Dynamics and Real-Time Simulation (DARTS) Laboratory

Spatial Operator Algebra (SOA)

3. Serial-Chain, Rigid-Body Kinematics

Abhinandan Jain

June 19, 2024

https://dartslab.jpl.nasa.gov/

Jet Propulsion Laboratory
California Institute of Technology

SOA Foundations Track Topics (serial-chain rigid body systems)

1. **Spatial (6D) notation** – spatial velocities, forces, inertias; spatial cross-product, rigid body transformations & properties; parallel axis theorem
2. **Single rigid body dynamics** – equations of motion about arbitrary frame using spatial notation
3. **Serial-chain kinematics** – minimal coordinate formulation, hinges, velocity recursions, Jacobians; first spatial operators; O(N) scatter and gather recursions
4. **Serial-chain dynamics** – equations of motion using spatial operators; Newton–Euler mass matrix factorization; O(N) inverse dynamics
5. **Mass matrix -** composite rigid body inertia; forward Lyapunov equation; mass matrix decomposition; mass matrix computation; alternative inverse dynamics
6. **Articulated body inertia -** Concept and definition; Riccati equation; alternative force decompositions
7. **Mass matrix factorization and inversion** – spatial operator identities; Innovations factorization of the mass matrix; Inversion of the mass matrix
8. **Recursive forward dynamics** – O(N) recursive forward dynamics algorithm; including gravity and external forces; inter-body forces identity

See https://dartslab.jpl.nasa.gov/References/index.php for publications and references on the SOA methodology.

Recap from last session

- Introduced 6D spatial notation to allow more concise and simpler handling of linear/angular terms together
  - Can work away from CM as needed
  - Rigid body transformation matrix
  - Generalized 6D cross-product
- Used spatial notation to derive equations of motion of a single rigid body

$$\mathfrak{f}(z) = M(z)\dot{\beta}_{\mathcal{J}}(z) + \mathfrak{b}_{\mathcal{J}}(z)$$

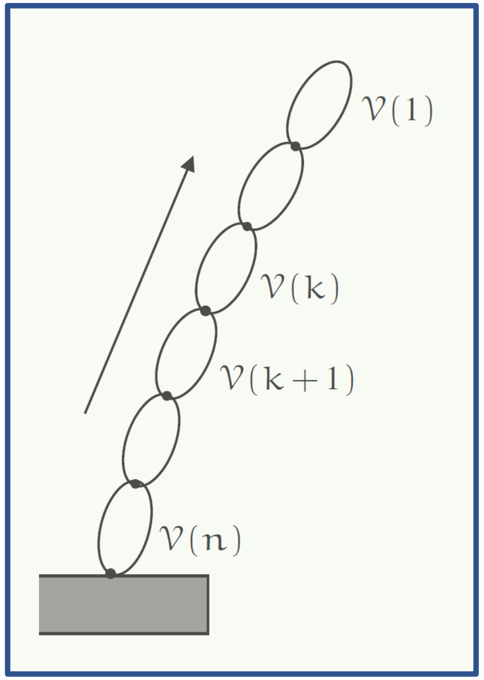
  - Equations capture both linear and rotational dynamics and their coupling
  - Used several choices for generalized velocities
  - Structure remained the same, variation in gyroscopic term

Serial-Chain Rigid Body Kinematics

Outline

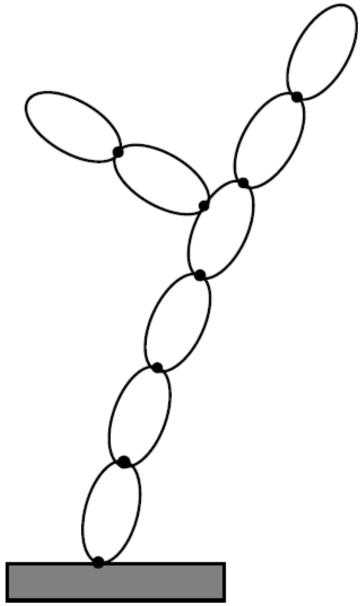
- Why serial chains?
- Hinges
- Configuration and velocity recursive kinematics
- Spatial operator representation
- Gather and scatter recursions
- Jacobians

Multibody system topologies



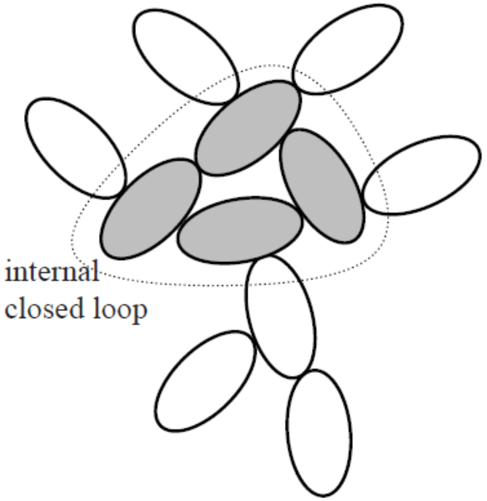
serial-chain systems

1 parent, 1 child



tree/branched systems

1 parent, children > 1



closed-chain systems

multiple parents

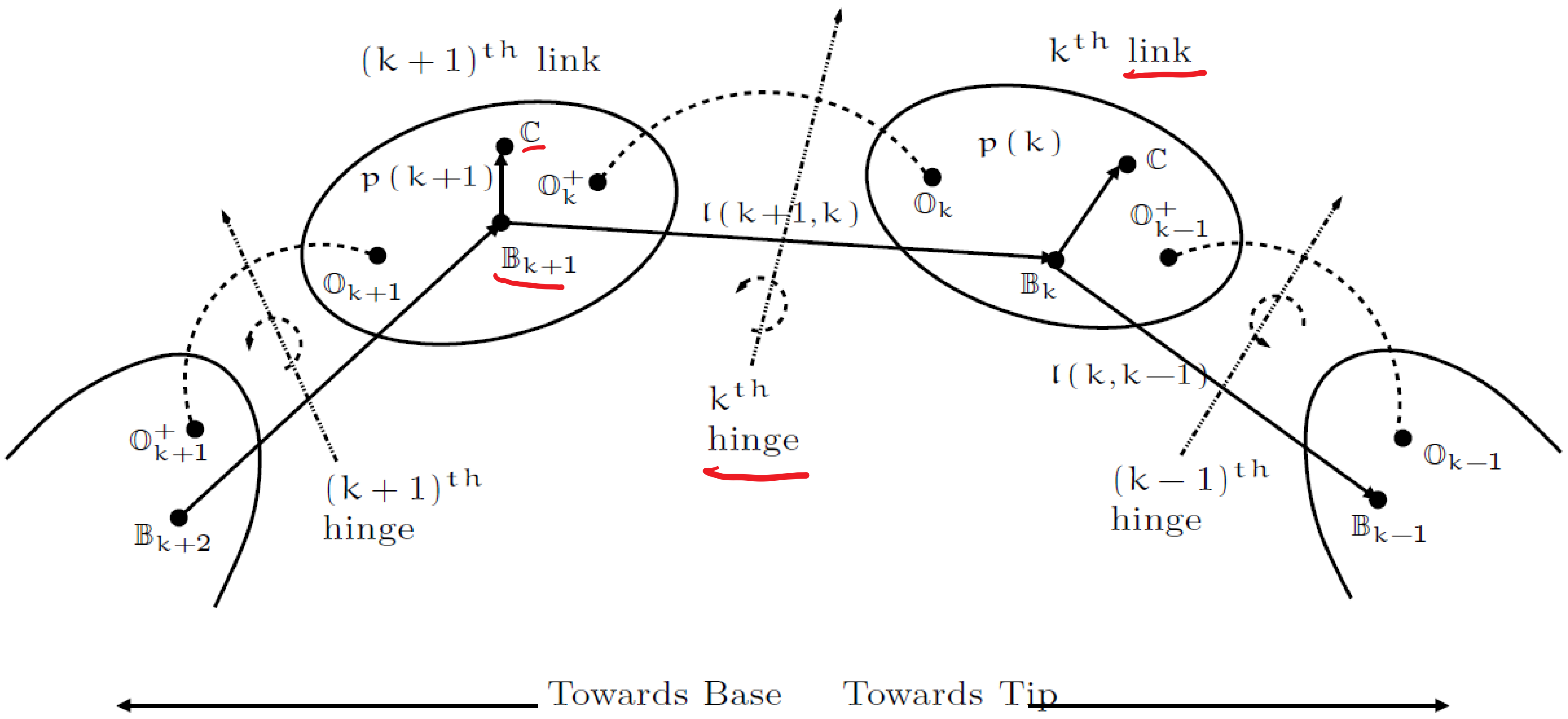
Why serial chains?

- Serial-chain rigid body systems are the simplest example of multibody systems
- However we do want to use SOA to tackle general multibody systems
  - rigid/flex bodies
  - arbitrary size and branched topologies
  - closed-chain topologies
- It turns out that the SOA methods developed for the serial-chain case carry over virtually entirely to the broader class of multibody systems
- Hence we will focus on serial-chains to simplify notation and will address generalization later

Hinges and Constraints

Inter-connected bodies

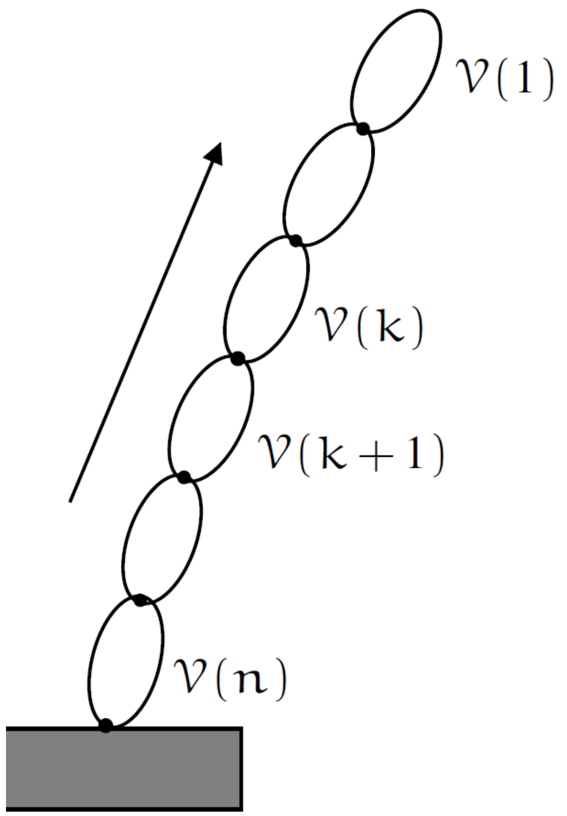
- Links/bodies are connected via hinges (aka *joints*)



Parent body index **k+1** > Child body index **k**; Tip body has index = **1**

Generalized coordinates & velocities

- Multibody state is the set of generalized coordinates and generalized velocities across all the hinges
- Often generalized velocities are just generalized coordinate time derivatives
- Quasi-velocities are a useful alternative

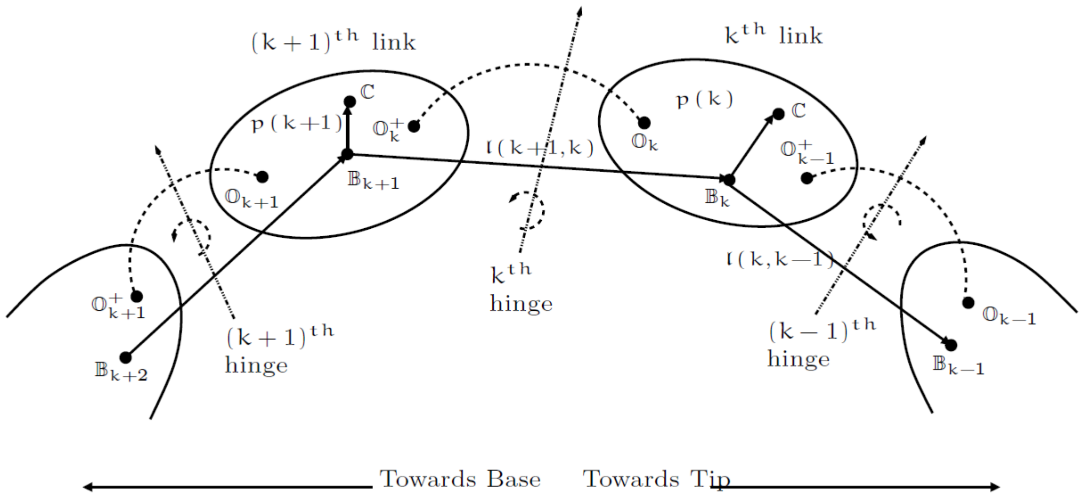


Jet Propulsion Laboratory
California Institute of Technology

Forward Configuration Kinematics

Relative body pose

The relative pose of connected bodies



constant for rigid bodies

$$ {}^{k+1}\mathbb{T}_{k} = {}^{k+1}\mathbb{T}_{\mathbb{O}_k} \cdot {}^{\mathbb{O}_k}\mathbb{T}_{k} = \left({}^{k+1}\mathbb{T}_{\mathbb{O}_k^+} \cdot {}^{\mathbb{O}_k^+}\mathbb{T}_{\mathbb{O}_k} \right) \cdot {}^{\mathbb{O}_k}\mathbb{T}_{k} $$

body to parent relative pose

hinge pose

Computing the pose of any body

- Forward kinematics problem is to compute the pose of any body in the system

$$^{\mathbb{I}}\mathbb{T}_{k} \overset{1.6}{=} {}^{\mathbb{I}}\mathbb{T}_{k+1} \cdot {}^{k+1}\mathbb{T}_{k}$$

- The computation of body poses can be done recursively from base to tip

$$\begin{cases} \quad {}^{\mathbb{I}}\mathbb{T}_{n+1} = \mathbf{I} \\ \textbf{for } \boldsymbol{k} \quad \mathbf{n \cdots 1} \\ \quad {}^{\mathbb{I}}\mathbb{T}_{k} = {}^{\mathbb{I}}\mathbb{T}_{k+1} \cdot {}^{k+1}\mathbb{T}_{k} \\ \textbf{end loop} \end{cases}$$

Computing relative pose

The relative pose of any pair of bodies j & k can also be computed recursively:

$$^{j}\mathbb{T}_{k} \overset{1.6}{=} {}^{j}\mathbb{T}_{j-1} \cdots {}^{k+1}\mathbb{T}_{k}$$

Hinge Map Matrix

Hinge differential kinematics

Time derivative of the hinge pose

$$\frac{d\,{}^{k+1}\mathbb{T}_{\mathbb{O}_k}}{dt} \overset{1.3}{=} \begin{pmatrix} \tilde{\omega}(\mathbb{O}_k^+, \mathbb{O}_k) & \nu(\mathbb{O}_k^+, \mathbb{O}_k) \\ \mathbf{0} & 0 \end{pmatrix}$$

$$\Delta_\omega(k) \triangleq \omega(\mathbb{O}_k^+, \mathbb{O}_k) \qquad \Delta_\nu(k) \triangleq \nu(\mathbb{O}_k^+, \mathbb{O}_k)$$

relative **angular** velocity — relative **linear** velocity

Joint map matrix

$$\Delta_{\mathcal{V}}(k) \triangleq \mathcal{V}(\mathbb{O}_k) - \mathcal{V}(\mathbb{O}_k^+) = \begin{bmatrix} \Delta_\omega(k) \\ \Delta_v(k) \end{bmatrix}$$

$$= H^*(k)\,\beta(k)$$

relative hinge spatial velocity → $\Delta_{\mathcal{V}}(k)$

joint map matrix → $H^*(k)$

generalized velocity → $\beta(k)$

The **joint map matrix** maps the (non-dimensional) hinge generalized velocities to the relative hinge spatial velocity

Structure of the joint map matrix

Can partition into angular/linear parts

$$H^*(k) = \begin{bmatrix} h_\omega(k) \\ h_\nu(k) \end{bmatrix}$$

$$\Delta_\omega(k) = h_\omega(k)\beta(k)$$

$$\Delta_\nu(k) = h_\nu(k)\beta(k)$$

$$\mathcal{R}^{6\times r_\nu(k)}$$

$$r_p(k) \geqslant r_\nu(k)$$

configuration dofs
can exceed
velocity dofs

Examples of joint map matrix

For different types of hinges

$$
\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}
\quad
\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}
\quad
\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \mathrm{p} \end{pmatrix}
\quad
\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}
\quad
\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}
$$

rotary pin hinge	prismatic hinge	helical hinge	cylindrical hinge	spherical hing
1 dof	1 dof	1 dof	2 dof	3 dof

For a full 6dof hinge, the hinge map matrix is the identity matrix.

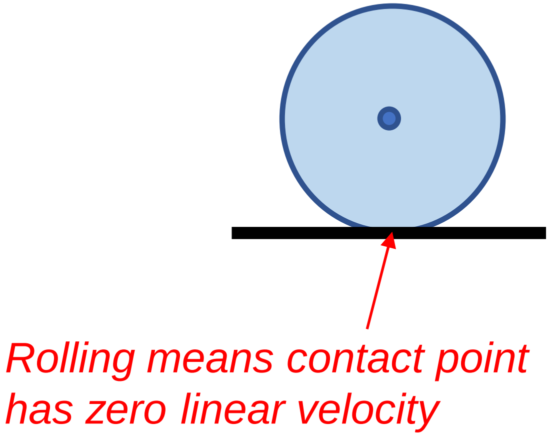
Example: Ball rolling on a surface

Rolling, no slipping can be treated as a hinge

$$\Delta_{\nu} + \widetilde{\Delta_{\omega}}\mathfrak{l} = \mathbf{0} \quad \Rightarrow \quad \begin{bmatrix} -\tilde{\mathfrak{l}}, & \mathbf{I}_3 \end{bmatrix} \Delta_{\mathcal{V}} = \mathbf{0}_3$$

$$\Delta_{\mathcal{V}} \triangleq \begin{bmatrix} \Delta_{\omega} \\ \Delta_{\nu} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 \\ \tilde{\mathfrak{l}} \end{bmatrix} \beta$$

$$\mathrm{H}^* = \begin{bmatrix} \mathbf{I}_3 \\ \tilde{\mathfrak{l}} \end{bmatrix} \in \mathcal{R}^{6\times 3}$$

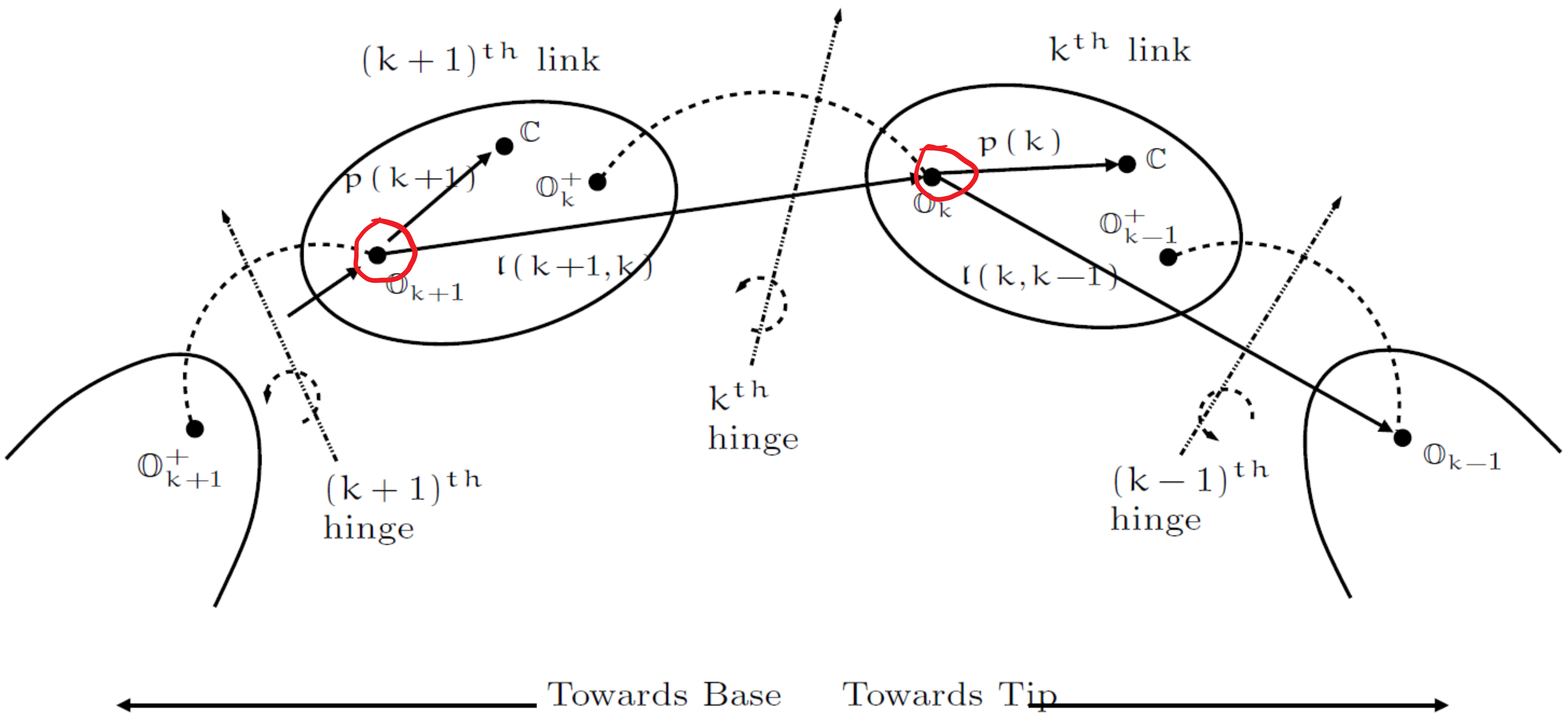


- Joint map matrix not constant in ball frame, only in inertial frame
- For an ellipsoid H is not constant.

Velocity Recursion

Body frame at hinge

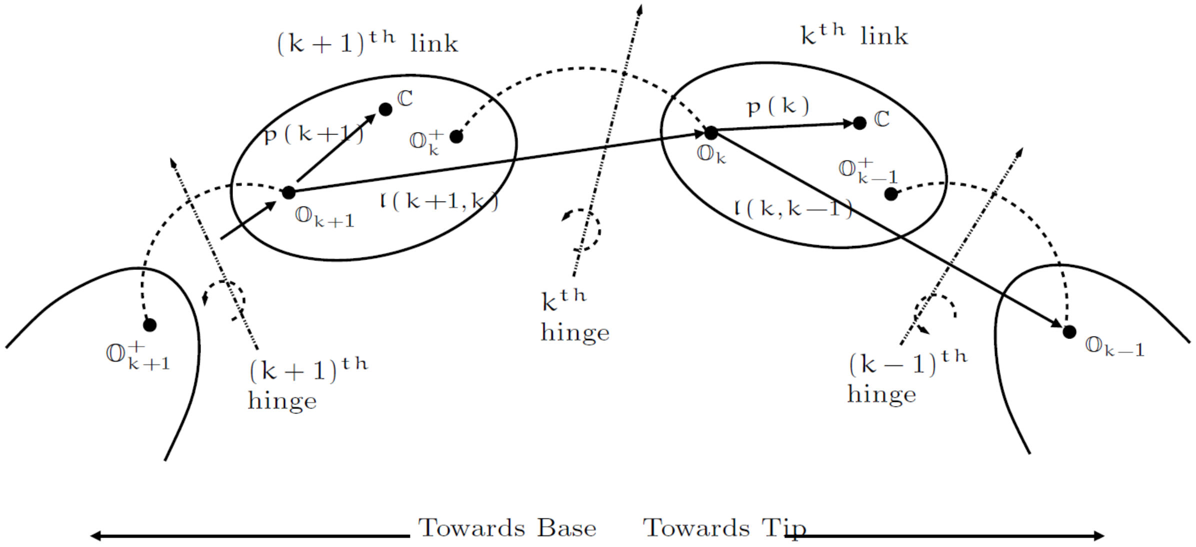
Body frame same as hinge frame $\mathbb{O}_k$



$$\mathbb{B}_k \equiv \mathbb{O}_k \qquad \mathcal{V}(k) \equiv \mathcal{V}(\mathbb{B}_k) \qquad {}^{k+1}\mathbb{T}_k = {}^{k+1}\mathbb{T}_{\mathbb{O}_k^+} \cdot {}^{\mathbb{O}_k^+}\mathbb{T}_{\mathbb{O}_k}$$

Inter-body rigid body transformation matrix

The rigid body transformation matrix from parent to child body



$$\mathfrak{l}(k+1,k) \triangleq \mathfrak{l}(\mathbb{B}_{k+1},\mathbb{B}_k)$$

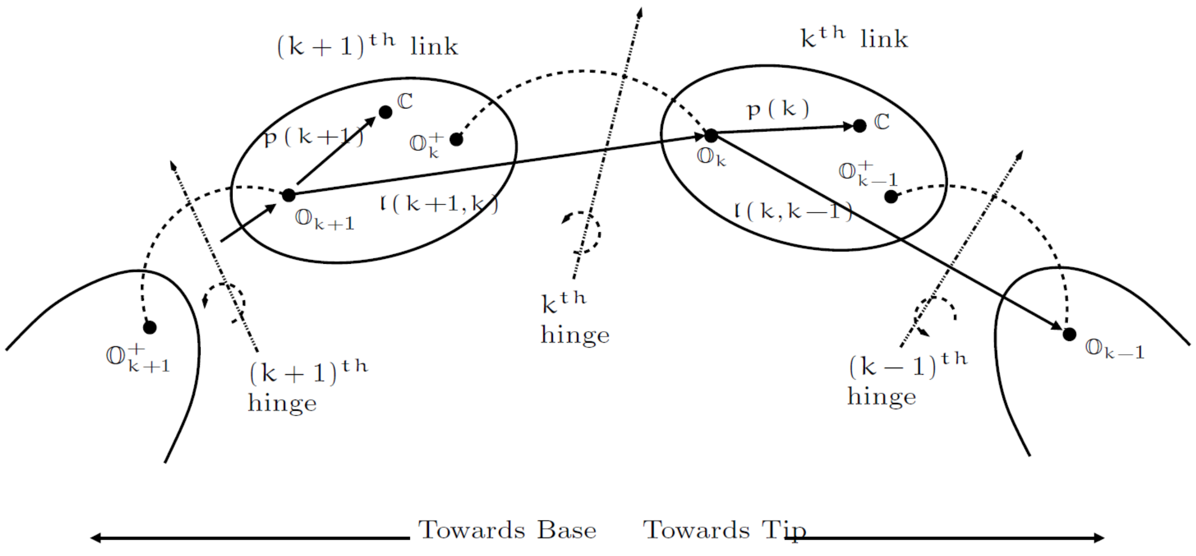
$$\boxed{\phi(k+1,k)} \triangleq \begin{pmatrix} \mathbf{I} & \widetilde{\mathfrak{l}}(\mathbb{B}_{k+1},\mathbb{B}_k) \\ \mathbf{0} & \mathbf{I} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \widetilde{\mathfrak{l}}(k+1,k) \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$$

$$\triangleq \phi(\mathbb{B}_{k+1},\mathbb{B}_k) = \phi(\mathbb{B}_{k+1},\mathbb{O}_k) = \phi(\mathbb{B}_{k+1},\mathbb{O}_k^+)$$

this is configuration dependent!

Body spatial velocity

Propagating body spatial velocity across hinges



$$\mathcal{V}^+(k) \overset{\triangle}{=} \mathcal{V}(\mathbb{O}_k^+)$$

$$\overset{1.37}{=} \phi^*(\mathbb{B}_{k+1}, \mathbb{O}_k^+)\mathcal{V}(k+1)$$

spatial velocity on the inboard side of the hinge

$$\mathcal{V}(k) \overset{3.12}{=} \mathcal{V}(\mathbb{O}_k) \overset{3.5}{=} \mathcal{V}^+(k) + \Delta_{\mathcal{V}}(k)$$

hinge contribution

spatial velocity on the outboard side of the hinge

Body spatial velocity computation

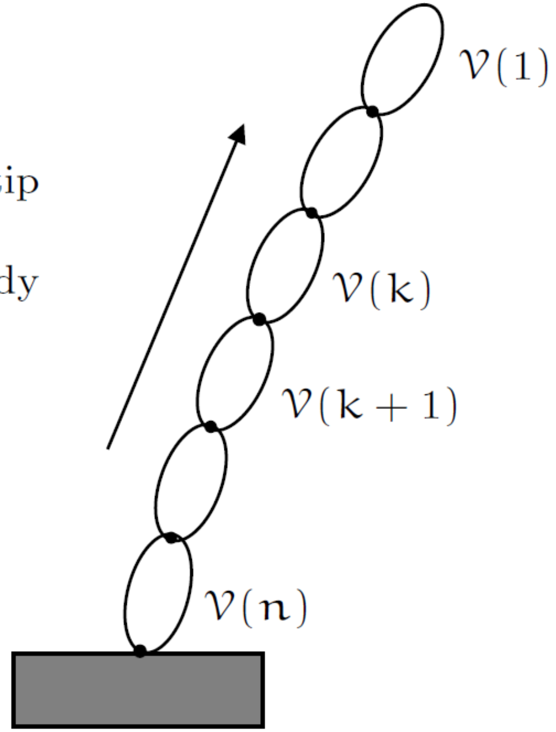
Body spatial velocities can be computed via a base-to-tip recursive algorithm

$$\mathcal{V}(k) \overset{3.14,3.19a}{=} \phi^*(k+1,k)\mathcal{V}(k+1) + \Delta_{\mathcal{V}}(k)$$
$$\overset{3.7}{=} \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\beta(k)$$

$$\begin{cases} \mathcal{V}(n+1) = 0 \\ \textbf{for } k \quad n \cdots 1 \\ \quad \mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\beta(k) \\ \textbf{end loop} \end{cases}$$

Recursive, base-to-tip, O(N) algorithm for the body spatial velocities

Base to tip recursion for body spatial velocities

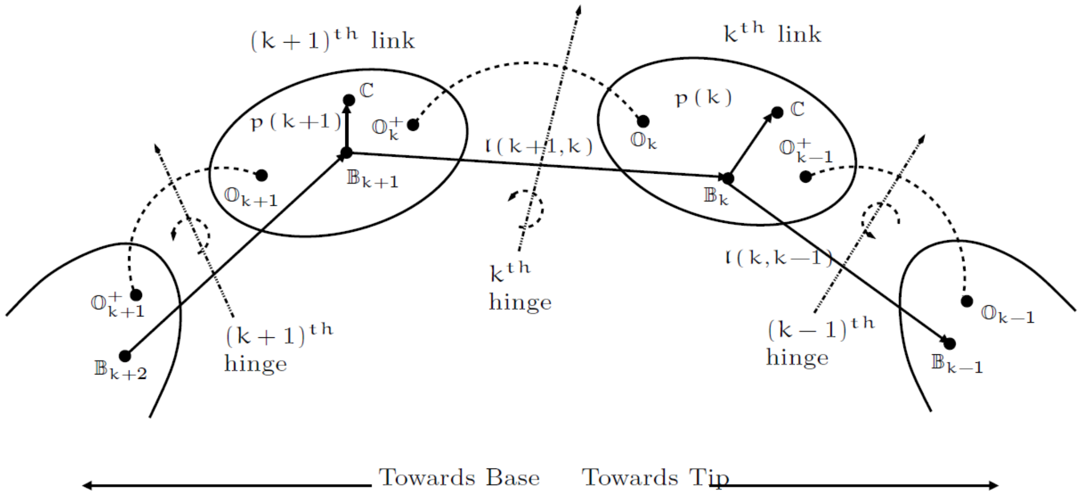


Velocity Recursion

Body frame not at hinge

Body frame not at the hinge

With minor alteration, the recursive body spatial velocity relationship continues to hold.



$$\Delta_{\mathcal{V}}(k) = H^*(k)\beta(k)$$

$$\Delta^{\mathbb{B}}_{\mathcal{V}}(k) \triangleq \phi^*(\mathbb{O}_k, \mathbb{B}_k)\Delta_{\mathcal{V}}(k)$$

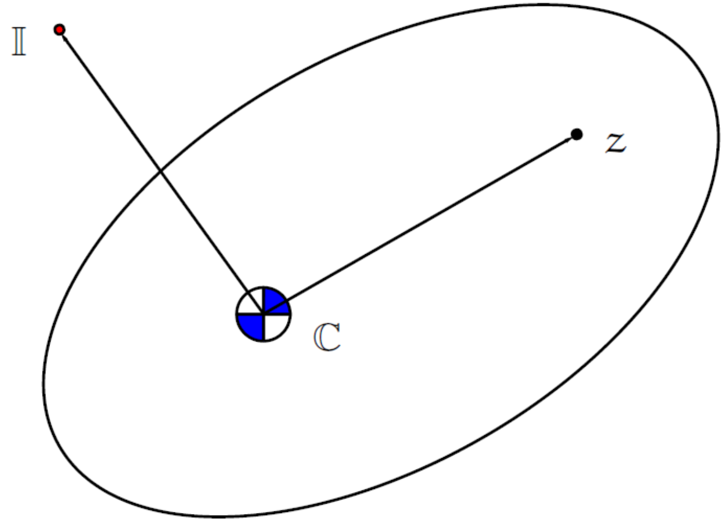
$$H^*_{\mathbb{B}}(k) \triangleq \phi^*(\mathbb{O}_k, \mathbb{B}_k)H^*(k)$$

hinge spatial velocity referenced to the body frame

$$\mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + \Delta^{\mathbb{B}}_{\mathcal{V}}(k)$$
$$= \phi^*(k+1,k)\mathcal{V}(k+1) + H^*_{\mathbb{B}}(k)\beta(k)$$

Inertially referenced body velocities

Can even use inertially referenced velocities for all the bodies – simplifies recursion relationship.



$$\mathcal{V}_{\mathbb{I}}(k) \triangleq \phi^*(\mathbb{O}_k, \mathbb{I})\mathcal{V}(\mathbb{O}_k)$$

$$H^*_{\mathbb{I}}(k) \triangleq \phi^*(\mathbb{O}_k, \mathbb{I})H^*(k)$$

hinge spatial velocity referenced to the body frame

$$\Delta^{\mathbb{I}}_{\mathcal{V}}(k) \triangleq H^*_{\mathbb{I}}(k)\beta(k)$$

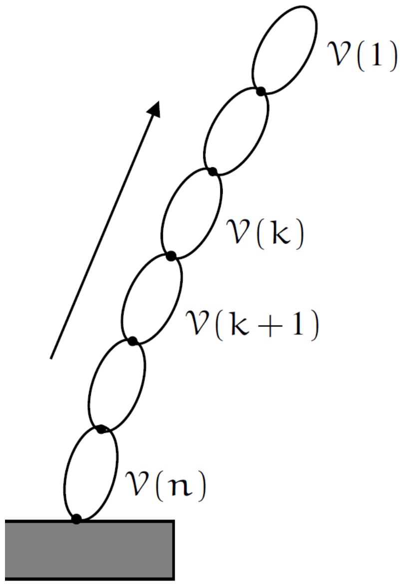
$$\mathcal{V}_{\mathbb{I}}(k) = \mathcal{V}_{\mathbb{I}}(k+1) + \Delta^{\mathbb{I}}_{\mathcal{V}}(k) = \mathcal{V}_{\mathbb{I}}(k+1) + H^*_{\mathbb{I}}(k)\beta(k)$$

rigid body transformation matrix not needed!

Stacked Notation

Stacked vectors

- We are interested in system level properties
- Stack up component quantities into system level vectors



$$\theta \triangleq \text{col}\left\{\theta(k)\right\}_{k=1}^{n} = \begin{bmatrix} \theta(1) \\ \theta(2) \\ \vdots \\ \theta(n) \end{bmatrix} \in \mathcal{R}^{\mathcal{N}}$$

$$\mathcal{V} \triangleq \text{col}\left\{\mathcal{V}(k)\right\}_{k=1}^{n} = \begin{bmatrix} \mathcal{V}(1) \\ \mathcal{V}(2) \\ \vdots \\ \mathcal{V}(n) \end{bmatrix} \in \mathcal{R}^{6n}$$

$$\mathcal{N} \triangleq \sum_{k=1}^{n} r_{\nu}(k)$$

overall dofs

More stacked vectors

- Build up additional system-level stacked vectors

$$\mathcal{V}^+ \triangleq \text{col}\left\{\mathcal{V}^+(k)\right\}_{k=1}^{n} \in \mathcal{R}^{6n}$$

$$\Delta_{\mathcal{V}} \triangleq \text{col}\left\{\Delta_{\mathcal{V}}(k)\right\}_{k=1}^{n} \in \mathcal{R}^{6n}$$

$$\mathcal{V}(k) \stackrel{3.12}{=} \mathcal{V}(\mathbb{O}_k) \stackrel{3.5}{=} \mathcal{V}^+(k) + \Delta_{\mathcal{V}}(k)$$

body-level expression

$\Updownarrow$

$$\mathcal{V} = \mathcal{V}^+ + \Delta_{\mathcal{V}}$$

equivalent system-level expression

First spatial operators

The $\mathcal{E}_\phi$ operator

First operator relating system level spatial velocity stacked vectors

$$\mathcal{V}^+(k) \overset{1.37}{=} \phi^*(\mathbb{B}_{k+1}, \mathbb{O}_k^+)\mathcal{V}(k+1)$$

$$\mathcal{V}^+ = \mathcal{E}_\phi^* \mathcal{V}$$

Rows: *parent body*
Columns: *child body*

$$\mathcal{E}_\phi \triangleq \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0} \end{pmatrix} \in \mathcal{R}^{6n \times 6n}$$

sparse, only one sub-diagonal with inter-body rigid transformation matrix elements

The Joint Map operator $\mathcal{H}$

Operator for relative hinge spatial velocities

$$\Delta_{\mathcal{V}}(k) = H^*(k)\beta(k)$$

$$\Updownarrow$$

$$\Delta_{\mathcal{V}} = \mathcal{H}^*\dot{\theta}$$

Rows: *body*
Columns: *body*

Block-diagonal with joint map matrix elements

$$\mathcal{H} \triangleq \text{diag}\left\{H(k)\right\}_{k=1}^{n} = \begin{pmatrix} H(1) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & H(2) & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & H(n) \end{pmatrix} \in \mathcal{R}^{\mathcal{N}\times 6n}$$

Structural properties of H

- Block-diagonal, and non-square, $\mathcal{R}^{\mathcal{N}\times 6n}$
- The block-diagonal non-zero entries are the transpose of the **configuration dependent** joint map matrices for the body hinges

$$\mathrm{H} = \begin{pmatrix} \mathrm{H}(1) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathrm{H}(2) & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathrm{H}(n) \end{pmatrix}$$

Body spatial velocities expression

Body spatial velocity expression is

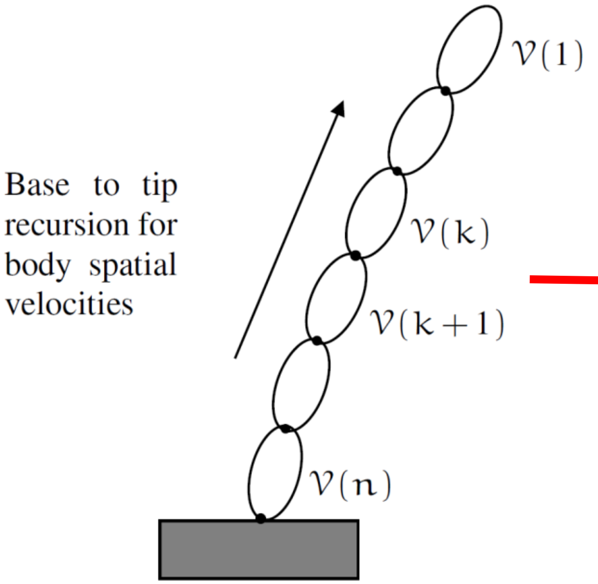
$$\mathcal{V}(k) \overset{3.12}{=} \mathcal{V}(\mathbb{O}_k) \overset{3.5}{=} \mathcal{V}^+(k) + \Delta_{\mathcal{V}}(k)$$

$$\mathcal{V}^+ = \mathcal{E}_\phi^* \mathcal{V}$$

$$\Delta_{\mathcal{V}} = H^* \dot{\theta}$$

$$\mathcal{V} = \mathcal{E}_\phi^* \mathcal{V} + H^* \dot{\theta}$$

Spatial Operator Recap



$$
\begin{cases}
\mathcal{V}(n+1) = 0 \\
\textbf{for } k = n \cdots 1 \\
\quad \mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\beta(k) \\
\textbf{end loop}
\end{cases}
$$

recursive algorithm

$$\mathcal{V} = \mathcal{E}_\phi^* \mathcal{V} + H^* \dot{\theta}$$

Implicit relationship

$$
\mathcal{E}_\phi \triangleq \begin{pmatrix}
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0}
\end{pmatrix}
\qquad
H = \begin{pmatrix}
H(1) & \mathbf{0} & \dots & \mathbf{0} \\
\mathbf{0} & H(2) & \dots & \mathbf{0} \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0} & \mathbf{0} & \dots & H(n)
\end{pmatrix}
$$

Spatial Operators

Operator expression for $\mathcal{V}$

While we have operation expression for the system level spatial velocities, it is implicit!

both sides

$$\mathcal{V} = \mathcal{E}_{\phi}^{*}\mathcal{V} + H^{*}\dot{\theta}$$

$$(\mathbf{I} - \mathcal{E}_{\phi}^{*})\mathcal{V} = H^{*}\dot{\theta}$$

*How to get rid of this to get an **explicit** expression?*

Explicit velocity expression

Nilpotent matrices & inverses

- A square matrix $\mathfrak{U}$ is said to be <u>nilpotent</u> if one of its powers becomes 0, i.e. if for some n

$$\mathfrak{U}^n = \mathbf{0}$$

- For a nilpotent $\mathfrak{U}$, we have

$$(\mathbf{I} - \mathfrak{U})^{-1} = \mathbf{I} + \mathfrak{U} + \mathfrak{U}^2 + \cdots + \mathfrak{U}^{n-1}$$

1-resolvent

*Series expansion terminates after only a **finite** number of terms for nilpotent matrix, hence the 1-resolvent inverse is well defined*

Derivation of nilpotent relative inverse

Define

$$W = \mathbf{I} + U + U^2 + \cdots + U^{n-1}$$

Thus

$$UW = WU = U + U^2 + \cdots + U^n = U + U^2 + \cdots + U^{n-1} = W - \mathbf{I}$$

$= 0$

and so

$$\mathbf{I} = W - UW = (\mathbf{I} - U)W \implies (\mathbf{I} - U)^{-1} = W$$

$\mathcal{E}_\phi$ is nilpotent

$$\mathcal{E}_\phi \triangleq \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0} \end{pmatrix}$$

- Every power of $\mathcal{E}_\phi$ results in a matrix with the sub-diagonal shifted one step lower

$$\mathcal{E}_{\mathbb{A}} = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \\ \cdot & X & \cdot & \cdot \\ \cdot & \cdot & X & \cdot \end{pmatrix}, \quad \mathcal{E}_{\mathbb{A}}^2 = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \\ \cdot & X & \cdot & \cdot \end{pmatrix}, \quad \mathcal{E}_{\mathbb{A}}^3 = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \end{pmatrix}$$

- At the nth power, the result is zero: $\boxed{\mathcal{E}_\phi^n = \mathbf{0}}$
- Hence $\mathcal{E}_\phi$ is **nilpotent**!

Structural properties of $\mathcal{E}_\phi$

- Strictly lower triangular, square, singular and nilpotent,
- Only the first sub-diagonal has nonzero elements
- The non-zero entries are the **configuration dependent** 6x6 inter-link rigid body transformation matrices (configuration dependent)

$$\mathcal{E}_\phi \triangleq \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0} \end{pmatrix}$$

The ϕ spatial operator

$$\mathcal{E}_\phi \triangleq \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0} \end{pmatrix}$$

$\mathcal{E}_\phi$ is **nilpotent** for a tree system, and we can thus define its 1-resolvent

$$\boxed{\phi} \triangleq (\mathbf{I} - \mathcal{E}_\phi)^{-1} = \mathbf{I} + \mathcal{E}_\phi + \mathcal{E}_\phi^2 + \cdots + \mathcal{E}_\phi^{n-1}$$

$$= \begin{pmatrix} \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \phi(2,1) & \mathbf{I} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \phi(n,1) & \phi(n,2) & \dots & \mathbf{I} \end{pmatrix}$$

Lower triangular with inter-body rigid transformation matrix elements

Rows: *parent body*
Columns: *child body*

Structural properties of ϕ

- Lower triangular, square and invertible
- Entirely generated by $\mathcal{E}_\phi$
- Has identity matrices on the main diagonal
- The first sub-diagonal has just the elements of $\mathcal{E}_\phi$
- The other sub-diagonals are powers of $\mathcal{E}_\phi$
- The lower-triangular entries are general, **configuration dependent** 6x6 rigid body transformation matrices

$$\phi = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \phi(2,1) & \mathbf{I} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \phi(n,1) & \phi(n,2) & \dots & \mathbf{I} \end{pmatrix}$$

$$(\mathbf{I} - \mathcal{E}_\phi)^{-1}$$

$$\phi(i,j) = \phi(i,i-1) \cdots \phi(j+1,j) = \begin{pmatrix} \mathbf{I} & \widetilde{l}(i,j) \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$$

Jet Propulsion Laboratory
California Institute of Technology

Explicit operator expression for body spatial velocities

Begin with earlier implicit expression

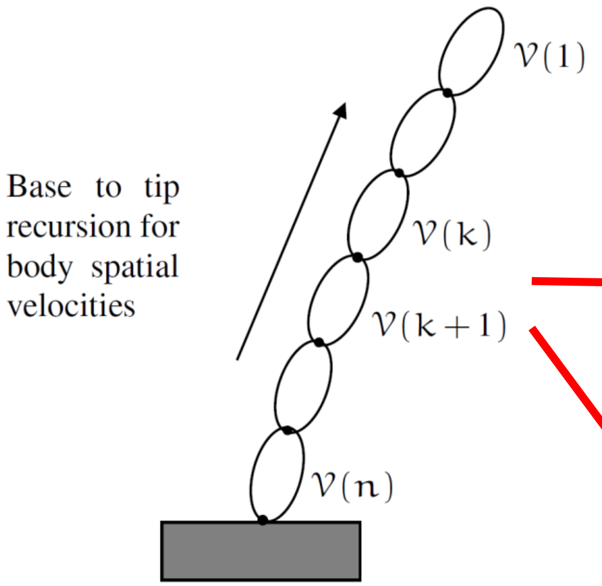
$$(\mathbf{I} - \mathcal{E}_{\phi}^{*})\mathcal{V} = \mathrm{H}^{*}\dot{\theta}$$

$$\mathcal{V} \overset{3.35}{=} (\mathbf{I} - \mathcal{E}_{\phi}^{*})^{-1}\mathrm{H}^{*}\dot{\theta} \overset{3.36}{=} \phi^{*}\mathrm{H}^{*}\dot{\theta}$$

$$\boxed{\mathcal{V} = \phi^{*}\mathrm{H}^{*}\dot{\theta}}$$

***Explicit** operator expression for* $\mathcal{V}$

Operator expression and recursions



$$\begin{cases} \mathcal{V}(n+1) = 0 \\ \textbf{for } k = n \cdots 1 \\ \quad \mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\beta(k) \\ \textbf{end loop} \end{cases}$$

O(N) recursive algorithm

$$\mathcal{V} = \phi^* H^* \dot{\theta}$$

Explicit relationship

The body spatial velocities can be expressed as a spatial operator expression, and computed via an equivalent recursive algorithm

Spatial operator $\tilde{\phi}$

Define the new spatial operator

$$\tilde{\phi} \triangleq \phi - I$$

Same as ϕ, except diagonal elements are now zero matrices

Claim:

$$\tilde{\phi} = \mathcal{E}_{\phi}\phi = \phi\mathcal{E}_{\phi}$$

Derivation:

$$I = (I - \mathcal{E}_{\phi})\phi = \phi - \mathcal{E}_{\phi}\phi \implies \phi - I = \tilde{\phi} = \mathcal{E}_{\phi}\phi$$

Operator expression for $\mathcal{V}^{+}$

Claim: $\mathcal{V}^{+} = \tilde{\phi}^{*} H^{*} \dot{\theta}$

Derivation:

$$\mathcal{V}^{+} \overset{3.15}{=} \mathcal{E}_{\phi}^{*} \mathcal{V}$$

$$\overset{3.39}{=} \mathcal{E}_{\phi}^{*} \phi^{*} H^{*} \dot{\theta} \overset{3.41}{=} \tilde{\phi}^{*} H^{*} \dot{\theta}$$

Operator Expressions
to
O(N) Scatter recursions

Base-to-tips structure-based O(N) scatter recursion

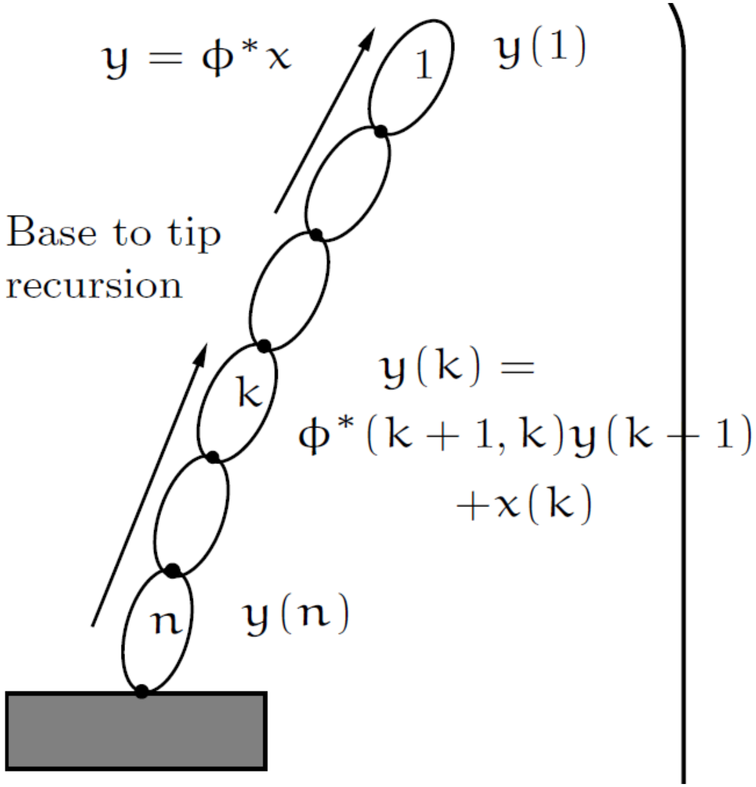
operator transpose/vector product

$$y = \phi^* x$$

- *Applies to any x*
- *Does not require explicit computation of ϕ at all*
- *Only depends on elements of $\mathcal{E}_\phi$*

$$\begin{cases} y(n+1) = 0 \\ \textbf{for } k \quad n \cdots 1 \\ \quad y(k) = \phi^*(k+1,k)y(k+1) + x(k) \\ \textbf{end loop} \end{cases}$$

O(N) structure-based, base-to-tip scatter recursion



Algorithm flow

Derivation of the scatter recursion

Have

$$y = \phi^* x$$

$$\phi = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \phi(2,1) & \mathbf{I} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \phi(n,1) & \phi(n,2) & \dots & \mathbf{I} \end{pmatrix}$$

$$y(k) \overset{3.37}{=} \sum_{j=n}^{k} \phi^*(j,k)x(j)$$

$$\overset{3.38}{=} \phi^*(k+1,k) \underbrace{\sum_{j=n}^{k+1} \phi^*(j,k+1)x(j)}_{y(k+1)} + x(k)$$

$$\overset{3.46}{=} \phi^*(k+1,k)y(k+1) + x(k)$$

Scatter recursion example
Velocity recursion

$$y = \phi^* x$$

$$y = \mathcal{V}$$

$$x = H^* \dot{\theta}$$

$$\mathcal{V} = \phi^* H^* \dot{\theta}$$

$$\begin{cases} \mathcal{V}(n+1) = 0 \\ \textbf{for } k = n \cdots 1 \\ \quad \mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\beta(k) \\ \textbf{end loop} \end{cases}$$

O(N) scatter recursive algorithm

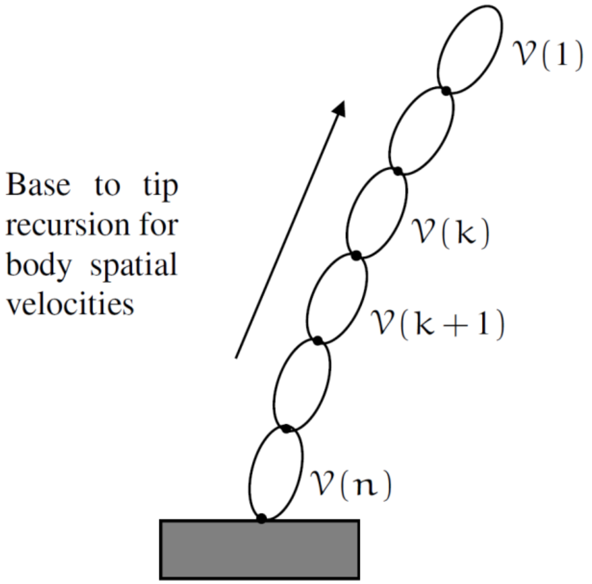
Computational Implications

Some noteworthy observations regarding $y = \phi^* x$

- For computation, this product can be computed by a base-to-tip scatter recursion for any x
- We do not need to compute ϕ^* at all in order to compute the product
- The computation cost in O(N), i.e. it only scales linearly with the number of bodies n
- Any time we encounter a operator expression with such an operator product, we know how to compute it recursively with O(N) cost
- Such mapping is a reflection of underlying structure

The auto-mapping of spatial operator expressions into low-cost recursive algorithms will be a recurring theme

Scatter Recursion Example
Body Velocities Computation



$$\mathcal{V} = \phi^* H^* \dot{\theta}$$

scatter algorithm

$$\begin{cases} \mathcal{V}(n+1) = 0 \\ \textbf{for } k = n \cdots 1 \\ \quad \mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\beta(k) \\ \textbf{end loop} \end{cases}$$

Kane's partial velocities

In the spatial velocities expression

$$\mathcal{V} = \phi^* H^* \dot{\theta}$$

the

$$\phi^* H^*$$

matrix elements are the "partial velocities" from Kane's method.
The key differences with SOA are

- We never need to compute the partial velocities, or either of the operators explicitly
- We keep the operator factors separate and preserve structure - unlike Kane's method where they get mashed up

Operator Expressions
to
O(N) Gather recursions

Tips-to-base structure-based O(N) gather recursion

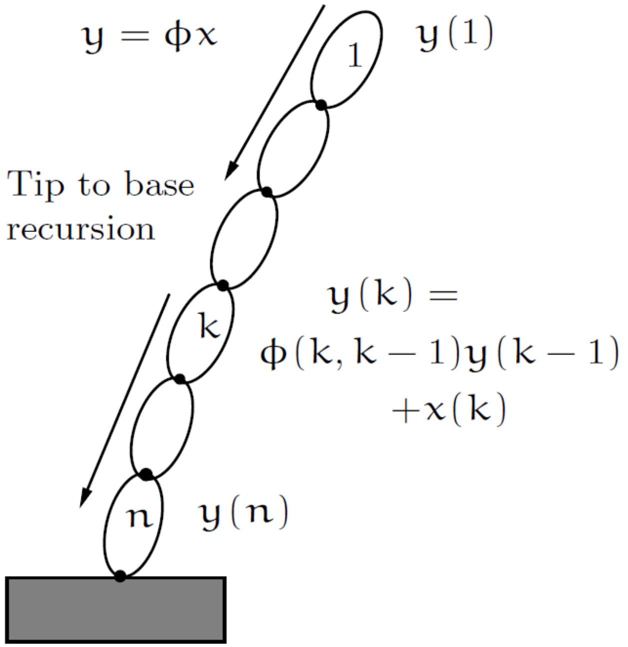
operator/vector product

$$y = \phi x$$

- *Applies to any x*
- *Does not require explicit computation of ϕ at all*
- *Only depends on elements of $\mathcal{E}_\phi$*

$$\begin{cases} y(0) = 0 \\ \textbf{for } k \quad 1 \cdots n \\ \quad y(k) = \phi(k, k-1) y(k-1) + x(k) \\ \textbf{end loop} \end{cases}$$

O(N) structure-based tip-to-base gather recursion



Algorithm flow

Derivation of gather recursion

Have

$$y = \phi x$$

$$\phi = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \phi(2,1) & \mathbf{I} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \phi(n,1) & \phi(n,2) & \dots & \mathbf{I} \end{pmatrix}$$

Thus

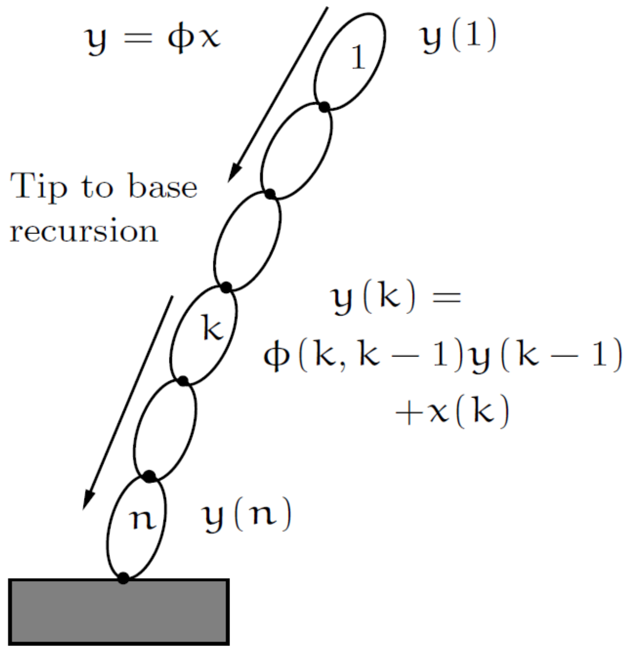
$$y(k) \overset{3.37}{=} \sum_{j=1}^{k} \phi(k,j)x(j)$$

$$\overset{3.38}{=} \phi(k,k-1)\underbrace{\sum_{j=1}^{k-1} \phi(k-1,j)x(j)}_{y(k-1)} + x(k)$$

$$\overset{3.44}{=} \phi(k,k-1)y(k-1) + x(k)$$

Gather recursion example
Spatial forces propagation

We will encounter examples of $y = \phi x$ operator expressions for external spatial forces propagation a little later



Additional O(N) recursions

O(N) scatter recursion for $\tilde{\phi}^* x$

Lets say $\bar{y} \triangleq \tilde{\phi}^* x$

$\mathcal{V}^+ = \tilde{\phi}^* H^* \dot{\theta}$

Example

Leads to O(N) scatter recursion

$$\bar{y}(k) = \phi^*(k+1, k)[\bar{y}(k+1) + x(k+1)]$$

Derivation

$\bar{y} \triangleq \tilde{\phi}^* x = \mathcal{E}_\phi^* y$ where $y = \phi^* x$

Thus $\bar{y}(k) = \phi^*(k+1, k) y(k+1)$

O(N) gather recursion for $\tilde{\phi} x$

Lets say $\bar{y} \triangleq \tilde{\phi} x$

Leads to O(N) gather recursion

$$\bar{y}(k) = \phi(k, k-1)[\bar{y}(k-1) + x(k-1)]$$

Derivation

$$\bar{y} \triangleq \tilde{\phi} x = \mathcal{E}_{\phi} y \quad \text{where} \quad y = \phi x$$

Thus $\bar{y}(k) = \phi(k, k-1) y(k-1)$

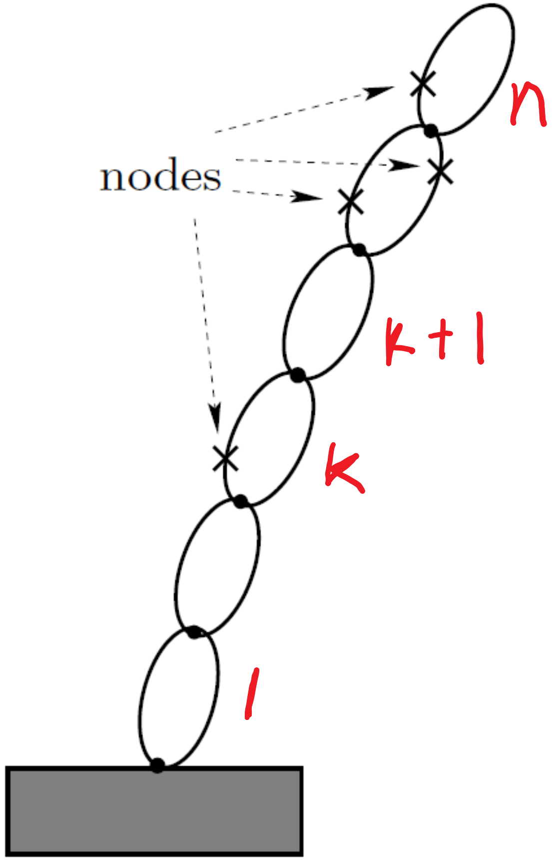
Jet Propulsion Laboratory
California Institute of Technology

Reverse Body Indices

Reversed body indices

Lets say

- We reverse the body indices
- Start with base body index being 1
- Parent index < Child index
- What is the impact?

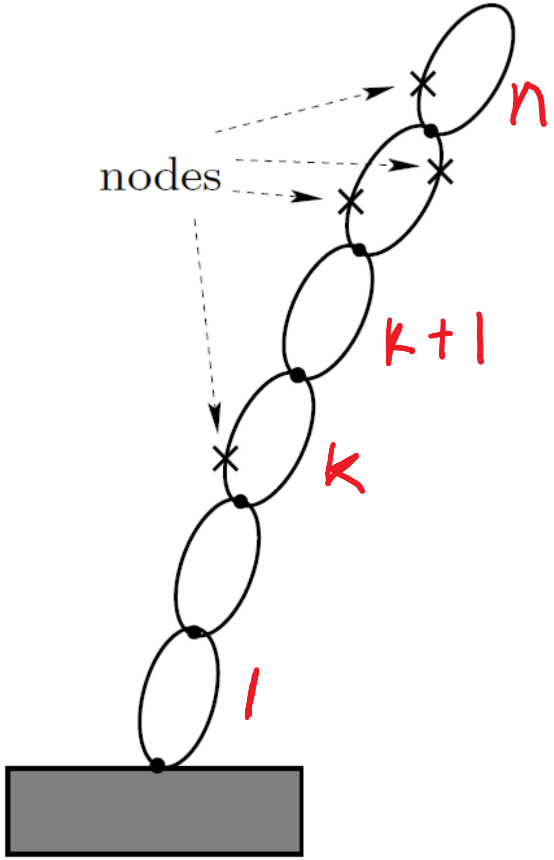


Reversed body indices impact on $\mathcal{E}_\phi$

$$\mathcal{E}_\phi \triangleq \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0} \end{pmatrix}$$

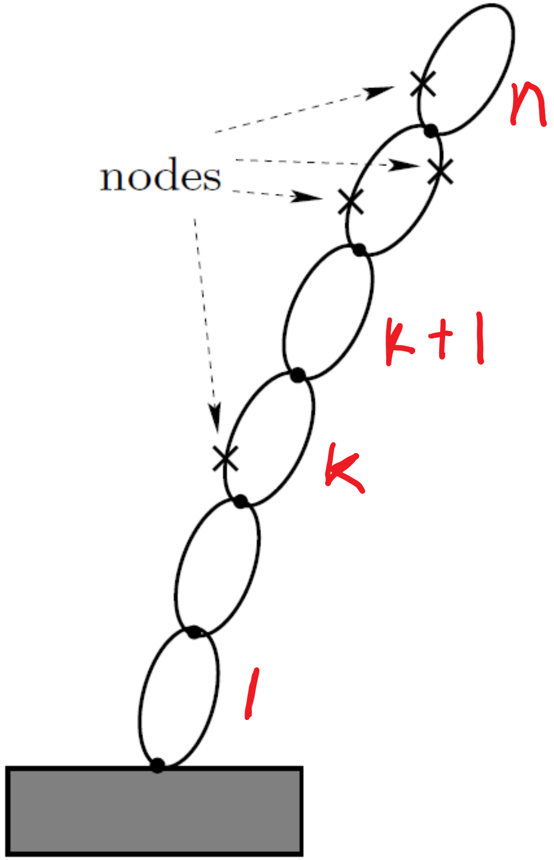
super-diagonal

$$\mathcal{E}_{\phi R} = \begin{pmatrix} \mathbf{0} & \phi(k,k+1) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n-2,n-1) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \phi(n-1,n) \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \end{pmatrix}$$



Reversed body indices impact on $\mathcal{E}_\phi$

$$\mathcal{E}_{\phi R} = \begin{pmatrix} \mathbf{0} & \phi(k,k+1) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n-2,n-1) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \phi(n-1,n) \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \end{pmatrix}$$



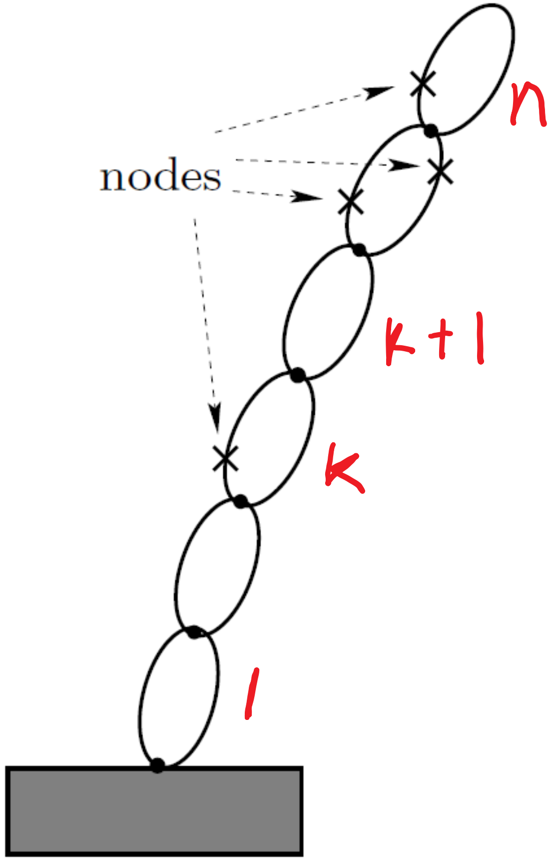
- The lower sub-diagonal shifts to the upper sub-diagonal
- Once again, only parent/child entries are non-zero
- $\mathcal{E}_\phi$ is still nilpotent

Reversed body indices impact on ϕ

- The 1-resolvent of $\mathcal{E}_\phi$ still exists

$$\phi \triangleq (\mathbf{I} - \mathcal{E}_\phi)^{-1}$$

- However, ϕ is now upper-triangular
- The duality with recursive O(N) algorithm continues to hold
  - $y = \phi^* x$ scatter
  - $y = \phi x$ gather



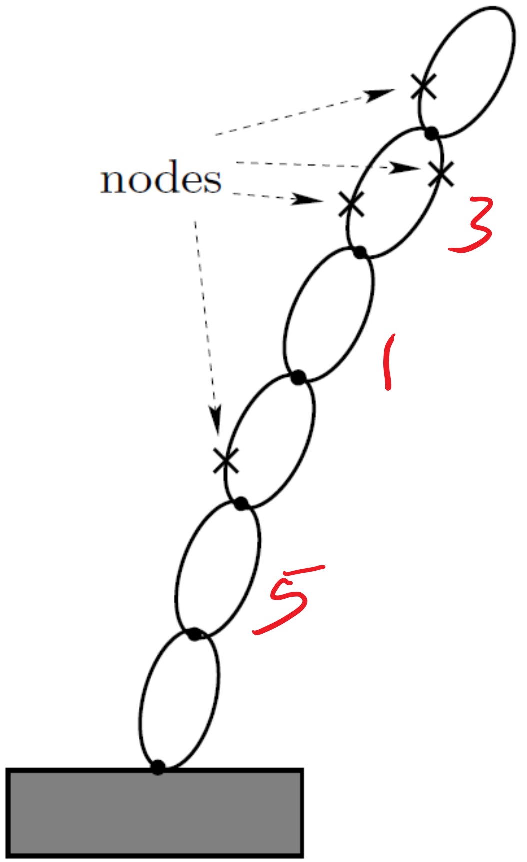
Index numbering has little fundamental impact! For consistency we will stick to our tip-to-base numbering for now

What about randomized indices?

- The 1-resolvent of $\mathcal{E}_\phi$ still exists

$$\phi \triangleq (\mathbf{I} - \mathcal{E}_\phi)^{-1}$$

- However, no longer have triangular structure
- The duality with recursive O(N) algorithm continues to hold
  - $y = \phi^* x$ scatter
  - $y = \phi x$ gather



The operator sparse structure starts to become non-obvious for randomized indices – however it is still there!

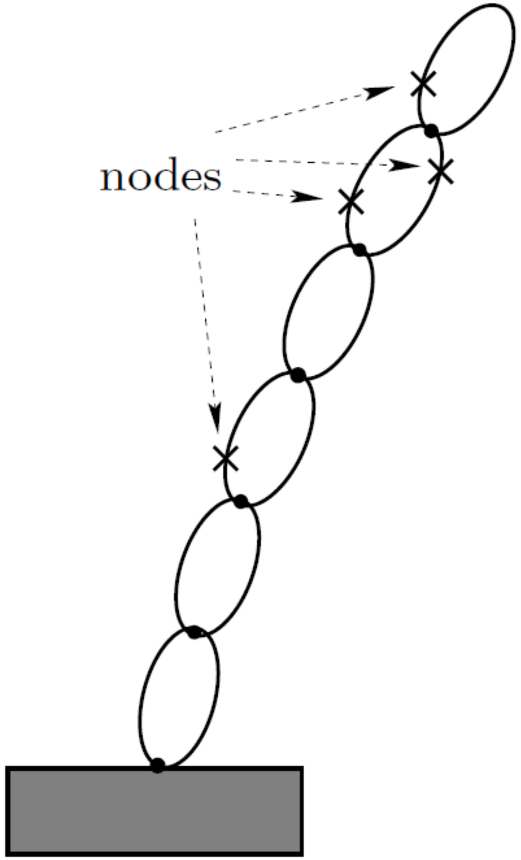
Jet Propulsion Laboratory
California Institute of Technology

Jacobian operator

Nodes on a body

There are typically points of interest on bodies that we will refer to a **nodes**, eg.

- End-effector frame for a robot
- Attachment points for actuators and sensors
- Reference frames for control algorithms



The spatial velocity for a node can be obtained from that of its parent body as follows:

$$\mathcal{V}\left(\mathbb{O}_k^i\right) \overset{1.41}{=} \phi^*(k, \mathbb{O}_k^i)\mathcal{V}(k)$$

Pick-Off Operator $\mathcal{B}$

There are times when we need to narrow attention to the nodes

$$\mathcal{V}_{nd} \triangleq \text{col}\left\{\mathcal{V}_{nd}(k)\right\}_{i=1}^{n} \in \mathcal{R}^{6n_{nd}}$$

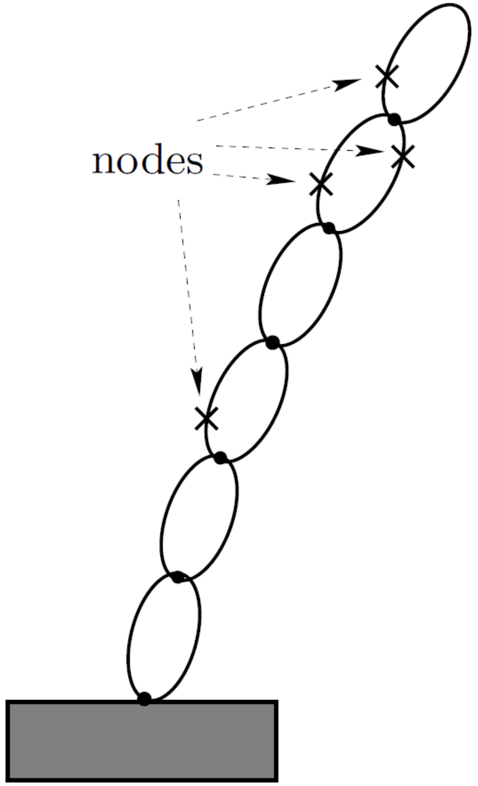
node spatial velocities

$$\mathcal{B} \triangleq \begin{bmatrix} \phi(1, O_1^0) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}$$

pick-off operator

$$\mathcal{V}(\mathbb{O}_k^i) \overset{1.41}{=} \phi^*(k, \mathbb{O}_k^i)\mathcal{V}(k)$$

single node spatial velocity



$$\mathcal{V}_{nd} = \mathcal{B}^*\mathcal{V}$$

mapping from body to node spatial velocities

Jacobian Matrix

Combining

$$\mathcal{V}_{nd} \overset{3.47}{=} \mathcal{B}^* \mathcal{V} \quad \textit{and} \quad \mathcal{V} = \phi^* H^* \dot{\theta}$$

we have

$$\mathcal{V}_{nd} = \mathcal{J}\,\dot{\theta} \quad \text{where} \quad \boxed{\mathcal{J} \triangleq \mathcal{B}^* \phi^* H^*} \in \mathcal{R}^{6n_{nd} \times \mathcal{N}}$$

operator expression for the Jacobian

Jacobian

The <u>Jacobian</u> relates the generalized velocities to the spatial velocity of a node of interest

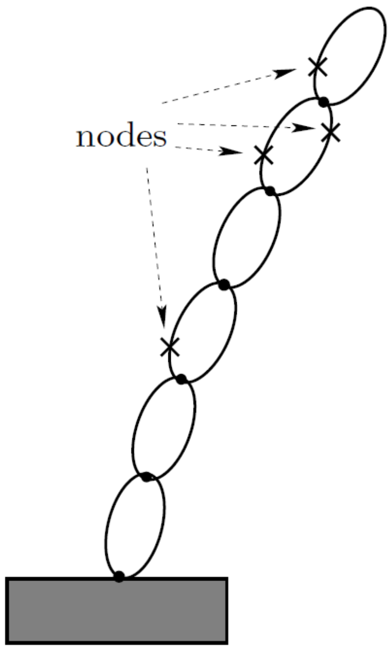
Example: O(N) compensating torque computation

- Lets say external spatial forces (eg. gravity, task object, end-effector forces) are being applied on the system, and we need to apply additional hinge torques to counter these forces
- The required torques are

$$\delta_{\mathcal{T}} \triangleq \mathcal{J}^* \mathfrak{f}_{ext} = H\phi\mathcal{B}\mathfrak{f}_{ext}$$

- Can compute using O(N) gather recursion

$$\begin{cases} x(0) = \mathbf{0} \\ \textbf{for } \boldsymbol{k} \quad \mathbf{1} \cdots \boldsymbol{n} \\ \quad x(k) = \phi(k, k-1)x(k-1) + \sum_i \phi(\mathbb{B}_k, \mathbb{O}_k^i)\mathfrak{f}_{ext}^i(k) \\ \quad \delta_{\mathcal{T}}(k) = H(k)x(k) \\ \textbf{end loop} \end{cases}$$



Transforming SOA operator expressions into recursive algorithms

SOA analysis that exploits mathematical structure of dynamics

*Mapping to **structure based**, fast recursive algorithms*

Dynamics properties → Transformed Expressions → Low-order structure-based algorithms

Dynamics properties
- *General approach*
- *Concise*
- *Rich vocabulary*

Transformed Expressions
- *Exploit structure*
- *Get new insights*
- *Solve new problems*

Low-order structure-based algorithms
- *Faster*
- *More robust*

"Structure-based": Because the pattern of the recursive algorithms is entirely driven by the underlying multibody topology.

Summary

- Discussed minimal coordinate kinematics model of a rigid body serial-chain
- Introduced stacked notation
- Introduced some spatial operators
- Discussed duality between operator expressions and O(N) recursive computations:
  - $y = \phi^* x$: base-to-tip O(N) scatter recursion
  - $y = \phi x$: tip-to-base O(N) gather recursion
- Introduced Jacobian and its operator expression

SOA Foundations Track Topics (serial-chain rigid body systems)

1. **Spatial (6D) notation** – spatial velocities, forces, inertias; spatial cross-product, rigid body transformations & properties; parallel axis theorem
2. **Single rigid body dynamics** – equations of motion about arbitrary frame using spatial notation
3. **Serial-chain kinematics** – minimal coordinate formulation, hinges, velocity recursions, Jacobians; first spatial operators; O(N) scatter and gather recursions
4. **Serial-chain dynamics** – equations of motion using spatial operators; Newton–Euler mass matrix factorization; O(N) inverse dynamics; composite rigid body inertia; forward Lyapunov equation; mass matrix decomposition; mass matrix computation; alternative inverse dynamics
5. **Articulated body inertia -** Concept and definition; Riccati equation; alternative force decompositions
6. **Mass matrix factorization and inversion** – spatial operator identities; Innovations factorization of the mass matrix; Inversion of the mass matrix
7. **Recursive forward dynamics** – O(N) recursive forward dynamics algorithm; including gravity and external forces; inter-body forces identity