ddot{\theta} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

    $\tau = [M_1(\theta)\ M_2(\theta) \cdots] \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \vdots \end{bmatrix} = M_1(\theta)$
  - We can compute the jth column of $M(\theta)$ by calling the inverse algorithm

$$M_{:,j}(\theta) = \text{RNEA}(\theta, 0, \ddot{\theta}_j^0, 0)$$

$$\ddot{\theta}_j^0 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \to j\text{th element}$$

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

$$\ddot{\theta}_1^0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix} \quad \ddot{\theta}_2^0 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- 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 $\ddot{\theta} = M^{-1}(\theta)\left[\tau - \tilde{c}(\theta, \dot{\theta})\right]$ $\quad \ddot{\theta} = FD(\tau, \theta, \dot{\theta}, \mathcal{F}_{ext})$

- 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 $\dot{\theta}^o$)

$x_1 = \theta, \quad x_2 = \dot{\theta} \quad \in \mathbb{R}^n$

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} x_2 \\ M^{-1}(x_1)\left[\tau - \tilde{c}(x_1, x_2)\right] \end{bmatrix} = f(x)$$

- 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

Inertial matrix symmetric / positive semidefinite

- $\tau = \left(\sum_{i}^{N} (J_i^T \mathcal{I}_i J_i)\right) \ddot{\theta} + \sum_{i=1}^{N} J_i^T (\mathcal{I}_i \dot{J}_i + \mathcal{I}_i V_i \times J_i + V_i \times^* \mathcal{I}_i J_i) \dot{\theta}$

- $M(\theta)$: mass matrix $\quad M(\theta)^T = M(\theta)$, $M(\theta)$ is also positive semi-definite

- There are many equivalent ways to define $C(\theta,\dot{\theta})$, they all lead to the same product $C(\theta,\dot{\theta})\dot{\theta}$

e.g. $\underbrace{C(\theta,\dot{\theta})\dot{\theta}} = \begin{bmatrix} -2\dot{\theta}_2\dot{\theta}_1 \\ \dot{\theta}_1^2 \end{bmatrix} = \underbrace{\begin{bmatrix} -2\dot{\theta}_2 & 0 \\ \dot{\theta}_1 & 0 \end{bmatrix}}_{C(\theta,\dot{\theta})} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0 & -2\dot{\theta}_1 \\ \dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$

More Discussions

- Typical expression for c $\quad [C]_{ij} = \sum_{k=1}^{n} \frac{1}{2}\left(\frac{\partial m_{ij}}{\partial \theta_k} + \frac{m_{ik}}{\partial \theta_j} - \frac{\partial m_{jk}}{\partial \theta_i} \right) \dot{\theta}_k$

$\triangleq \Gamma_{ijk}$ christoffel

- $c(\theta, \dot{\theta})$ defined using Γ_{ijk}

satisfies: $\underbrace{\dot{M} - 2C}_{n \times n}$ is skew symmetric

- $M(\theta), C(\theta, \dot{\theta}), \tau_g$ can depend on $\mathcal{I}_i$ linearly

$\Downarrow$

$M(\theta) \triangleq \sum_i J_i^T \mathcal{I}_i J_i$

Fix θ

$M(\mathcal{I}_i)$ $\qquad M(\alpha \mathcal{I}_i^{(1)} + \beta \mathcal{I}_i^{(2)}) = \alpha M(\mathcal{I}_i^{(1)}) + \beta M(\mathcal{I}_i^{(2)})$

$f(x)$

$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$

$\Rightarrow$ Identification of $\{\mathcal{I}_i\}$ can be done using linear least squares.