ECE5463: Introduction to Robotics
Lecture Note 13: Dynamics of Open Chains: Newton Euler Approach

Prof. Wei Zhang

Department of Electrical and Computer Engineering
Ohio State University
Columbus, Ohio, USA

Spring 2018
Outline

• Introduction

• Newton-Euler Inverse Dynamics of Open Chains

• Forward Dynamics of Open Chains
Lagrangian vs. Newton-Euler Methods

- We have introduced the Lagrangian approach in the previous lecture. This lecture will introduce the Newton-Euler method.

- Recall the differences between the two methods:

**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
Main Ideas for Inverse Dynamics

- **Goal:** Given state \((\theta, \dot{\theta})\) and acceleration \(\ddot{\theta}\), compute the required \(\tau \in \mathbb{R}^n\)

\[
\tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta})
\]

- We will not drive the analytical expressions for the overall \(M(\theta)\) and \(h(\theta, \dot{\theta})\).

- Instead, we aim to write dynamic equations separately for each link/body.

\[
F_{i}^{total} = G_i \dot{V}_i - \left[ \text{ad}_V_i \right]^T (G_i V_i)
\]
Main Ideas for Inverse Dynamics (Continued)

• Note that $G_i$ can be easily computed for each link. Thus, the main task is to find $V_i, \dot{V}_i$ and the total wrench for each link in terms of the given $(\theta, \dot{\theta}, \ddot{\theta})$.

• This can be done recursively:
  - The velocity $V_i$ of link $i$ is determined by $V_{i-1}$ of link $i-1$ and the joint rate $\dot{\theta}_i$ of joint $i$
  - The acceleration $\dot{V}_i$ of link $i$ is determined by $\dot{V}_{i-1}$ of link $i-1$ and the acceleration $\ddot{\theta}_i$ of joint $i$
  - The total wrench of link $i$ can be determined by gravity and the wrench of adjacent links using the principle of action and reaction:

    \[
    \text{Wrench applied by body } i \text{ to body } i+1 \nonumber \]
    \[
    = - \text{Wrench applied by body } i+1 \text{ to body } i \nonumber
    \]
Outline

- Introduction
- Newton-Euler Inverse Dynamics of Open Chains
- Forward Dynamics of Open Chains
Some Notations (1/3)

• For each link $i = 1, \ldots, n$, Frame $\{i\}$ is attached to its center of mass.

• The base frame is denoted $\{0\}$.

• The end-effector frame is denoted $\{n + 1\}$. Its configuration in $\{n\}$ is fixed.

• $M_{i,j}$: Configuration of $\{j\}$ in $\{i\}$ when the robot is at home position.

• $T_{i,j}(\theta)$: Configuration of $\{j\}$ in $\{i\}$ when the robot is at $\theta$ position.

\[ \theta = 0 \quad \Rightarrow \quad T_{i,j}(0) = M_{i,j} \]
Some Notations (2/3)

- All the following quantities are expressed in frame \( \{i\} \)
  - \( \mathcal{V}_i \): Twist of link \( \{i\} \)
  - \( \bar{m}_i \): mass; \( \mathcal{I}_i \): rotational inertia matrix;
  - \( G_i = \begin{bmatrix} \mathcal{I}_i & 0 \\ 0 & \bar{m}_i I \end{bmatrix} \): Spatial inertia matrix
  - \( A_i \): Screw axis of joint \( i \), expressed in \( \{i\} \)
  - \( \mathcal{F}_i = (m_i, f_i) \): wrench transmitted through joint \( i \) to link \( \{i\} \)
  - \( S_i \): Screw axis of joint \( i \), expressed in \( \{53\} \), when \( \theta = 0 \)
Some Notations (3/3)

- Some simple facts:
  - \( M_{i-1,i} = M_{i-1}^{-1} M_i \) and \( M_{i,i-1} = M_i^{-1} M_{i-1} \)

Define \( M_{i} \in M_{0,i} \Rightarrow M_{i-1,i} = M_{i-1,0} M_{0,i} = M_{i-1}^{-1} M_i \)

- \( A_i = \left[ \text{Ad}_{M_i^{-1}} \right] S_i \), where \( S_i \) is the screw axis of joint \( i \) (relative to \( \{0\} \) frame) when \( \theta = 0 \)

\[
A_i = \left[ \text{Ad}_{T_{i,0}} \right] S_i
\]

Note: \( A_i \) is relative to (link\( i+1 \)) frame, so it does not change with \( \theta \)

- \( T_{i-1,i}(\theta) = T_{i-1,i}(\theta_i) = M_{i-1,i} e^{[A_i] \theta} \) and \( T_{i,i-1}(\theta_i) = e^{-[A_i] \theta_i} M_{i,i-1} \)
Forward Propagation of Twist (1/3)

- **Goal:** Given $\mathcal{V}_{i-1}$ and $(\theta_i, \dot{\theta}_i)$, we want to find $\mathcal{V}_i$

- Generally speaking, there are two key elements in defining a rigid body velocity:
  - The inertia frame relative to which the motion is considered
  - The representation frame in which the coordinate of the velocity vector is determined.

- So far, we have mainly played with the representation frames, while the motion is always considered to be relative to the fixed $\{s\}$ frame.

- Sometimes, it is convenient consider a relative velocity with respect to a moving frame.
Forward Propagation of Twist (2/3)

- Define $\mathcal{V}_{a/c}^b$ as the velocity (twist) of $\{b\}$ relative to $\{c\}$, expressed in $\{a\}$.

- Connecting to our previous shorthand notations:

$$\mathcal{V}_s = \mathcal{V}_{s/s}^b, \quad \mathcal{V}_b = \mathcal{V}_{b/s}^b, \quad \mathcal{V}_i = \mathcal{V}_{i/s}^i$$

- With Newtonian approximation: we have

$$\mathcal{V}_{*/s}^c = \mathcal{V}_{*/s}^b + \mathcal{V}_{*}^{c/b}$$

- The "*" can be replaced with $\{s\}$ or $\{b\}$ or any other stationary frames.

- Note the representation frame must be the same for all the three twists.
Forward Propagation of Twist (3/3)

- Now consider velocity of link $i$: we will have

$$\mathbf{V}_i^{i/s} = \mathbf{V}_i^{i/i-1} + \mathbf{V}_{i-1}^{i-1/s}$$

- Note that $\mathbf{V}_i^{i/i-1}$ can be viewed as the body twist of a 1-joint robot where \{i-1\} is viewed as the “fixed” frame $\Rightarrow \mathbf{V}_i^{i/i-1} = \mathbf{A}_i \dot{\theta}_i$

- Moreover, with our shorthand notation:

$$\mathbf{V}_i^{i-1/s} = [\text{Ad}_{T_i,i-1}] \mathbf{V}_{i-1}^{i-1/s} = [\text{Ad}_{T_i,i-1}] \mathbf{V}_{i-1}$$

- Therefore, velocity (twist) can be propagated by:

$$\mathbf{V}_i = \mathbf{A}_i \dot{\theta}_i + [\text{Ad}_{T_i,i-1}] \mathbf{V}_{i-1} \quad (1)$$
Forward Propagation of Acceleration (1/2)

• **Goal**: Given $\dot{\mathbf{v}}_{i-1}$, $\mathbf{v}_{i-1}$ and $(\theta_i, \dot{\theta}_i, \ddot{\theta}_i)$, we want to find $\dot{\mathbf{v}}_i$,

• We first mention a result about the derivative of the adjoint operator.

• Given a time-varying $T(t) = (R(t), p(t))$, the associated adjoint $[\text{Ad}_{T(t)}]$ is also a time-varying matrix. We have the following result about its derivative:

**Derivative of adjoint**: If $T(t) = e^{[\mathcal{V}]\theta(t)} M$, where $\mathcal{V}$ is a constant twist and $M$ is a constant matrix in $SE(3)$, and $\theta(t)$ is a scalar time-varying function. Then

\[
\frac{d}{dt} \text{Ad}_{T(t)} = [\text{ad}_{\mathcal{V}\dot{\theta}}] \text{Ad}_{T(t)}
\] (2)
Forward Propagation of Acceleration (2/2)

- With the result in (2), we can propagate acceleration by taking derivative on both sides of (1):

\[ \dot{V}_i = A_i \ddot{\theta}_i + [\text{Ad}_{T_{i,i-1}}] \dot{V}_{i-1} + \frac{d}{dt} \left[ \text{Ad}_{T_{i,i-1}} \right] V_{i-1} \]

\[ \Rightarrow \dot{V}_i = A_i \ddot{\theta}_i + [\text{Ad}_{T_{i,i-1}}] \dot{V}_{i-1} + [\text{ad}_{\dot{V}_i}] A_i \dot{\theta}_i \]

\[ T_{i,i-1} = e^{-[A_i] \theta_o} M_{i,i-1} \]

\[ \Rightarrow \frac{d}{dt} \left[ \text{Ad}_{T_{i,i-1}} \right] = [\text{ad}_{-A_i \dot{\theta}_i}] \left[ \text{Ad}_{T_{i,i-1}} \right] V_{i-1} \]

\[ = [\text{ad}_{-A_i \dot{\theta}_i}] V_i = -[\text{ad}_{\dot{V}_i}] A_i \dot{\theta}_i \]

- Thus, the acceleration of line \( i \) has three components:
  - a component due to joint acceleration \( \ddot{\theta} \)
  - a component due to the acceleration of link \( i-1 \) expressed in \( \{i\} \)
  - a velocity-product component
Backward Propagation of Wrenches

- Suppose we have computed the twist $\mathcal{V}_i$ and acceleration $\dot{\mathcal{V}}_i$ for all the links.

- Now we can compute forces/torques. The Newton-Euler Eq. for link $i$ is:

\[
\mathcal{F}_i - \left[\text{Ad}_{T_{i+1,i}}\right]^T \mathcal{F}_{i+1} = \mathcal{G}_i \dot{\mathcal{V}}_i - [\text{ad}_{\mathcal{V}_i}]^T (\mathcal{G}_i \mathcal{V}_i)
\]

- If we know $\mathcal{F}_{i+1}$, we can solve for $\mathcal{F}_i$.

- We can compute all the wrenches $\mathcal{F}_i$ backward from joint $n$ to joint 1.

- Each joint has 1 dof $\Rightarrow \tau_i = \mathcal{F}_i^T A_i$

\[
\tau_i \dot{\theta}_i = \mathcal{F}_i^T (A_i, \dot{A}_i, \ddot{A}_i)
\]

Twist due to Joint $i$ motion.
Newton-Euler Inverse Dynamics Algorithm

**Initialization:**
- $V_0 = 0$ and $\dot{V}_0 = (\dot{\omega}_0, \dot{v}_0) = (0, -g)$
  - Setting $\dot{v}_0 = -g$ is a trick to handle gravity; otherwise one needs to consider gravity for each link during the backward wrench propagation

- $\mathcal{F}_{n+1}$: given a priori which is the wrench applied by the environment expressed in the end-effector frame

**Forward Iterations:** Given $\theta, \dot{\theta}, \ddot{\theta}$, for $i = 1 \rightarrow n$ do:

$$T_{i,i-1}(\theta_i) = e^{-[A_i]^{\theta_i}M_{i,i-1}}$$  \hspace{1cm} (3)

$$\dot{V}_i = [\text{Ad}T_{i,i-1}] \dot{V}_{i-1} + A_i \dot{\theta}_i$$ \hspace{1cm} (4)

$$\ddot{V}_i = [\text{Ad}T_{i,i-1}] \ddot{V}_{i-1} + [\text{ad}_{V_i}] A_i \dot{\theta}_i + A_i \ddot{\theta}_i$$ \hspace{1cm} (5)

**Backward Iterations:** For $i = n$ to 1, do

$$\mathcal{F}_i = [\text{Ad}T_{i+1,i}]^T \mathcal{F}_{i+1} + G_i \dot{V}_i - [\text{ad}_{V_i}]^T (G_i V_i)$$ \hspace{1cm} (6)

$$\tau_i = \mathcal{F}_i^T A_i$$ \hspace{1cm} (7)

**Return:** Joint torque vector $\tau \in \mathbb{R}^n$
Outline

- Introduction

- Newton-Euler Inverse Dynamics of Open Chains

- Forward Dynamics of Open Chains
Forward Dynamics Problem

- Overall dynamics equation (with nonzero wrench applied by the end-effector):

\[ M(\theta)\ddot{\theta} = \tau(t) - h(\theta, \dot{\theta}) - J^T(\theta)F_{tip} \]  \(\text{(8)}\)

- Forward dynamics: Given \(\theta, \dot{\theta}, \tau\) and \(F_{tip}\), compute the acceleration \(\ddot{\theta}\)

\[ \ddot{\theta} = M(\theta)^{-1} \left( \tau(t) - h(\theta, \dot{\theta}) - J^T(\theta)F_{tip} \right) \]  \(\text{(8a)}\)

- Jacobian \(J(\theta)\) can be obtained relatively easily.

- However, \(M(\theta)\) and \(h(\theta, \dot{\theta})\) can be hard to derive analytically

- We can use inverse dynamics algorithm to numerically compute \(M(\theta)\) and \(h(\theta, \dot{\theta})\) for any given joint state \(\theta, \dot{\theta}\)
 Computations of $M(\theta)$ and $h(\theta, \dot{\theta})$

- Denote our inverse dynamics algorithm: $\tau = \text{InverseDyn}(\theta, \dot{\theta}, \ddot{\theta}, F_{\text{tip}})$

- Obviously, $\tau = h(\theta, \dot{\theta})$ if $F_{\text{tip}} = 0$ and $\ddot{\theta} = 0$

- Therefore, we can compute $h(\theta, \dot{\theta}) = \text{InverseDyn}(\theta, \dot{\theta}, 0, 0)$

- Recall the form of $h(\theta, \dot{\theta}) = C(\theta, \dot{\theta})\dot{\theta} + g(\theta)$.
  - If ignore external forces, i.e. set $g = 0$ and $F_{\text{tip}} = 0$, then $g(\theta) = 0$
  - If we further choose $\dot{\theta} = 0$, then $h(\theta, \dot{\theta}) = 0 \Rightarrow \tau = M(\theta)\ddot{\theta}$

- We can compute the $j$th column of $M(\theta)$ by calling the inverse algorithm $M_{:,j}(\theta) = \text{InverseDyn}(\theta, 0, \ddot{\theta}_0^j, 0)$ where $\ddot{\theta}_0^j$ is a vector with all zeros except for a 1 at the $j$th entry.
Forward Dynamics Algorithm

• Given $\theta, \dot{\theta}$, the last slide shows how to compute $M(\theta)$ and $h(\theta, \dot{\theta})$

• Now we assume we have $\theta, \dot{\theta}, \tau, M(\theta), h(\theta, \dot{\theta}), F_{tip}$, then we can immediately compute $\ddot{\theta}$ using (8).

\[ \ddot{\theta} = \text{ForwardDyn}(\theta, \dot{\theta}, \tau, F_{tip}) \]

• Denote this algorithm as $\ddot{\theta} = \text{ForwardDyn}(\theta, \dot{\theta}, \tau, F_{tip})$

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

\[ x(t + \Delta t) = x(t) + f(x, x_2, \tau, F_{tip}) \Delta t \]
Examples

MATLAB Examples
More Discussions

Body frame $F_i$: 

$T(i) = M e^{(B_2)\theta_2} e^{(B_1)\theta_1}$

$B_{S_1} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{L_1+c}{2} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

$\mathbf{J}_{B_2} = \begin{bmatrix} 1 & 0 & 0 & -\frac{L_2+c}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

$M_{12} = \mathbf{J} (R_{B_2}) (J_{B_1}) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{L_1+c}{2} & 0 \\ 1 & 0 & 0 & \frac{L_2+c}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow T_{12}(\theta_2) = M_{12} e^{(B_2)\theta_2}$

$A_{2} = \begin{bmatrix} w_{A_2} \\ v_{A_2} \end{bmatrix}$, $\omega_{A_2} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $v_{A_2} = -w_{A_2} \times q_{A_2} = -\begin{bmatrix} 0 \\ 0 \end{bmatrix} \times \begin{bmatrix} \frac{L_2+c}{2} \\ 0 \end{bmatrix}$
Matlab Example:

Inertia matrix:

\[ \text{mass} = \rho \cdot L \cdot c^2 \]

\[ I_{xx} = \text{mass} \left( \frac{c^2 + c^2}{2} \right) \]

\[ I_{yy} = I_{zz} = \text{mass} \cdot \frac{L^2 + c^2}{2} \]

\[ I_b = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \]