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/

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:

external wrench

$$\underbrace{\mathcal{F} = \frac{d}{dt}h = \left(\mathcal{I}\mathcal{A} + \mathcal{V} \times^* \mathcal{I}\mathcal{V}\right)}_{\text{coordinate free}} \quad \cdots \quad (1)$$

$$\mathcal{F}_2 + mg - \mathcal{F}_3 = \mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{V}_2$$

- 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:

$\tau \in \mathbb{R}^n$

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

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

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

simulation

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

initial

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

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

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

find the required torque to generate the 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

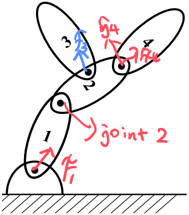
- Balance of forces/torques
- Dynamic equations in numeric/recursive form
- Often used for numerical solution of forward/inverse dynamics

- We focus on Newton-Euler Formulation Featherstone

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. $p(2)=1$, $p(3)=2$, $p(4)=2$
  - Children: $c(i)$ — $c(2)=\{3,4\}$, $c(1)=\{2\}$
- Joint i connects $p(i)$ to i
- Frame {i} attached to body i at the joint i; moves with the body.
- $\mathcal{S}_i$: Spatial velocity (screw axis) of joint i

  $\mathcal{V}_{4/2} = S_4 \dot{\theta}_4$ — body twist relative to body 2

  e.g. ${}^4S_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$
- $\mathcal{V}_i$ and $\mathcal{A}_i$: spatial velocity and acceleration of body i
- $\mathcal{F}_i$: force (wrench) onto body i from body $p(i)$
- Note: By default, all vectors $(\mathcal{S}_i, \mathcal{V}_i, \mathcal{F}_i)$ are expressed in local frame {i}

  either means coordinate free
  or coordinates 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 $\mathcal{A}_i$

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

Goal: Derive inverse dynamics Algo.

Given: $\theta, \dot{\theta}, \ddot{\theta}$, all kinematics & Dynamics parameters of each body, find the required torque τ.

Recall: $\tau_i = S_i^T \mathcal{F}_i$, so we need to compute $\mathcal{F}_i$; to compute $\mathcal{F}_i$, we need $\mathcal{V}_i$, $\mathcal{A}_i$

Acclleration: $\mathcal{A}_2 = \dot{\mathcal{V}}_2 = \dot{\mathcal{V}}_1 + \dot{\mathcal{V}}_{2/1} = \mathcal{A}_1 + \mathcal{A}_{2/1}$

In coordinate: ${}^2\mathcal{A}_2 = {}^2X_1\, {}^1\mathcal{A}_1 + {}^2\left(\frac{d}{dt}\left[S_2\dot{\theta}_2\right]\right)$

Velocity: $\mathcal{V}_1 = S_1\dot{\theta}_1$, ${}^1\mathcal{V}_1 = {}^1S_1\dot{\theta}_1$

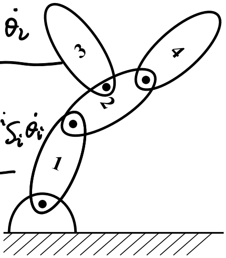
$\mathcal{V}_2 = \mathcal{V}_1 + \mathcal{V}_{2/1} = S_1\dot{\theta}_1 + S_2\dot{\theta}_2$

use {2} frame to express.

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

In general,

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



$$ {}^{2}\left(\frac{d}{dt}\left(s_2\dot{\theta}_2\right)\right) = \frac{d}{dt}\left({}^{2}S_2\dot{\theta}_2\right) + {}^{2}V_2 \times {}^{2}S_2\dot{\theta}_2 = {}^{2}S_2\ddot{\theta}_2 + {}^{2}V_2 \times {}^{2}S_2\dot{\theta}_2 $$

${}^{2}V_2$

$$ {}^{2}A_2 = {}^{2}X_1\,{}^{1}A_1 + {}^{2}V_2 \times {}^{2}S_2\dot{\theta}_2 + {}^{2}S_2\ddot{\theta}_2 $$

. . .

we can get $A_1, A_2, \dots A_n$

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:

$$\mathcal{F}_{g4} = \mathcal{I}_4 \mathcal{A}_g$$

$$\mathcal{F}_4 + \mathcal{F}_{g4} = \mathcal{I}_4 \mathcal{A}_4 + \mathcal{V}_4 \times^* \mathcal{I}_4 \mathcal{V}_4$$

$$\mathcal{F}_4 = \mathcal{I}_4 \mathcal{A}_4 + \mathcal{V}_4 \times^* \mathcal{I}_4 \mathcal{V}_4 - \mathcal{F}_{g4}.$$

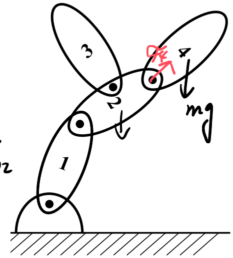
$$\tau_4 = \mathcal{S}_4^T \mathcal{F}_4$$

Body 3: similar.

Body 2: ~~similar~~

$$\mathcal{F}_2 = \mathcal{I}_2 \mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2 \mathcal{V}_2 + \mathcal{F}_3 + \mathcal{F}_4 - \mathcal{F}_{g2}$$

$$\tau_2 = \mathcal{S}_2^T \mathcal{F}_2$$



Recursive Newton-Euler Algorithm

kinematics
dynamics

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

Initialize: $\mathcal{V}_0 = 0$, $\mathcal{A}_0 = -\mathcal{A}_g$ (trick to "ignore" gravity)

- Forward pass: For $i = 1$ to N

$$\boxed{\begin{aligned} {}^i\mathcal{V}_i &= {}^iX_{p(i)} \, {}^{p(i)}\mathcal{V}_{p(i)} + {}^iS_i \dot{\theta}_i \\ \mathcal{A}_i &= {}^iX_{p(i)} \, {}^{p(i)}\mathcal{A}_{p(i)} + {}^iS_i \ddot{\theta}_i + {}^i\mathcal{V}_i \times {}^iS_i \dot{\theta}_i \\ {}^i\mathcal{F}_i &= {}^i\mathcal{I}_i \, {}^i\mathcal{A}_i + {}^i\mathcal{V}_i \times^* {}^i\mathcal{I}_i \, {}^i\mathcal{V}_i \end{aligned}}$$

wrench to generate motion of body i

- Backward pass:

For $i = N : -1 : 1$

$$\underline{\tau_i = {}^iS_i^T \, {}^i\mathcal{F}_i}$$

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

End

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): $J_1, \ldots, J_4$

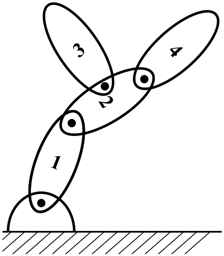
J_i : denote the Jacobian of body i , e.g. $\mathcal{V}_i = J_i \dot{\theta} = [J_{i1} \; J_{i2} \; J_{i3} \; J_{i4}] \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_4 \end{bmatrix}$

link

$$\mathcal{V}_2 = J_2 \dot{\theta} = [S_1 \mid S_2 \mid 0 \mid 0] \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}$$

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

$${}^4J_4 = [{}^4X_1 {}^1S_1 \mid {}^4X_2 {}^2S_2 \mid 0 \mid {}^4S_4]$$



Structures in Dynamic Equation (2/3)

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

Consider 2-body problem, with external wrench $\mathcal{F}_2^{ex}$ ∴ use RNEA to compute τ

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

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

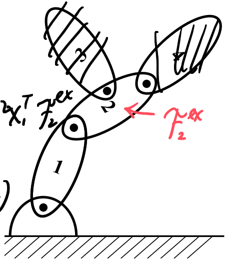
② Backward pass: $\mathcal{F}_2 = \mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{V}_2 - \mathcal{F}_2^{ex}$

$\to \tau_2 = S_2^T(\mathcal{I}_2\mathcal{A}_2 + \cdots) - S_2^T\mathcal{F}_2^{ex}$

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

$$= \mathcal{I}_1\mathcal{A}_1 + \mathcal{V}_1 \times^* \mathcal{I}_1\mathcal{V}_1 + ({}^2X_1)^T(\mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{V}_2) - {}^2X_1^T\mathcal{F}_2^{ex}$$

$$\tau_1 = S_1^T\mathcal{F}_1 = \underbrace{S_1^T(\mathcal{I}_1\mathcal{A}_1 + \mathcal{V}_1 \times^* \mathcal{I}_1\mathcal{V}_1)}_{①} + \underbrace{({}^2X_1 S_1)^T(\mathcal{I}_2\mathcal{A}_2 + \mathcal{V}_2 \times^* \mathcal{I}_2\mathcal{V}_2)}_{②} \underbrace{- ({}^2X_1 S_1)^T\mathcal{F}_2^{ex}}_{③}$$



Structures in Dynamic Equation (3/3)

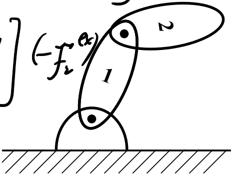
- Overall torque expression: ①: torque required at joint 1 to generate motion of body 1.

② torque @ Joint 1 due to motion of body 2

③ . . . . due to external wrench $\mathcal{F}_2^{ex}$

$$\tau = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} S_1^T(\mathcal{I}_1 A_1 + \cdots) + ({}^2X_1 S_1)^T(\mathcal{I}_2 A_2 + \cdots) + ({}^2X_1 S_1)^T(-\mathcal{F}_2^{ex}) \\ 0\cdot(\mathcal{I}_2 A_1 + \cdots) + S_2^T(\mathcal{I}_2 A_2 + \cdots) + S_2^T(-\mathcal{F}_2^{ex}) \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} S_1^T \\ 0 \end{bmatrix}}_{J_1^T}(\mathcal{I}_1 A_1 + \cdots) + \underbrace{\begin{bmatrix} ({}^2X_1 S_1)^T \\ S_2^T \end{bmatrix}}_{J_2^T}(\mathcal{I}_2 A_2 + \cdots) + \underbrace{\begin{bmatrix} {}^2X_1 S_1 \\ S_2^T \end{bmatrix}}_{J_2^T}(-\mathcal{F}_2^{ex})$$



Recall: $J_1 = [S_1 \quad 0]$, $J_2 = [{}^2X_1 S_1 \mid S_2]$

Derivation of Overall Dynamics Equation

- overall, in general with N- bodies.

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

from body to enviroment

To relate to joint variables.

$\mathcal{V}_i = J_i\dot\theta$, $\mathcal{A}_i = \dot{\mathcal{V}}_i = J_i\ddot\theta + \underbrace{\dot J_i\dot\theta_i +}_{\mathcal{V}_i \times J_i\dot\theta_i}$

F_1

F_2^{ex}

e.g. this term is $-F_2^{ex}$

$J_1^T(-F_1)$

$$\tau = \sum_{i=1}^{N}\left(J_i^T\mathcal{I}_iJ_i\ddot\theta + J_i^T\mathcal{I}_i\dot J_i\dot\theta + J_i^T\mathcal{I}_i\mathcal{V}_i \times J_i\dot\theta + J_i^T\mathcal{V}_i\times^*\mathcal{I}_i\mathcal{V}_i + J_i^T(\text{external})\right)$$

$J_i\dot\theta$

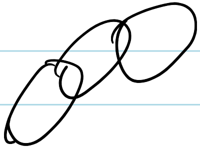
$$= \underbrace{\left(\sum_{i=1}^{N}(J_i^T\mathcal{I}_iJ_i)\right)}_{\triangleq M(\theta)}\ddot\theta + \underbrace{\sum_{i=1}^{N}\left(J_i^T(\mathcal{I}_i\dot J_i + \mathcal{I}_i\mathcal{V}_i\times J_i + \mathcal{V}_i\times^*\mathcal{I}_iJ_i)\dot\theta\right)}_{\triangleq C(\theta,\dot\theta)\dot\theta}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

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

If we consider gravity, we need to add $\boxed{\sum J_i^T\mathcal{I}_i\,{}^iX_o(-{}^oA_g)} = \tau_g \in \mathbb{R}^n$

single rigid body

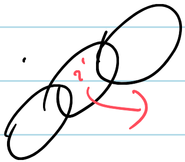
Recall:  $\underbrace{\mathcal{F} = \mathcal{I}\mathcal{A} + \mathcal{V}\times^{*} \mathcal{I}\mathcal{V}}$

 : Forward Dynamics:

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

Inverse Dynamics.

$$\tau = ID(\theta, \dot{\theta}, \ddot{\theta}, \mathcal{F}_{ext}) \Leftarrow RNEA$$

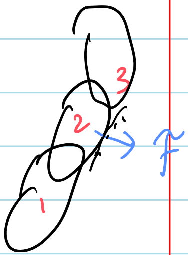
- RNEA ⇒ derive robot Dynamics.
-  $J_i \triangleq$ body/Link i Jacobian. $\mathcal{V}_i = J_i \dot{\theta}$

$$= \begin{bmatrix} J_{i1} & \cdots & J_{in} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{bmatrix}$$

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

① : generate motion

② : generate wrench



- If we only consider body's effect (without gravity)

$$\tau = J_2^T(\mathcal{I}_2 A_2 + \mathcal{V}_2 \times^* \mathcal{I}_2 \mathcal{V}_2) + J_2^T \mathcal{F}$$

(If we consider gravity of body 2:)

$$\tau = J_2^T(\mathcal{I}_2 A_2 + \mathcal{V}_2 \times^* \mathcal{I}_2 \mathcal{V}_2) + J_2^T \mathcal{F} + J_2^T(-\mathcal{I}_2\, {}^2X_0\, {}^0A_g)$$

Overall, the overall dynamics.

$$\tau = \text{all motions} + \text{all external forces}.$$

$$= \sum_i \left[J_i^T(\mathcal{I}_i A_i + \mathcal{V}_i \times^* \mathcal{I}_i \mathcal{V}_i) + J_i^T(-\mathcal{I}_i\, {}^iX_0\, {}^0A_g) \right]$$

Properties of Dynamics Model of Multi-body Systems

-

Outline

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

Forward Dynamics Problem

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

given — unknown to be solved — given

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

given

$\ominus = \tilde{c}(\theta, \dot{\theta})$

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

     given

     we have computed

This is not the most efficient way to do FD

ABA

Calculations of $\tilde{c}$ and M

does not depend on $\ddot{\theta}$

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

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

- **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 $\tilde{c}(\theta, \dot{\theta}) = 0 \Rightarrow \tau = M(\theta)\ddot{\theta}$

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

    $\ddot{\theta}_j^0 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ 0 \end{bmatrix} \leftarrow j^{th}$-row
$$M_{:,j}(\theta) = \text{RNEA}(\theta, 0, \ddot{\theta}_j^0, 0)$$
    If $\ddot{\theta} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, then $\tau = M_1(\theta)$

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

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

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

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

- 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): $O(N)$

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

$$\underbrace{\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}}_{2n \times 1} = \begin{bmatrix} x_2 \\ \underbrace{M^{-1}(x_1)(\tau - \tilde{c}(x_1, x_2))}_{\text{vector field}} \end{bmatrix}$$

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

$x_{k+1} = x_k + \Delta t \cdot f(x_k)$

x_0

More Discussions

symmetric, p.s.d.

- $$\tau = \Big(\underbrace{\Big(\sum_i J_i^T \mathcal{I}_i J_i\Big)}_{M(\theta)}\Big)\ddot{\theta} + \underbrace{\Big(\Big(\sum (\quad)\Big)\dot{\theta}\Big)}_{C(\theta,\dot{\theta})} + \tau_g \cdots$$

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

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

$C(\theta,\dot{\theta})$

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

also $$= \begin{bmatrix} 0 & -2\dot{\theta}_1 \\ \dot{\theta}_1 & 0 \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

More Discussions

Γ_{ijk} - christoffel

- Typical expression for c: $[C_{ij}] = \sum_{k=1} \frac{1}{2}\left(\frac{\partial M_{ij}}{\partial \theta_k} + \frac{\partial M_{ik}}{\partial \theta_j} - \frac{\partial M_{jk}}{\partial \theta_i}\right) \dot{\theta}_k$

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

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

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

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

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

$$\tau = g(I_i ; \theta, \dot{\theta}, \ddot{\theta})$$

$$y = Hx + v$$

$$x = (H^T H)^{-1} H^T y$$

system ID of robot Dynamics can be done using least square.

$$I_i = \begin{bmatrix} I_{xx} & I_{xy} & \cdots & & \\ I_{yx} & \ddots & & 0 & \\ & & & & \\ 0 & & & m I_{3\times3} & \end{bmatrix}$$

e.g. $I_i = \begin{bmatrix} \begin{matrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{matrix} & 0 \\ 0 & m I_{3\times3} \end{bmatrix}$

$$\beta = \begin{bmatrix} I_{xx} \\ I_{yy} \\ I_{zz} \\ m \end{bmatrix}$$

$$\tau = \underbrace{\tilde{g}}_{\text{linear in } \beta}(\beta ; \theta, \dot{\theta}, \ddot{\theta}) = \underline{G \beta}$$

$G \to (\theta, \dot{\theta}, \ddot{\theta})$

$$I_i = I_{xx}\begin{bmatrix} 1 & 0 & 0 & \\ 0 & 0 & 0 & 0 \\ 0 & 0 & & \\ & 0 & & 0 \end{bmatrix} + I_{yy}\begin{bmatrix} 0 & 0 & & \\ 0 & 1 & 0 & \\ & & & \\ & & 0 & \end{bmatrix}$$

$$+ I_{zz}\begin{bmatrix} & \\ & \end{bmatrix} + m\begin{bmatrix} & \\ & \end{bmatrix}$$

$$\tau = \sum \beta_i \begin{bmatrix} & \end{bmatrix} \begin{bmatrix} I_{xx} \\ \\ \end{bmatrix}$$

$$\begin{bmatrix} \tau(t_1) \\ \tau(t_2) \\ \tau(t_3) \\ \vdots \end{bmatrix} = \begin{bmatrix} G(\underbrace{\theta(t_1), \dot{\theta}(t_1), \ddot{\theta}(t_1)}) \\ G(t_2 \qquad) \\ \vdots \\ \vdots \end{bmatrix} \beta$$