Module 13

Fixed and Free Degrees of Freedom

- What happens when boundary conditions are applied?

Goal

- Understand fixed and free degrees of freedom
- Adapt our method to incorporate boundary conditions

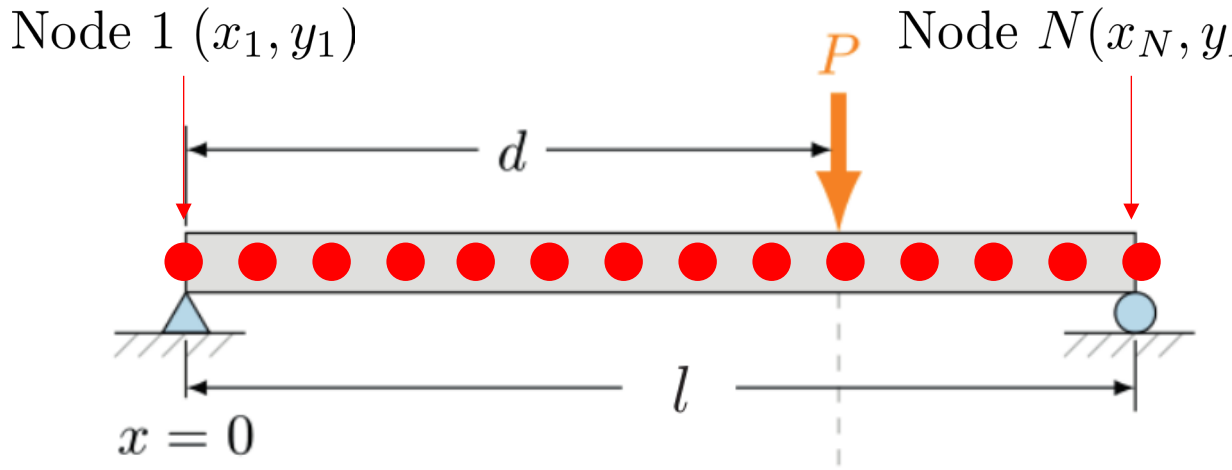
Resources

- Read *"Chapter 4: Conservative force and potential energy"* of Course Notes

Boundary conditions

DOF Vector with size $2N$:

$$\mathbf{q} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \cdots \\ \cdots \\ x_k \\ y_k \\ \cdots \\ \cdots \\ x_N \\ y_N \end{bmatrix}$$



Fixed DOF:

$x_1 = 0$

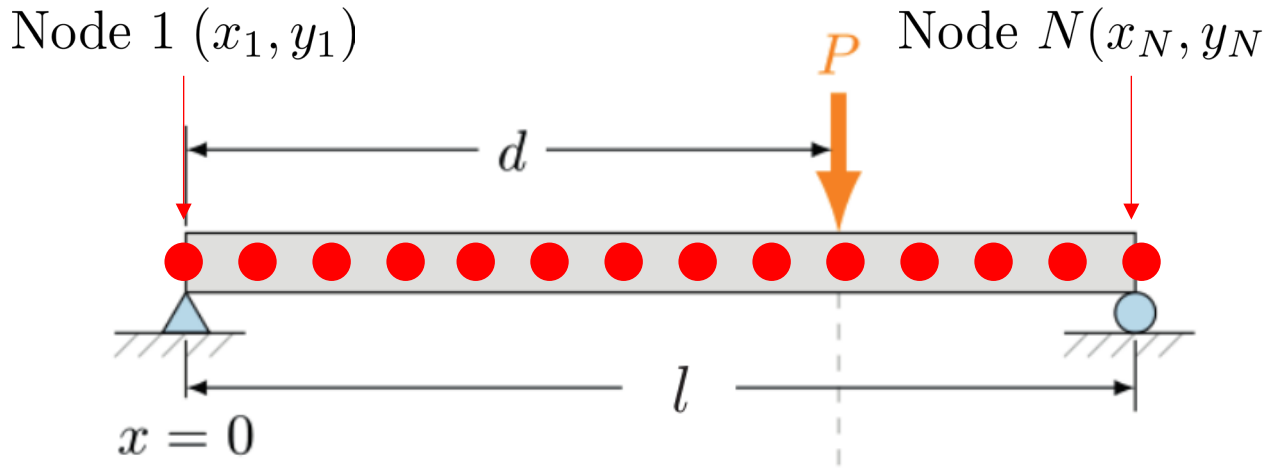
$y_1 = 0$

$y_N = 0$

Fixed and Free Degrees of Freedom

DOF Vector with size $2N$:

$$\mathbf{q} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}_{\mathbf{q}_{\text{fixed}}} \begin{bmatrix} x_2 \\ y_2 \\ \cdots \\ \cdots \\ x_k \\ y_k \\ \cdots \\ \cdots \\ x_N \end{bmatrix}_{\mathbf{q}_{\text{free}}} \begin{bmatrix} y_N \end{bmatrix}_{\mathbf{q}_{\text{fixed}}}$$



Fixed and Free Degrees of Freedom

STRUCTURES-COMPUTER INTERACTION LABORATORY

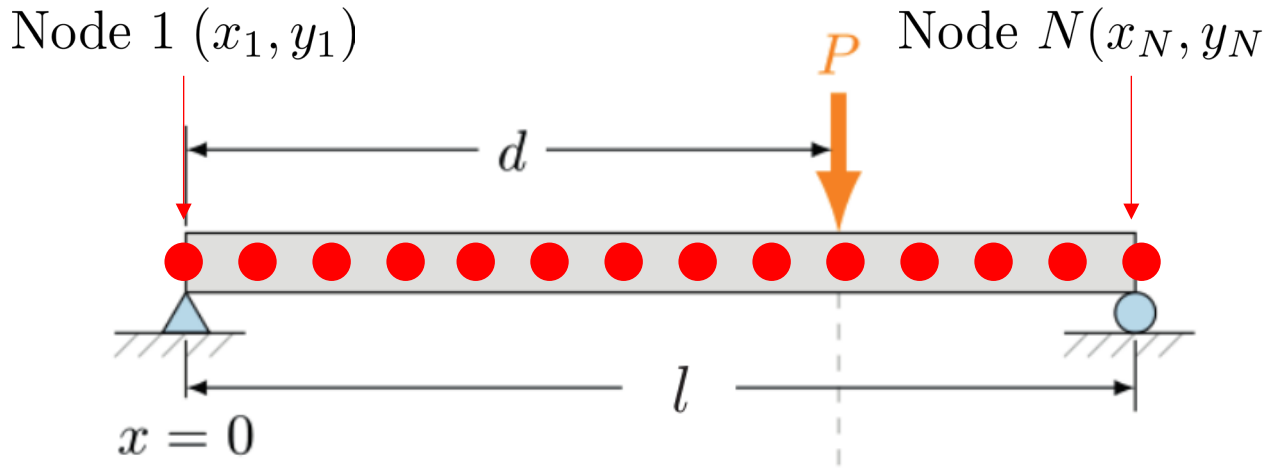
DOF Vector with size $2N$:

$$\mathbf{q} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}_{\mathbf{q}_{\text{fixed}}} \begin{bmatrix} x_2 \\ y_2 \\ \cdots \\ \cdots \\ x_k \\ y_k \\ \cdots \\ \cdots \\ x_N \end{bmatrix}_{\mathbf{q}_{\text{free}}} \begin{bmatrix} y_N \end{bmatrix}_{\mathbf{q}_{\text{fixed}}}$$

Equations of motion are not applicable to fixed DOFs. Instead, we have the following equations:

Fixed DOF:

$$x_1 = 0$$
$$y_1 = 0$$
$$y_N = 0$$



Fixed index: $[1, 2, 2N]$

Free index: $[3, 4, \ldots, 2N - 1]$

We will need this later.

Programming Implementation

$$[\mathbf{q}(t_{k+1}), \dot{\mathbf{q}}(t_{k+1})] = \text{SimulationLoop}(\ [\mathbf{q}(t_k), \dot{\mathbf{q}}(t_k)]\ )$$

When all DOFs are free

Compute $\mathbf{f} = \mathtt{f}(\mathbf{q}(t_{k+1}))$
Compute $\mathbb{J} = \mathtt{J}(\mathbf{q}(t_{k+1}))$

$\Delta\mathbf{q} = \mathbb{J} \backslash \mathbf{f}$

New estimate $\mathbf{q} = -\Delta\mathbf{q} + \mathbf{q}$

When some DOFs are fixed

Compute $\mathbf{f} = \mathtt{f}(\mathbf{q}(t_{k+1}))$
Compute $\mathbb{J} = \mathtt{J}(\mathbf{q}(t_{k+1}))$

$\mathbf{f}_{\text{free}} = \mathbf{f}(\text{free index})$
$\mathbb{J}_{\text{free}} = \mathbb{J}(\text{free index}, \text{free index})$

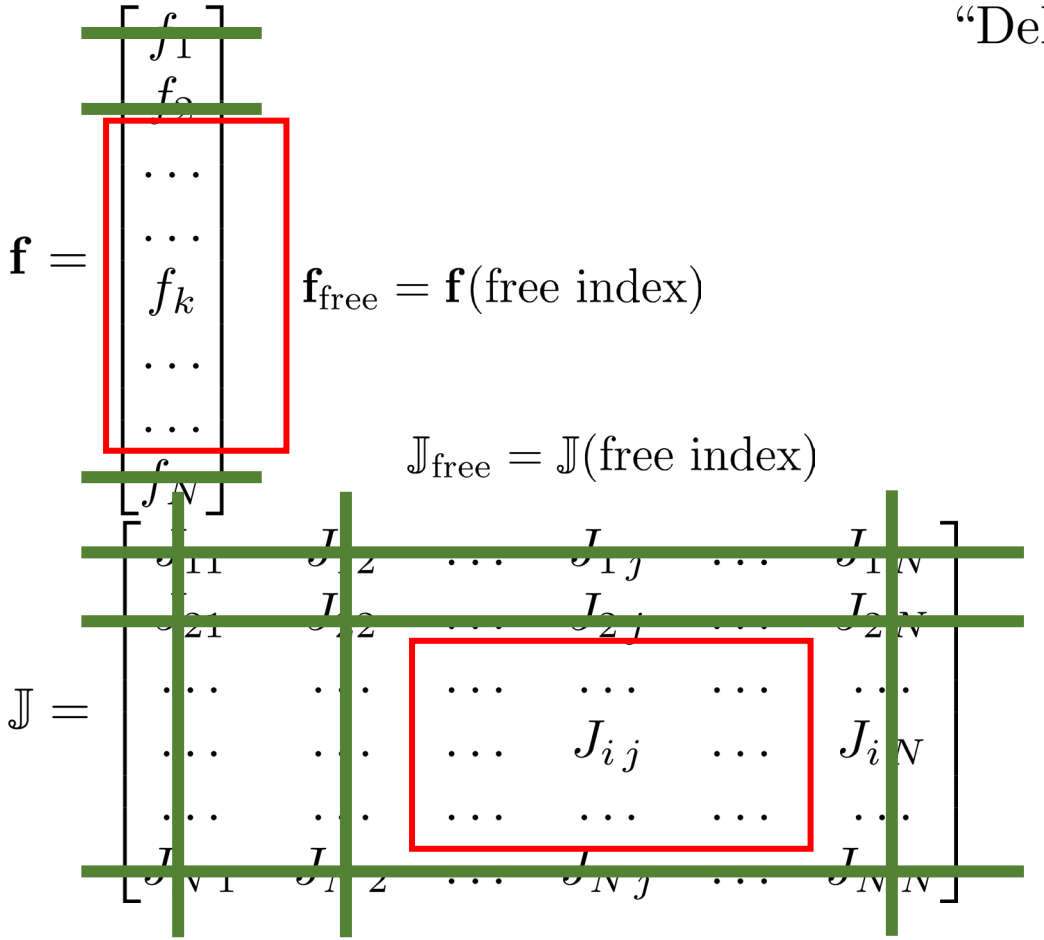
$\Delta\mathbf{q}_{\text{free}} = \mathbb{J}_{\text{free}} \backslash \mathbf{f}_{\text{free}}$

New estimate $\mathbf{q}_{\text{free}} = -\Delta\mathbf{q}_{\text{free}} + \mathbf{q}_{\text{free}}$

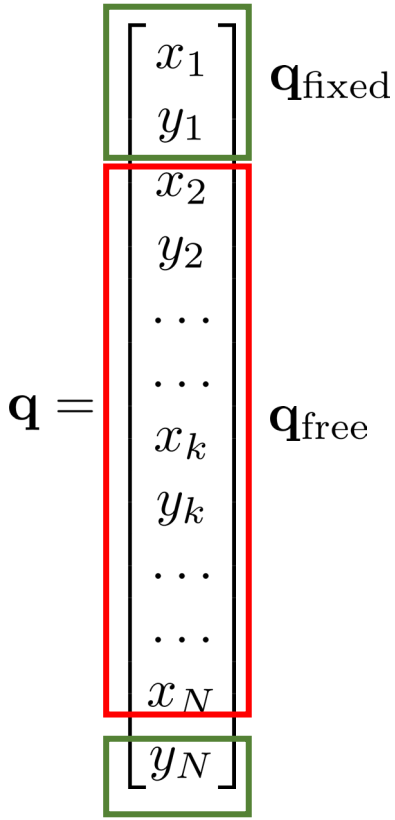
New estimate $\mathbf{q}(\text{free index}) = \mathbf{q}_{\text{free}}$

New estimate $\mathbf{q}(\text{fixed index}) = \mathbf{q}_{\text{fixed}}$

Programming Implementation



"Delete" the rows and columns corresponding to fixed index



Take-away Message

- Don’t be afraid of fixed degrees of freedom
  - More fixed degrees of freedom mean faster computation time due to smaller linear system

STRUCTURES-COMPUTER
INTERACTION LABORATORY

Module 14

Starter on Beam Bending

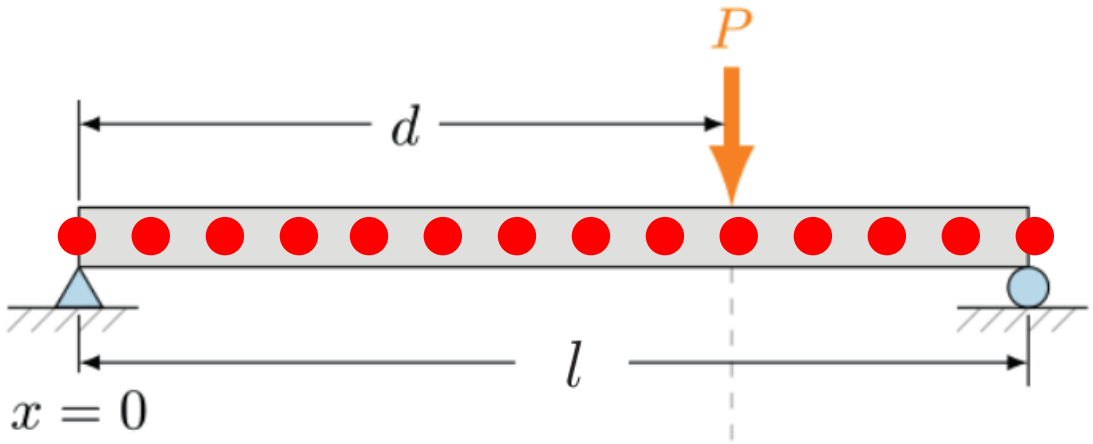
- Example on beam bending
- Comparison with Euler-Bernoulli beam theory

Goal

- Setup a problem with an Euler-Bernoulli type beam bending problem
- Understand fixed and free degrees of freedom
- Compare discrete simulation with Euler-Bernoulli beam theory

Resources

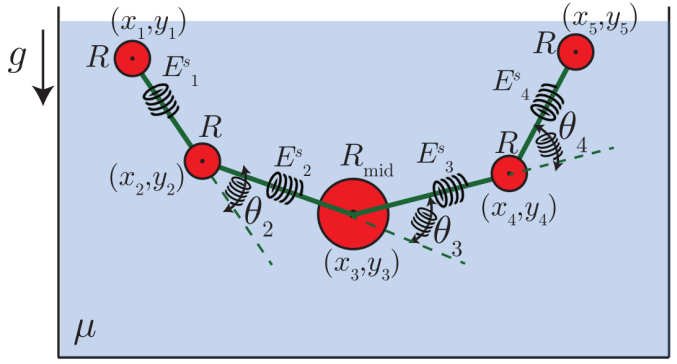
- Read *"Chapter 4: Conservative force and potential energy"* of Course Notes

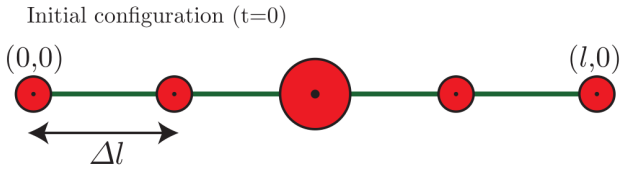


Equations of Motion (EOM)

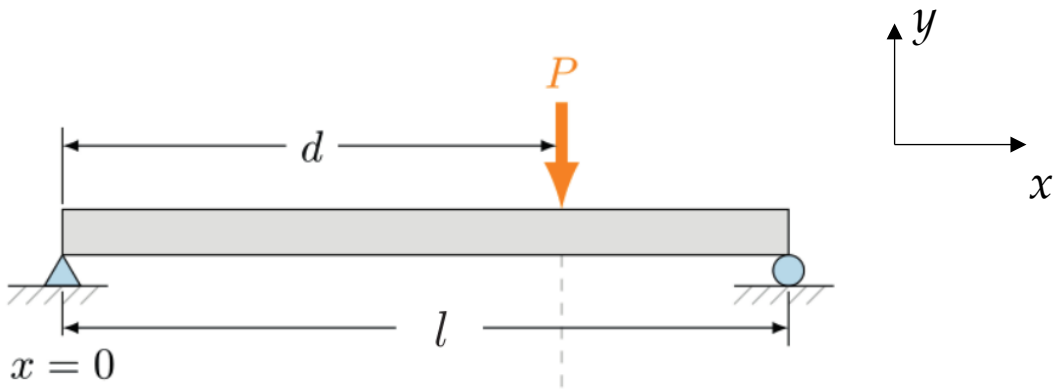
$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} - \mathbf{W} = \mathbf{0}$$

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{P} = \mathbf{0}$$





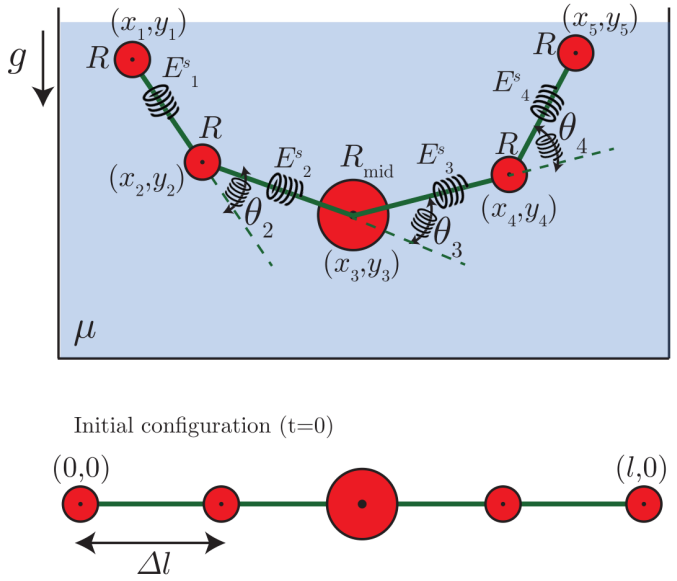
Old Problem



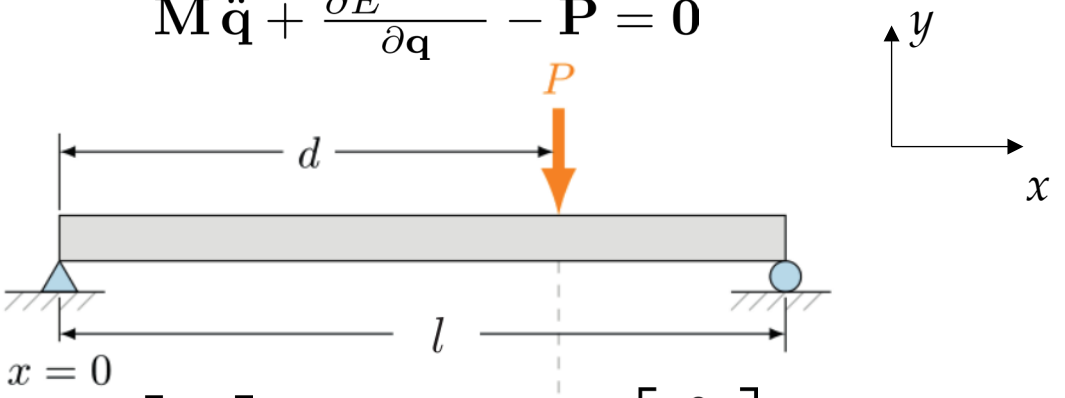
New Problem

Equations of Motion (EOM)

$$\mathbf{M}\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} - \mathbf{W} = \mathbf{0}$$



$$\mathbf{M}\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{P} = \mathbf{0}$$



$$\mathbf{q} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \cdots \\ \cdots \\ x_k \\ y_k \\ \cdots \\ \cdots \\ x_N \\ y_N \end{bmatrix} \qquad \mathbf{P} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \cdots \\ \cdots \\ 0 \\ -P \\ \cdots \\ \cdots \\ 0 \\ 0 \end{bmatrix}$$

Programming Implementation

$$[\mathbf{q}(t_{k+1}), \dot{\mathbf{q}}(t_{k+1})] = \text{SimulationLoop}(\ [\mathbf{q}(t_k), \dot{\mathbf{q}}(t_k)]\ )$$

Old Problem

Compute $\mathbf{f} = \mathtt{f}(\mathbf{q}(t_{k+1}))$
Compute $\mathbb{J} = \mathtt{J}(\mathbf{q}(t_{k+1}))$

$\Delta\mathbf{q} = \mathbb{J} \backslash \mathbf{f}$

New estimate $\mathbf{q} = \Delta\mathbf{q} + \mathbf{q}$

New Problem

Compute $\mathbf{f} = \mathtt{f}(\mathbf{q}(t_{k+1}))$
Compute $\mathbb{J} = \mathtt{J}(\mathbf{q}(t_{k+1}))$

$\mathbf{f}_{\text{free}} = \mathbf{f}(\text{free index})$
$\mathbb{J}_{\text{free}} = \mathbb{J}(\text{free index}, \text{free index})$

$\Delta\mathbf{q}_{\text{free}} = \mathbb{J}_{\text{free}} \backslash \mathbf{f}_{\text{free}}$

New estimate $\mathbf{q}_{\text{free}} = \Delta\mathbf{q}_{\text{free}} + \mathbf{q}_{\text{free}}$

New estimate $\mathbf{q}(\text{free index}) = \mathbf{q}_{\text{free}}$

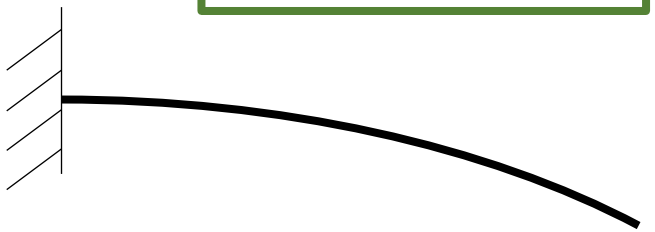
New estimate $\mathbf{q}(\text{fixed index}) = \mathbf{q}_{\text{fixed}}$
//Calculate error based on $\mathbf{f}_{\text{free}}$

Comparison with Euler-Bernoulli Beam Theory

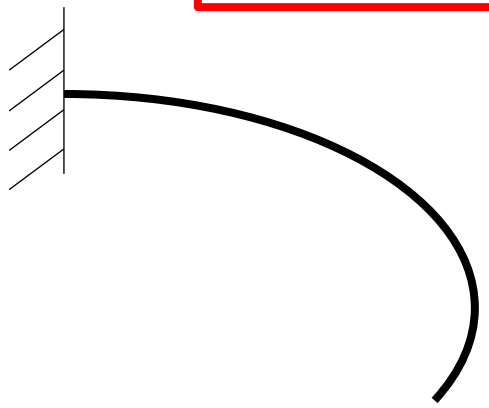
- Euler-Bernoulli beam theory is *geometrically linear*
- Only applicable when the slope is small

Curvature of a function $y = y(x)$ is $\kappa = \frac{y''}{(1+y'^2)^{3/2}} \approx y''$

Linear



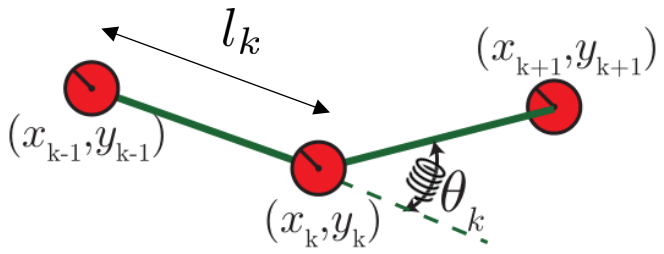
Nonlinear



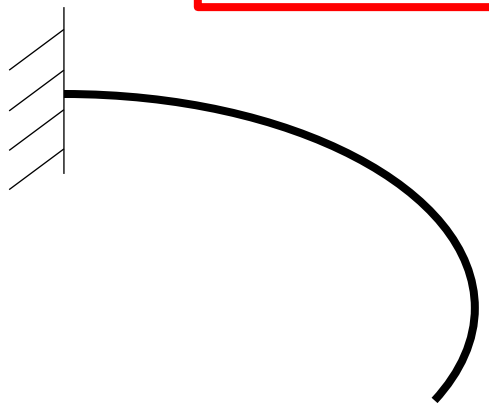
Comparison with Euler-Bernoulli Beam Theory

- Our discrete simulation is *geometrically nonlinear*
- It can handle "arbitrary" slopes

Curvature at node k is
$\kappa = 2\tan\left(\frac{\theta_k}{2}\right)/l_k$



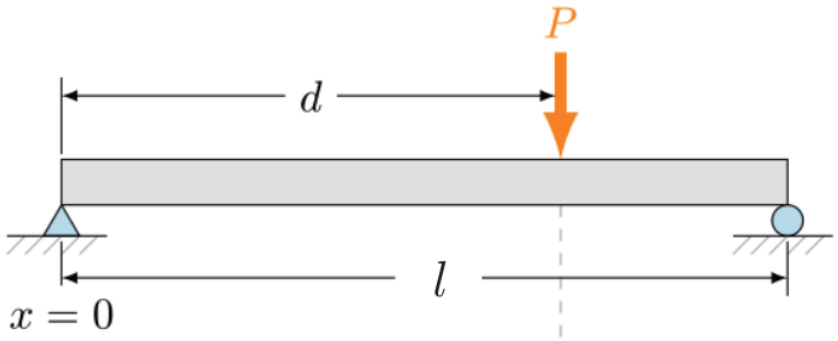
Nonlinear

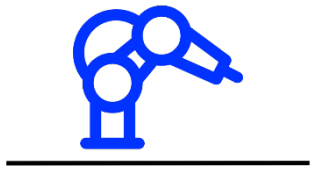


- More details in "A primer on the kinematics of discrete elastic rods" by *MK Jawed, A Novelia, OM O'Reilly*
- Available for free on the course website

Assignments

- Complete **Assignment 3** in **Chapter 4**: *Generalized case of elastic beam falling in viscous flow*





Module 15

Discrete Elastic Energies of a Beam

- Continuous and Discrete Elastic Energies of a Beam
- Natural Curvature

Goal

- Differences between curvature and “discrete integrated” curvature
- Natural curvature
- Discrete representation of the elastic (bending and stretching) energies in a beam

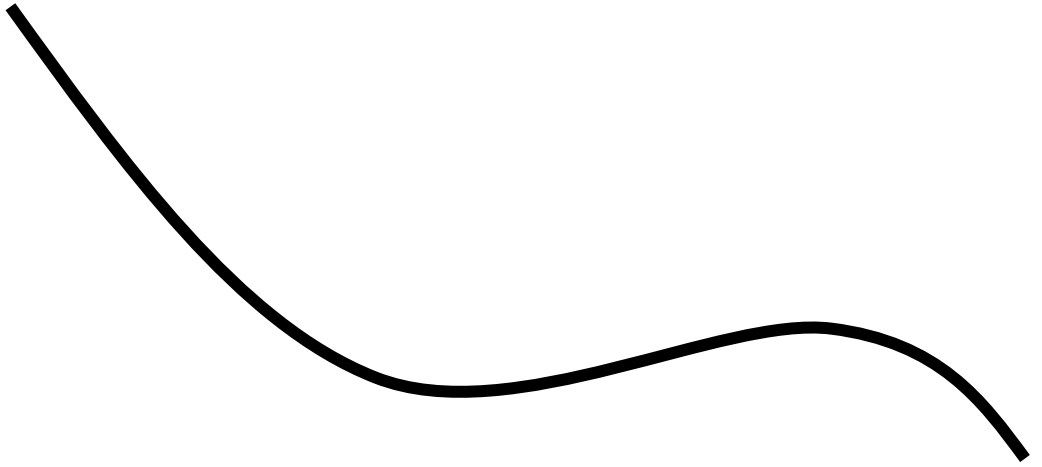
Resources

- Read *"Chapter 5: Discrete simulation of an elastic beam"* of Course Notes
- Recall "*Module: Elastic Beam in Viscous Flow*" and "*Module: Starter on Beam Bending*"

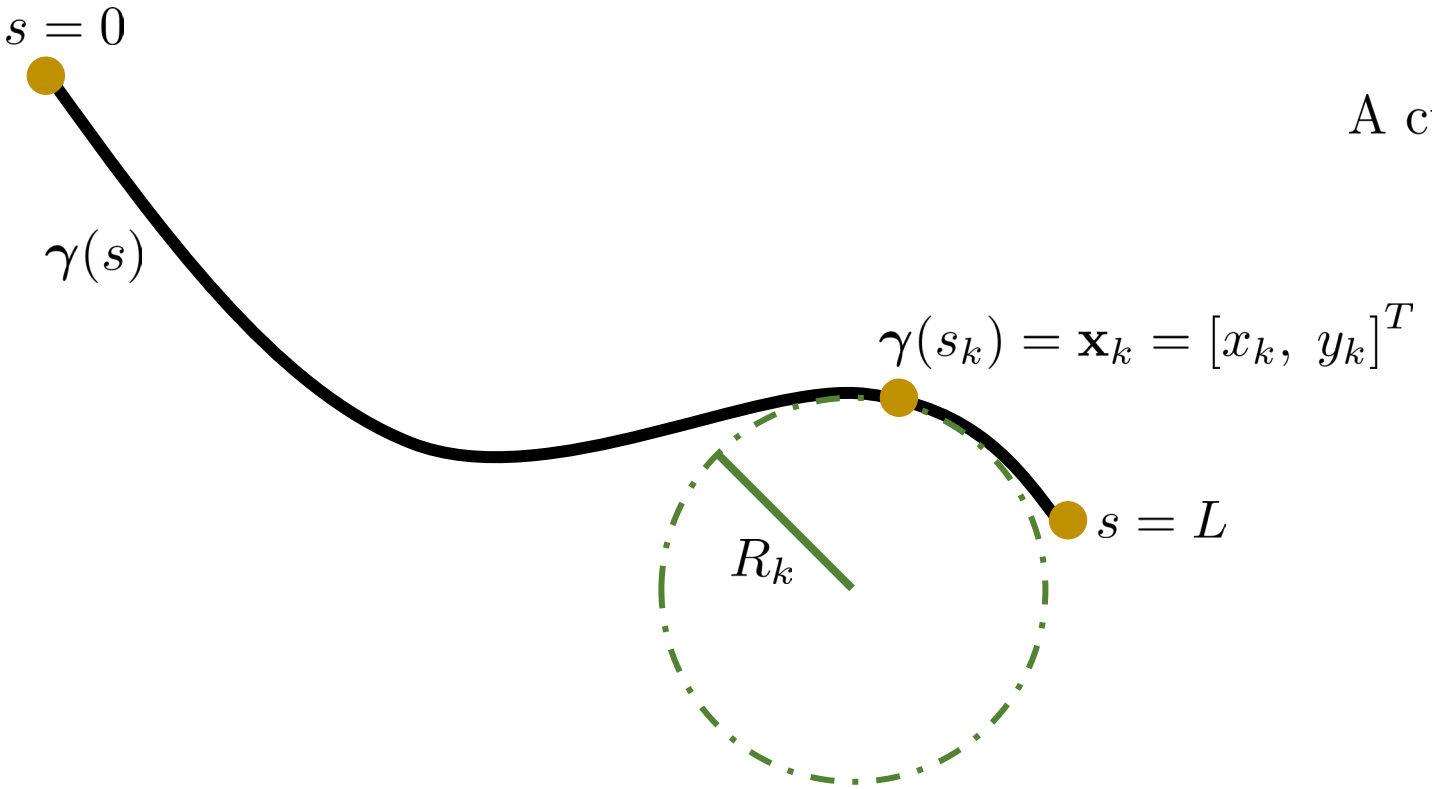
Elastic Energies of a Beam

STRUCTURES-COMPUTER INTERACTION LABORATORY

- A beam has two types of elastic energies:
  - Bending energy
  - Stretching energy (often neglected)



Curvature

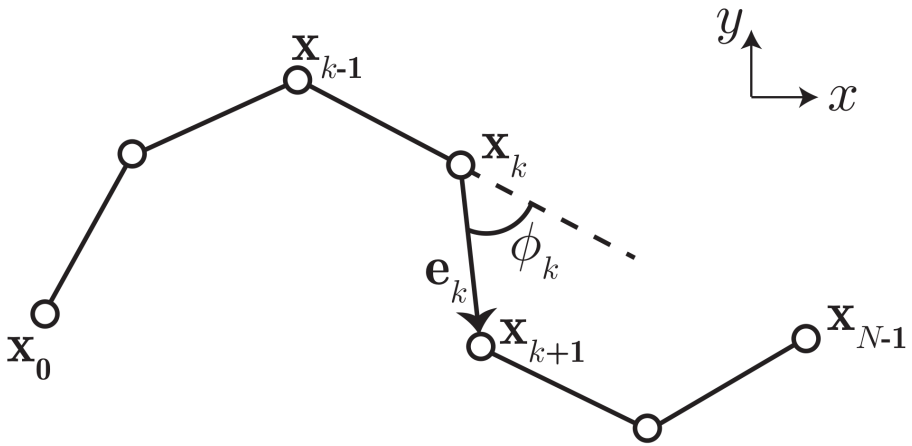


A curve γ with an arc-length parameter s

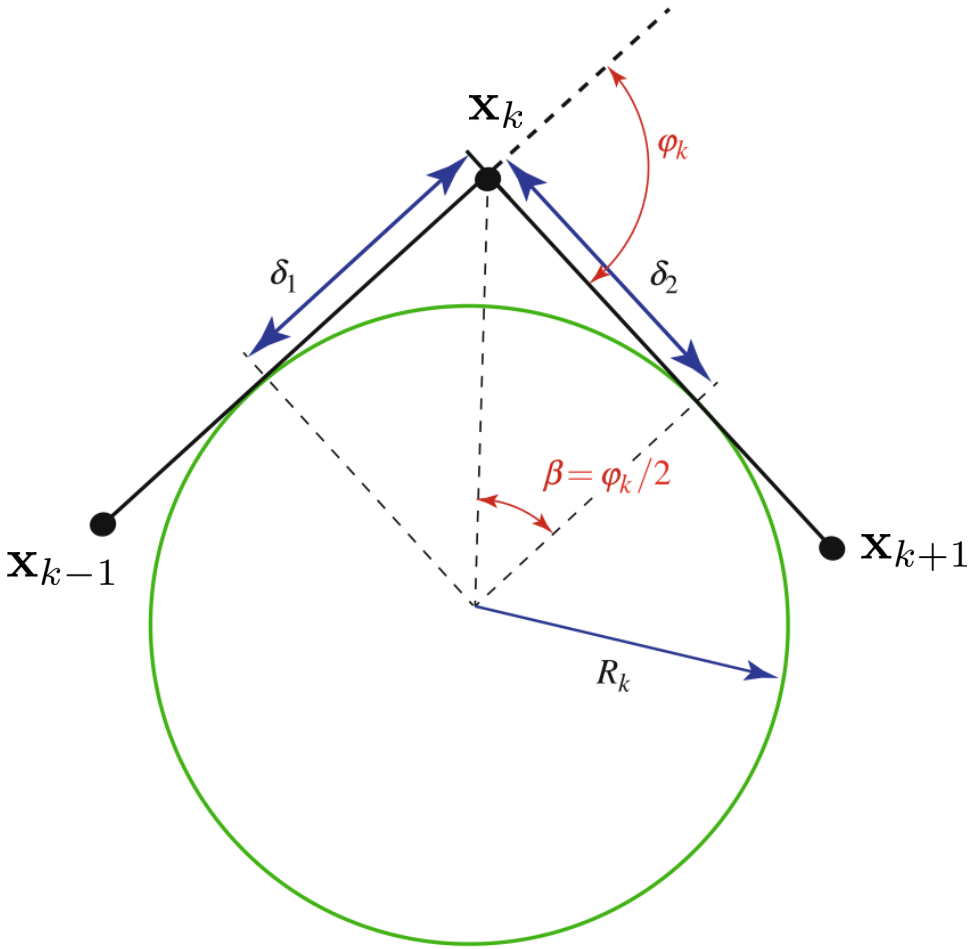
$$\text{Curvature, } \kappa_k = \frac{1}{R_k}$$

$$\gamma''(s_k) \equiv \frac{\partial^2 \gamma}{\partial s^2}|_{s=s_k} = \frac{1}{R_k}$$

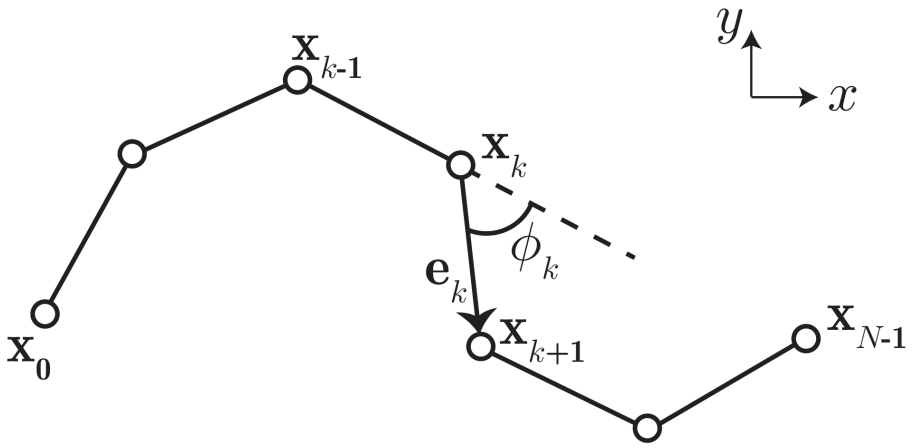
Discrete Integrated Curvature



$R_k = ?$



Discrete Integrated Curvature

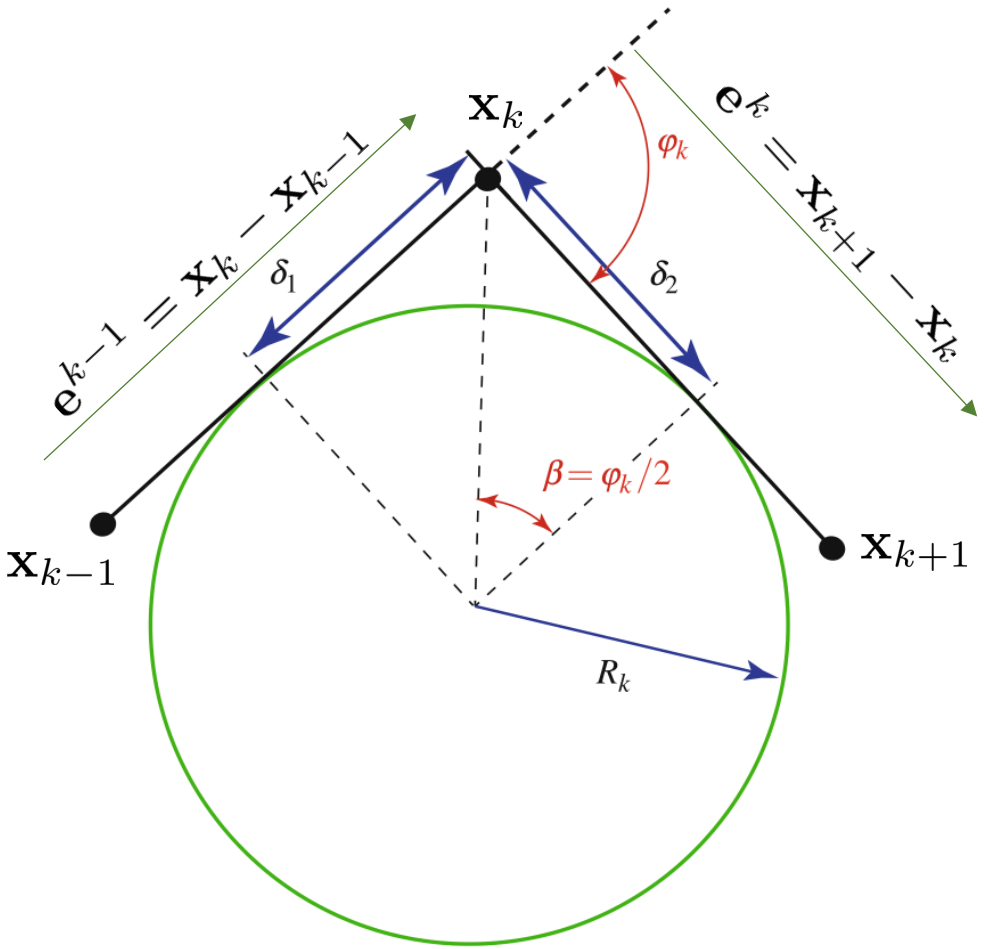


$$\delta_1 = \delta_2 \equiv \frac{dl}{2}$$

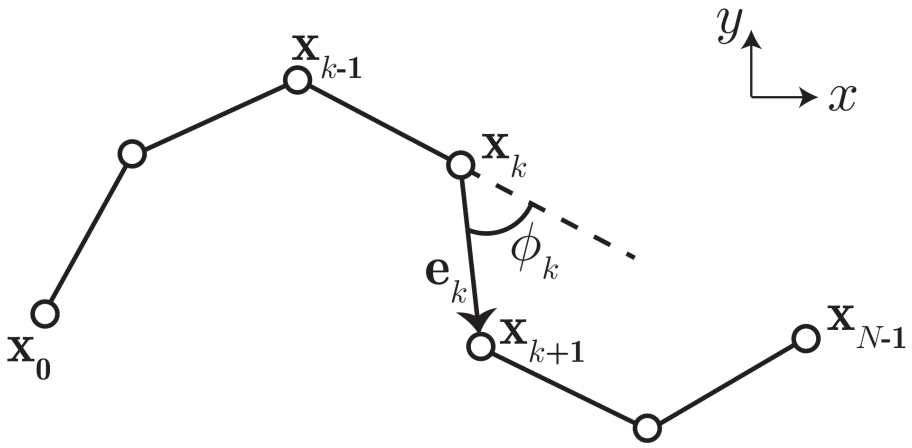
Simple geometry: $R_k = \frac{dl}{2\tan(\phi_k/2)}$

$$\phi_k = \tan^{-1}\frac{(\mathbf{e}^{k-1}\times\mathbf{e}^k)\cdot\hat{e}_z}{\mathbf{e}^{k-1}\cdot\mathbf{e}^k}$$

Discrete integrated curvature: $\kappa_k \equiv 2\tan(\phi_k/2)$
Standard definition of curvature $= \frac{\kappa_k}{dl} = \frac{1}{R_k}$

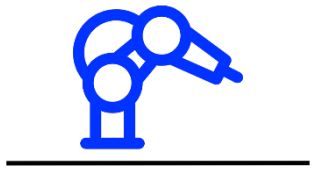


Summary: Discrete Integrated Curvature



We learned how to compute the curvature along the beam from the degrees of freedom (DOFs) x_k, y_k.

Notation:
Subscript indicates node-based quantities, e.g. κ_k
Superscript indicates edge-based quantities, e.g. $\mathbf{e}^k$



Module 16

Discrete Elastic Beams

- Continuous and Discrete Equations of Motion
- Programming Implementation

Goal

- Equations of motion for an elastic beam
- Programming implementation

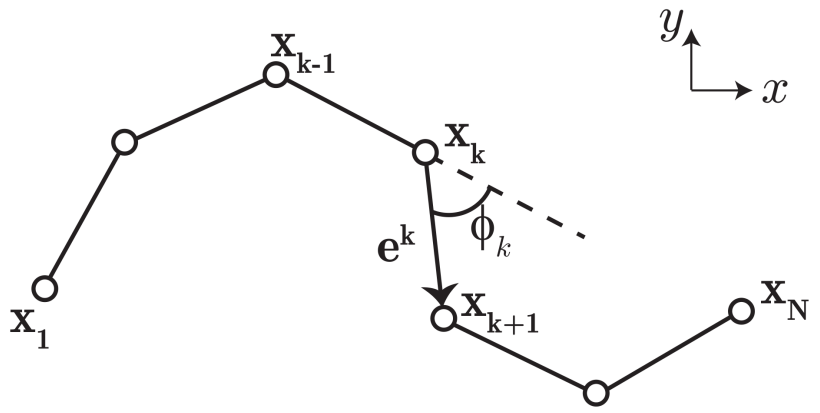
Resources

- Read *"Chapter 5: Discrete simulation of an elastic beam"* of Course Notes
- Recall "*Module: Elastic Beam in Viscous Flow*" and "*Module: Starter on Beam Bending*"
  - Only new concept is *natural curvature*

Continuous Equations of Motion (EOM)

Degrees of Freedom (DOF) vector,

$$\mathbf{q} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \cdots \\ \cdots \\ x_k \\ y_k \\ \cdots \\ \cdots \\ x_N \\ y_N \end{bmatrix}$$



$2N$ equations of motion for N nodes:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{F}_{\text{external}} = \mathbf{0}$$

$$\Longrightarrow \mathbf{M}\,\ddot{\mathbf{q}} + \boxed{\frac{\partial E_{\text{b}}}{\partial \mathbf{q}} + \frac{\partial E_{\text{s}}}{\partial \mathbf{q}}} - \mathbf{F}_{\text{external}} = \mathbf{0}$$

Elastic Energy = Bending + Stretching

Discrete Equations of Motion (EOM)

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E_{\mathrm{b}}}{\partial \mathbf{q}} + \frac{\partial E_{\mathrm{s}}}{\partial \mathbf{q}} - \mathbf{F}_{\mathrm{external}} = \mathbf{0}$$

$$f_i \equiv \frac{m_i}{\Delta t}\left[\frac{q_i(t_{k+1}) - q_i(t_k)}{\Delta t} - \dot{q}_i(t_k)\right] + \frac{\partial E_b}{\partial q_i} + \frac{\partial E_s}{\partial q_i} - F_{i,\mathrm{external}} = 0$$

$$\mathbb{J}_{ij} = \frac{\partial f_i}{\partial q_j} = \mathbb{J}_{ij}^{\mathrm{inertia}} + \mathbb{J}_{ij}^{\mathrm{elastic}} + \mathbb{J}_{ij}^{\mathrm{external}},$$

where

$$\mathbb{J}_{ij}^{\mathrm{inertia}} = \frac{m_i}{\Delta t^2}\delta_{ij},$$

$$\mathbb{J}_{ij}^{\mathrm{elastic}} = \frac{\partial^2 E_b}{\partial q_i \partial q_j} + \frac{\partial^2 E_s}{\partial q_i \partial q_j},$$

$$\mathbb{J}_{ij}^{\mathrm{external}} = -\frac{\partial F_{i,\mathrm{external}}}{\partial q_j}.$$

Programming Implementation

$t = 0$:
Compute natural discrete curvature κ_k^0 and undeformed length of each edge dl

Move from one time step to the next one:
$[\mathbf{q}(t_{k+1}), \dot{\mathbf{q}}(t_{k+1})] = \text{SimulationLoop}(\ [\mathbf{q}(t_k), \dot{\mathbf{q}}(t_k)]\ )$

Compute $\mathbf{f} = \mathtt{f}(\mathbf{q}(t_{k+1}))$
Compute $\mathbb{J} = \mathtt{J}(\mathbf{q}(t_{k+1}))$

$\mathbf{f}_{\text{free}} = \mathbf{f}(\text{free index})$
$\mathbb{J}_{\text{free}} = \mathbb{J}(\text{free index}, \text{free index})$

$\Delta\mathbf{q}_{\text{free}} = \mathbb{J}_{\text{free}} \backslash \mathbf{f}_{\text{free}}$

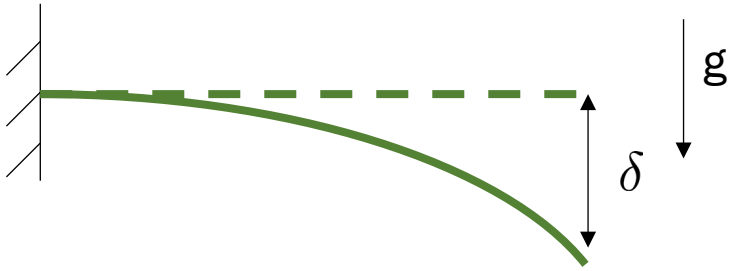
New estimate $\mathbf{q}_{\text{free}} = \Delta\mathbf{q}_{\text{free}} + \mathbf{q}_{\text{free}}$

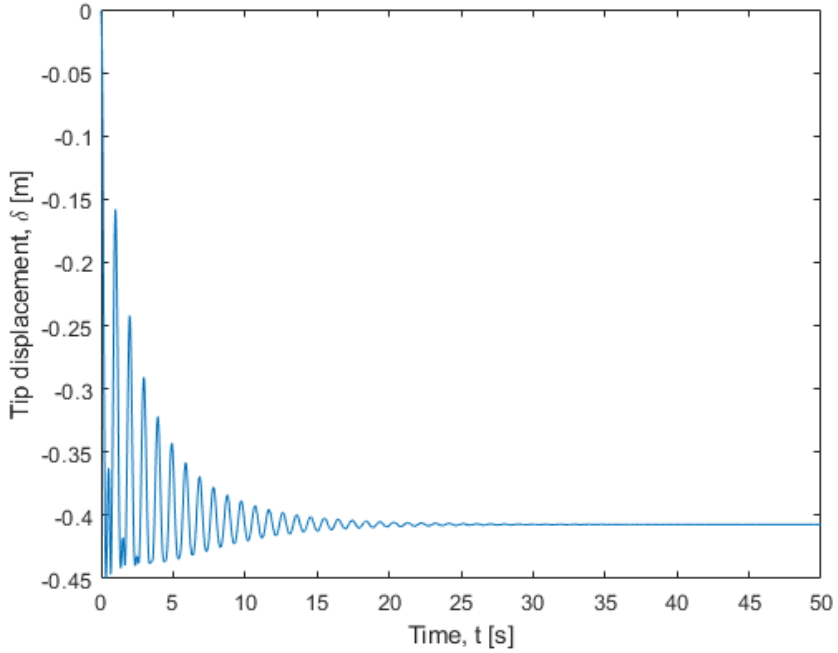
New estimate $\mathbf{q}(\text{free index}) = \mathbf{q}_{\text{free}}$

New estimate $\mathbf{q}(\text{fixed index}) = \mathbf{q}_{\text{fixed}}$

Dynamic Simulation vs Static Simulation

- Conventional Euler-Bernoulli beam theory has no time dependence
- Discrete simulations presented here include *dynamics*
  - Example: clamped beam under self-weight

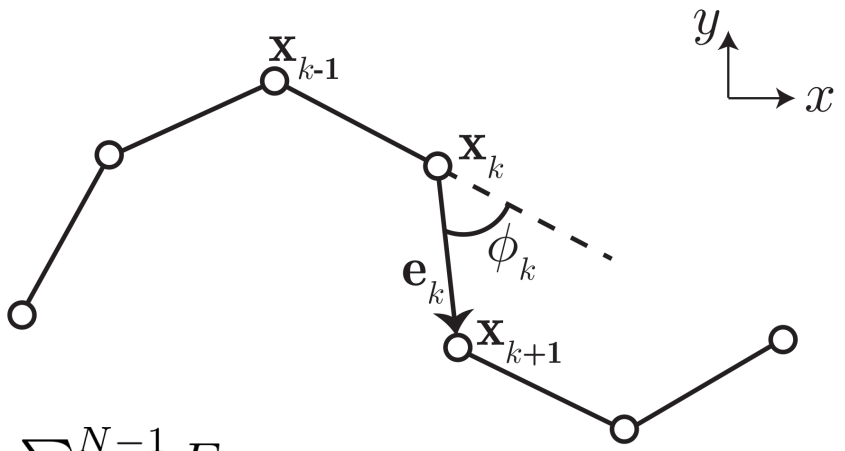




Caution: Implicit Euler artificially dissipates energy. Careful about physical accuracy.
Reference: "Newmark-Beta Method in Discrete Elastic Rods Algorithm to Avoid Energy Dissipation", W Huang, MK Jawed, Journal of Applied Mechanics 86 (8)

Discrete Elastic Bending Energy

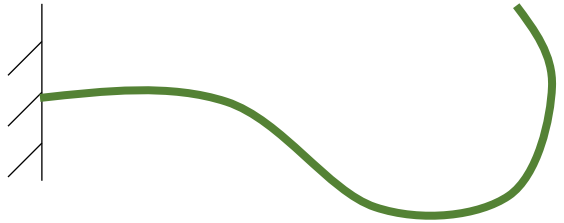
$$E_b = \sum_{k=2}^{N-1} E_{b,k}$$
$$\text{where } E_{b,k} = \frac{1}{2}\frac{EI}{dl}\left(\kappa_k\right)^2$$



What is the beam is naturally curved?

$$E_b = \sum_{k=2}^{N-1} E_{b,k}$$
$$\text{where } E_{b,k} = \frac{1}{2}\frac{EI}{dl}\left(\kappa_k - \kappa_k^0\right)^2$$

vs.



Straight beam

Naturally curved beam

Module 17

Networks of Beams

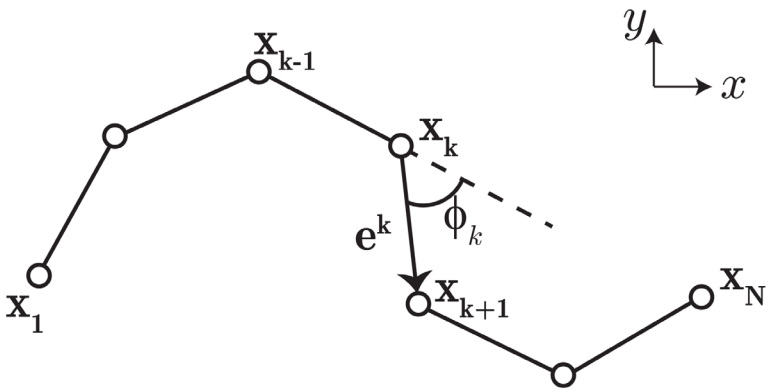
- Simulation of Networks of Beams
- Application to Soft Robotics

Goal

- Discrete Beam simulation for complex structures
- Application to soft robotics

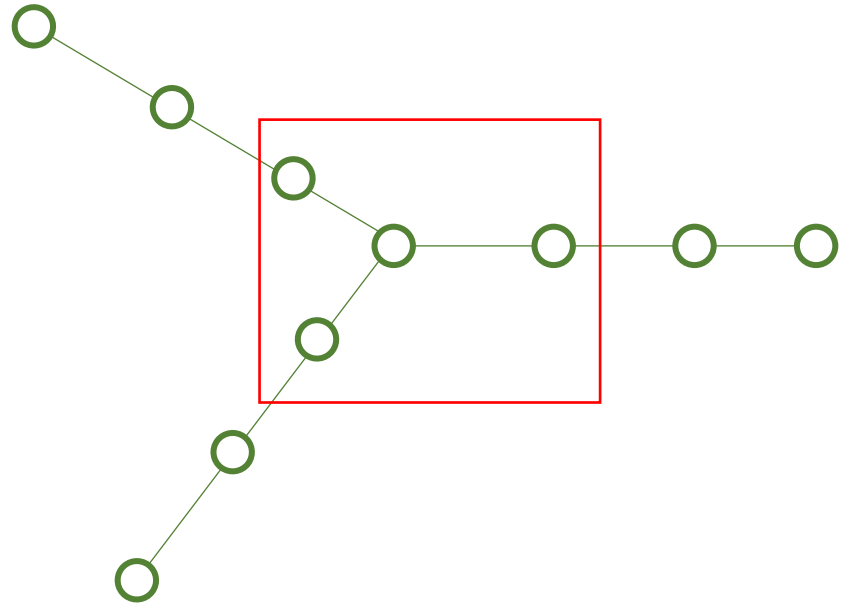
Sea-star Inspired Soft Robot

Beam vs Soft Robot (Sea-Star)



Elastic energy

- Bending energy between three consecutive nodes
- Stretching energy between two consecutive nodes



Soft robot

- “Joint” needs special treatment
- Actuation has to be incorporated

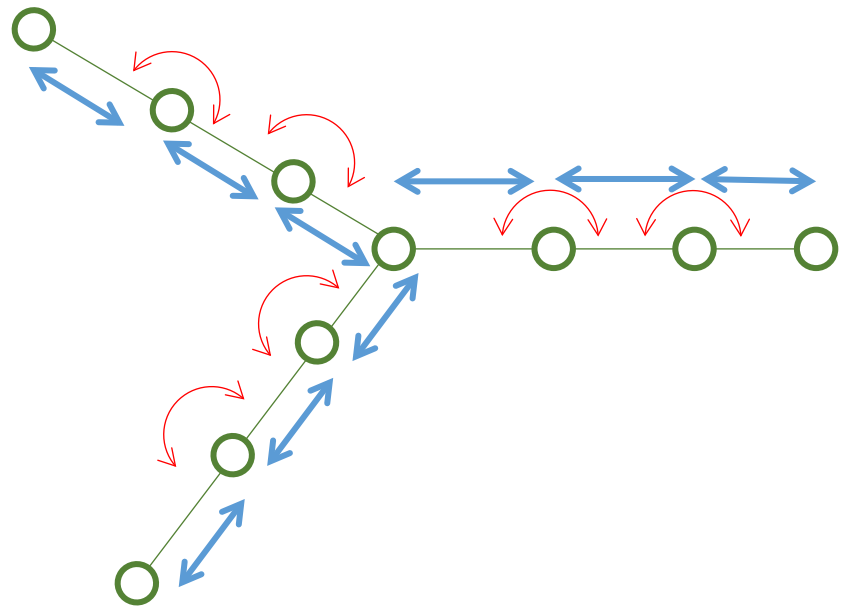
Energies of a Soft Robot (Sea-Star)

$2N$ equations of motion for N nodes:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{F}_{\text{external}} = \mathbf{0}$$

Bending energies

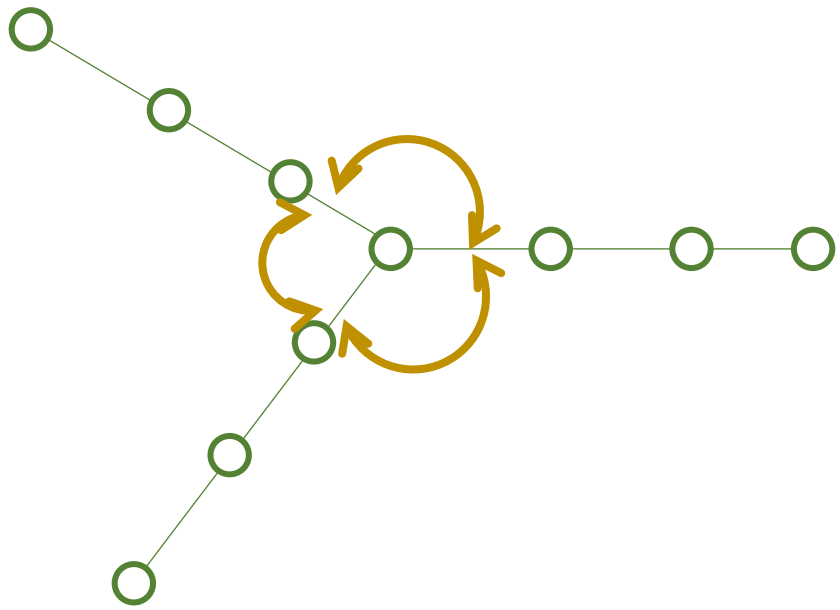
Stretching energies



Energies of a Soft Robot (Sea-Star)

$2N$ equations of motion for N nodes:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{F}_{\text{external}} = \mathbf{0}$$



Joint bending energy
- One node has three discrete bending energies

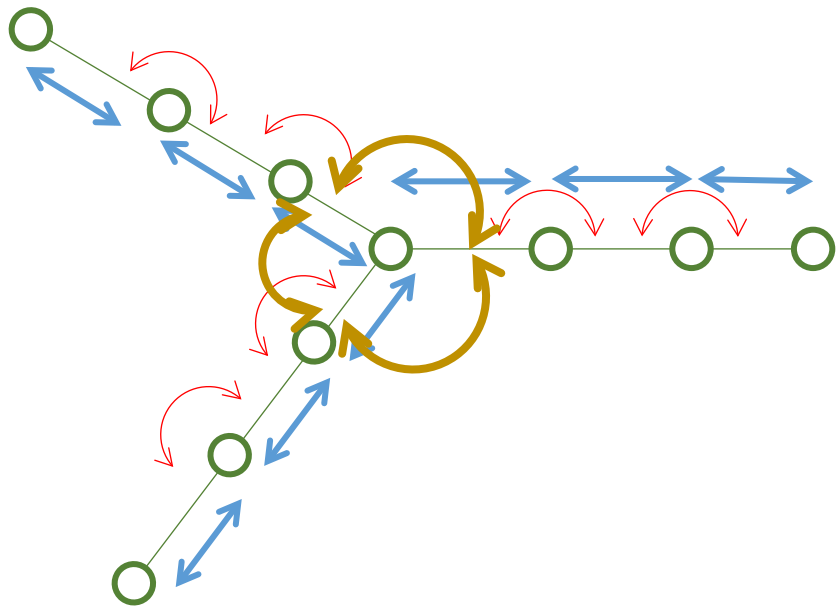
Energies of a Soft Robot (Sea-Star)

$2N$ equations of motion for N nodes:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{F}_{\text{external}} = \mathbf{0}$$

Bending energies

Stretching energies



Summary:
Elastic energy = bending energy + stretching energy, with special treatment for the joints.

Joint bending energy
- One node has three discrete bending energies

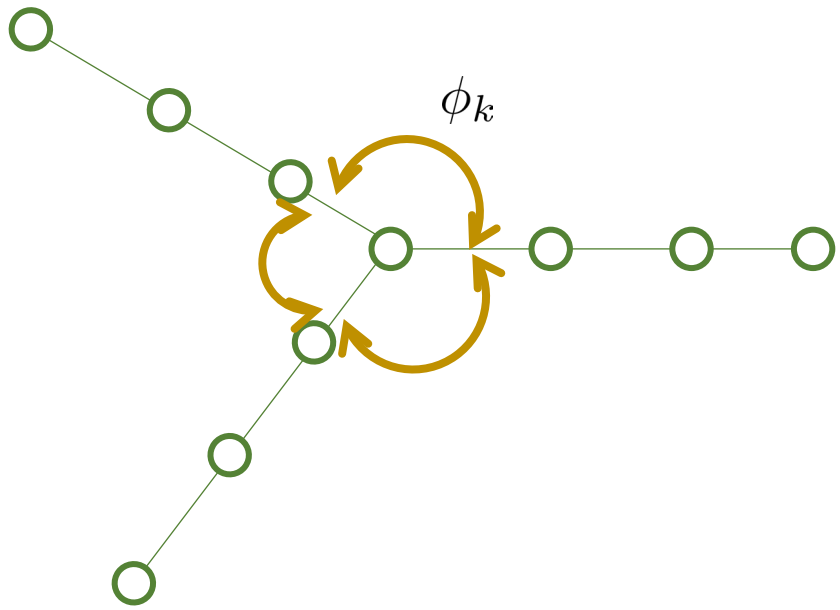
Actuation: vary natural curvature with time

Recall bending energy formulation:

$$\text{Total bending energy, } E_b = \sum E_{b,k}$$

$$E_{b,k} = \frac{1}{2}\frac{EI}{dl}\left(\kappa_k - \kappa_k^0\right)^2$$

$$\text{Discrete curvature, } \kappa_k = 2\tan(\phi_k/2)$$



Natural curvature varies with time for the bending energies associated with the head:

$$E_{b,k} = \frac{1}{2}\frac{EI}{dl}\left(\kappa_k - \kappa_k^0(t)\right)^2$$

External Forces

$2N$ equations of motion for N nodes:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \frac{\partial E^{\text{elastic}}}{\partial \mathbf{q}} - \mathbf{F}_{\text{external}} = \mathbf{0}$$

- External forces, e.g. hydrodynamic forces, can be incorporated using the $\boldsymbol{F_{external}}$ term in the equations of motion
  - A simple force model at low Reynolds number:

$$\mathbf{F}_{\text{external}} = \begin{bmatrix} \cdots \\ \cdots \\ -\mu\, dl \left(\frac{q_i(t_{k+1}) - q_i(t_k)}{\Delta t} \right) \\ \cdots \\ \cdots \end{bmatrix},$$

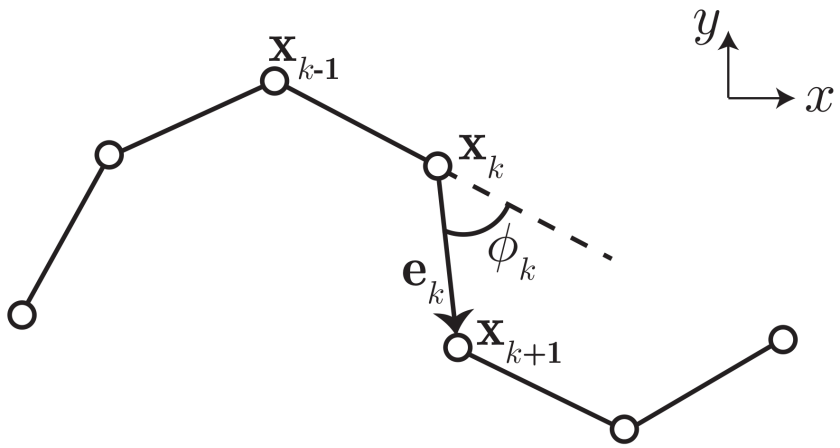
where μ is viscosity (Pa-s) and dl is the length of an edge

Recap: Beam to Soft Robots

1. Special treatment for the joints
2. Actuation
3. External forces

Discrete Elastic Bending Energy

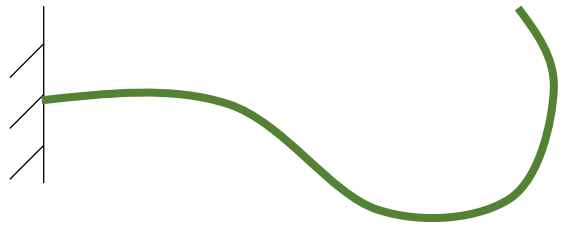
$$E_b = \sum_{k=2}^{N-2} E_{b,k}$$
$$\text{where } E_{b,k} = \frac{1}{2}\frac{EI}{dl}\left(\kappa_k - \kappa_k^0\right)^2$$



Curvature computed in *undeformed* (i.e. *stress-free*) configuration

In most examples, undeformed configuration is the same as initial configuration (i.e. shape of the beam at $t = 0$)

However, *undeformed* and *initial* are not necessarily the same



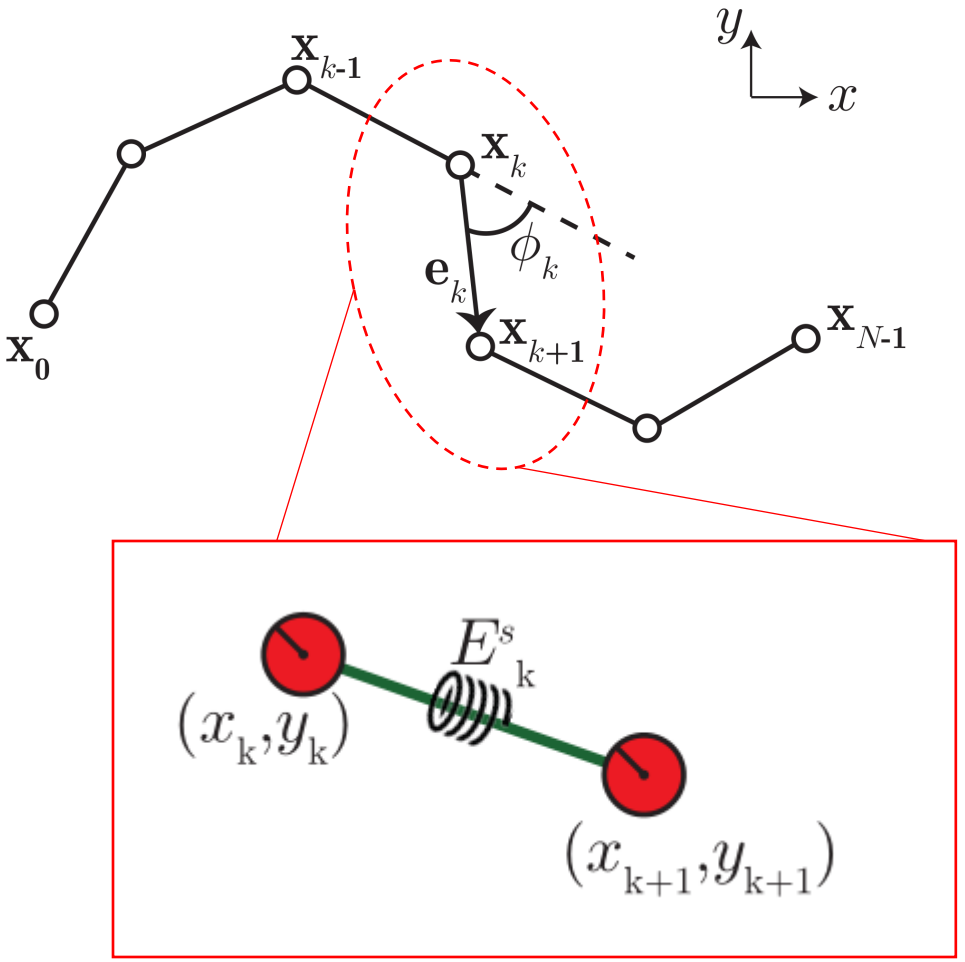
Naturally curved beam

Discrete Elastic Stretching Energy

Total elastic stretching energy: $E_s = \sum_{k=1}^{N-1} E_s^k$

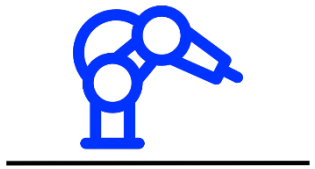
Stretching energy per edge: $E_s^k = \frac{1}{2} EA\varepsilon^2 dl$

Axial stretch: $\varepsilon = 1 - \frac{\sqrt{(x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2}}{dl}$,
where dl is the *undeformed* length of the edge



From a Beam to a Rod

- Rod
  - 3D configuration (Beam: 2D configuration)
  - Bending energy
  - Stretching energy
  - Twisting energy (Beam does not twist)



Module 18

Twist in a Discrete Rod

- Continuous Definition of Twist
- Discrete Definition of Twist
- Parallel Transport

Goal

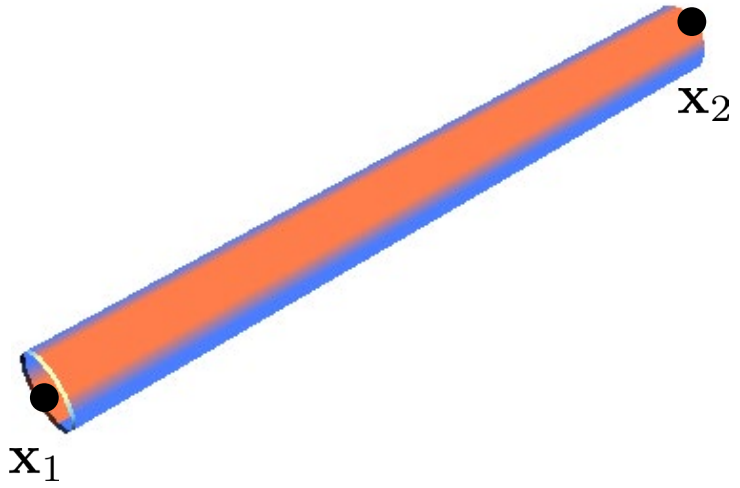
- Kinematics of a rod and the need for a material frame
- Twist of a rod
- Discrete representation of twist
- Orthonormal Adapted Frame
- Parallel Transport

Resources

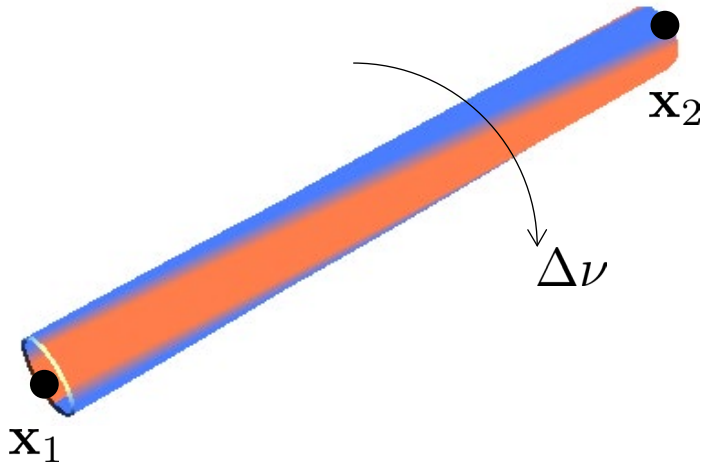
- Read *“Chapter 6: Discrete Twist”*
- Review undergraduate-level mechanics
  - *Chapter 6: Torsion of Prismatic Bars* of *Advanced Mechanics of Materials and Applied Elasticity* by A. C. Ugural and Saul Fenster

Nodes and its coordinates are not enough!

Undeformed rod

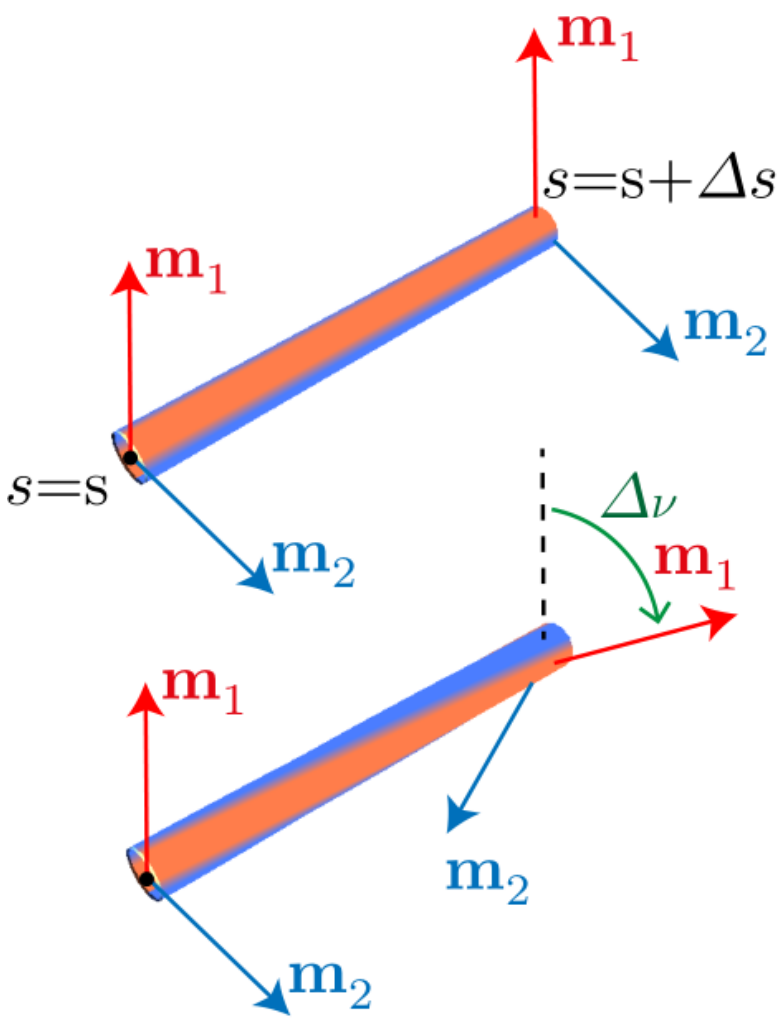


Deformed rod



If only nodes on the centerline are used to describe the kinematics of a rod, The energy will remain the same between the undeformed and deformed configurations.

Twist: Continuous Case



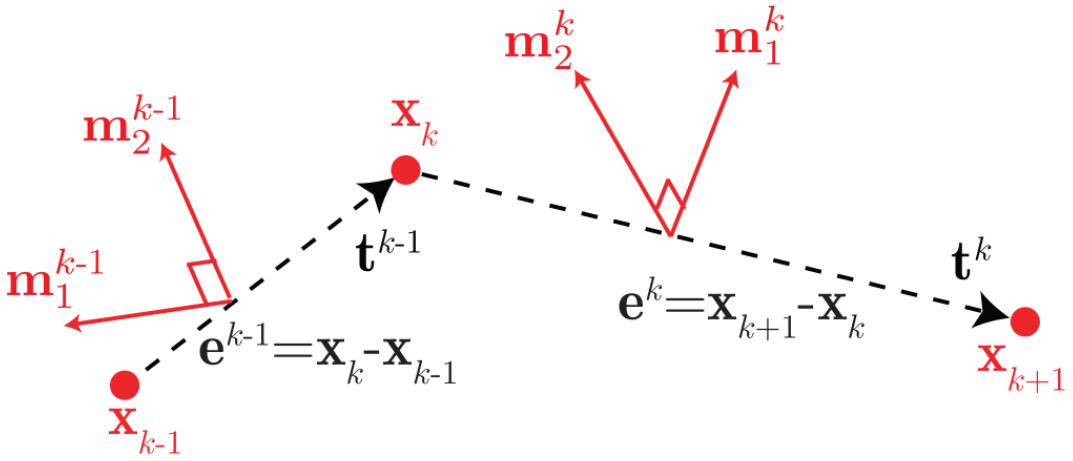
Continuous twist [unit: 1/meter]:

$$\tau(s) = \lim_{\Delta s \to 0} \frac{\Delta \nu}{\Delta s},$$

Twist: Discrete Case

- We will need three preliminary concepts before explaining the twist in a discrete rod
  1. Orthonormal adapted frame
  2. Signed angle
  3. Parallel transport

1. Orthonormal adapted frame



N nodes long the centerline, $\mathbf{x}_k$ $(1 \leq k \leq N)$

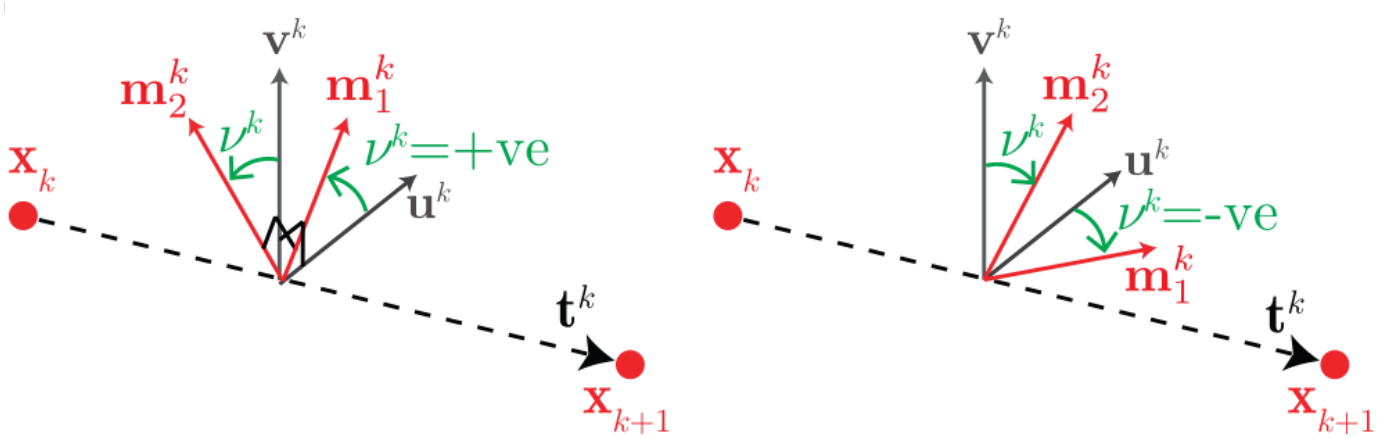
$N-1$ edges long the centerline, $\mathbf{e}^k = \mathbf{x}_{k+1} - \mathbf{x}_k$

Tangent on the k-th edge is $\mathbf{t}^k = \frac{\mathbf{e}^k}{\|\mathbf{e}^k\|}$

Adapted frame is any frame $\left(\mathbf{m}_1^k, \mathbf{m}_2^k, \mathbf{t}^k\right)$ that shares the tangent as the third director

Orthonormal means the three directors $\left(\mathbf{m}_1^k, \mathbf{m}_2^k, \mathbf{t}^k\right)$ are unit vectors and perpendicular to one another

2. Signed angle



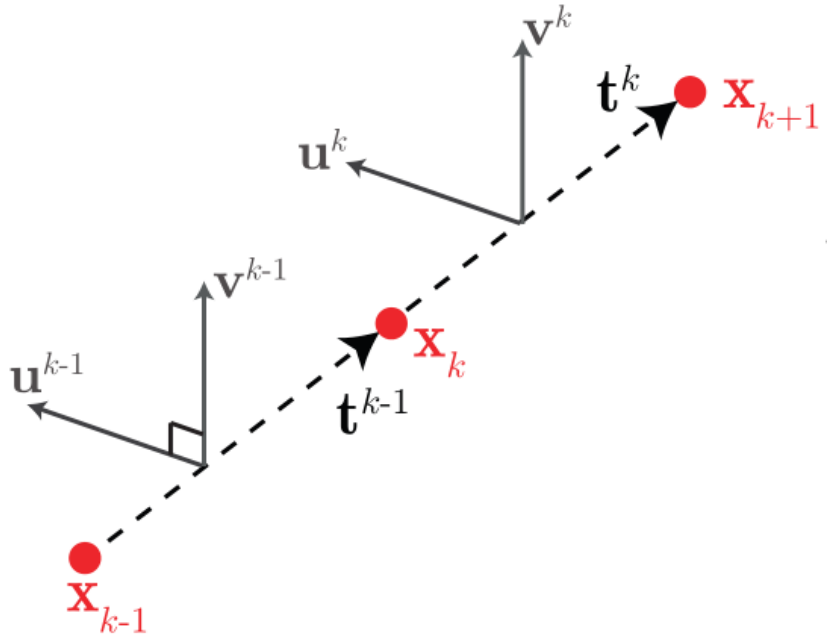
Consider two frames:

1. Material frame $\left(\mathbf{m}_1^k, \mathbf{m}_2^k, \mathbf{t}^k\right)$
2. Reference frame $\left(\mathbf{u}^k, \mathbf{v}^k, \mathbf{t}^k\right)$ (this frame is also orthonormal adapted)

The angle from $\mathbf{u}^k$ to $\mathbf{m}_1^k$ is ν^k.
This angle should have a "direction" $\mathbf{t}^k$ associated with it.
The sign of the angle, ν^k, can be determined using right hand rule.

3. Parallel transport

- What is the *most natural* or *twist free* way of moving an adapted frame one edge (e^{k-1}) to the next (e^{k})?
  - Easy if the two edges are aligned
  - (u^{k-1}, v^{k-1}, t^{k-1}) and (u^{k}, v^{k}, t^{k}) are the same

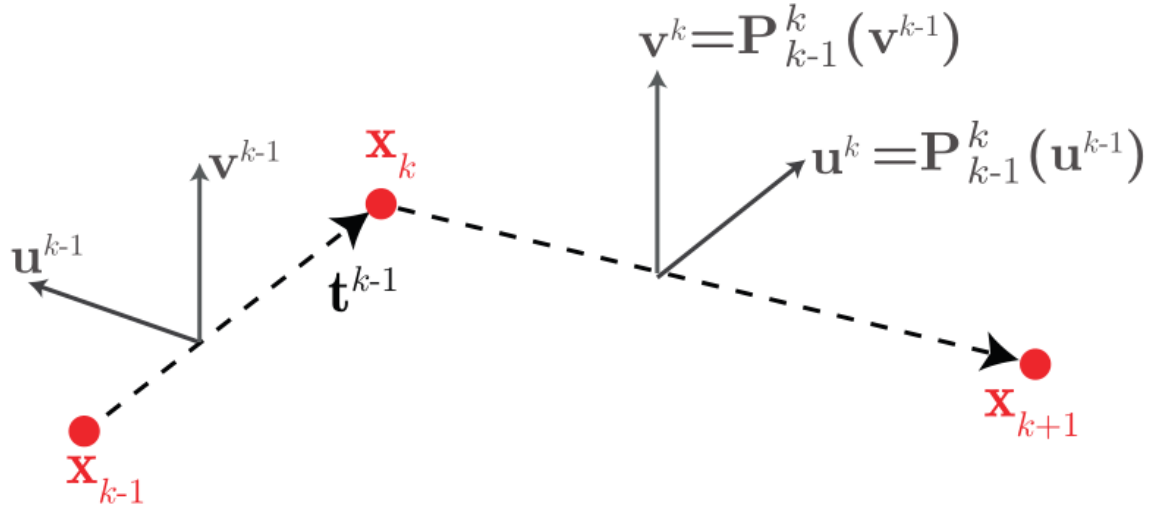


3. Parallel transport

- What is the *most natural* or *twist free* way of moving an adapted frame one edge (e^{k-1}) to the next (e^{k})?
  - If the two edges are not aligned, we need *parallel transport.*

$$P_{k-1}^{k}(\mathbf{u}) \equiv \texttt{parallel\_transport}(\mathbf{u}, \mathbf{t}^{k-1}, \mathbf{t}^{k})$$

$P_{k-1}^{k}(\mathbf{u})$ is the vector obtained after moving $\mathbf{u}$ from $\mathbf{t}^{k-1}$ to $\mathbf{t}_k$ without any twist (tangential angular velocity). $P_{k-1}^{k}(\mathbf{u})$ is perpendicular to $\mathbf{t}^{k}$ and $\mathbf{u}$ is perpendicular to $\mathbf{t}^{k-1}$.



3. Parallel transport

- What is the *most natural* or *twist free* way of moving an adapted frame one edge (e^{k-1}) to the next (e^{k})?
  - If the two edges are not aligned, we need *parallel transport.*

$$P_{k-1}^{k}(\mathbf{u}) \equiv \texttt{parallel\_transport}(\mathbf{u}, \mathbf{t}^{k-1}, \mathbf{t}^{k})$$

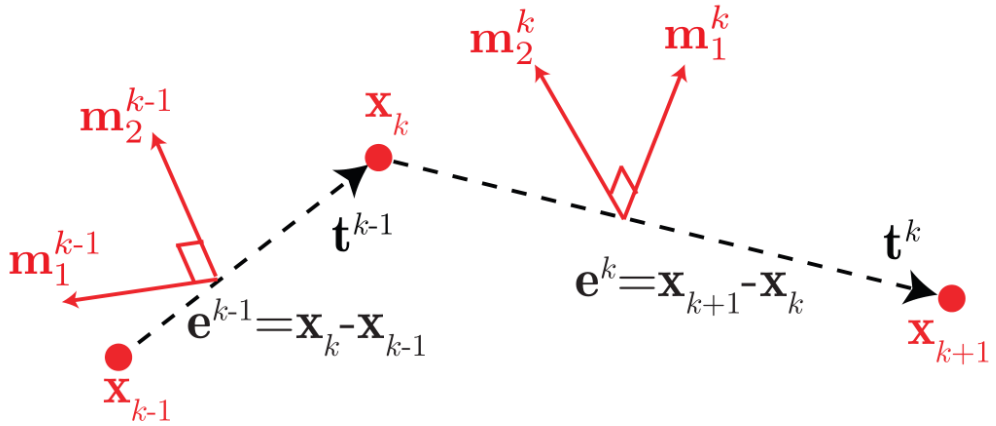
$$\begin{aligned}
\mathbf{b} &= \mathbf{t}^{k-1} \times \mathbf{t}^{k}, \\
\hat{\mathbf{b}} &= \frac{\mathbf{b}}{|\mathbf{b}|}, \\
\mathbf{n}_1 &= \mathbf{t}^{k-1} \times \hat{\mathbf{b}}, \\
\mathbf{n}_2 &= \mathbf{t}^{k} \times \hat{\mathbf{b}}, \\
P_{k-1}^{k}(\mathbf{u}) &= (\mathbf{u} \cdot \mathbf{t}^{k-1})\mathbf{t}^{k} + (\mathbf{u} \cdot \mathbf{n}_1)\mathbf{n}_2 + (\mathbf{u} \cdot \hat{\mathbf{b}})\hat{\mathbf{b}}.
\end{aligned}$$

Twist: Discrete Case

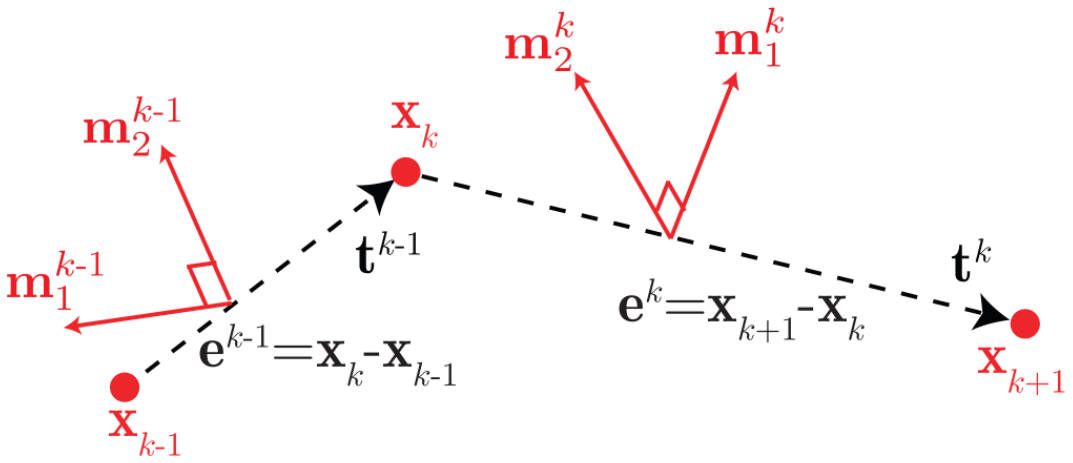
The integrated discrete twist (or, more simply, *twist*) of an adapted frame [e.g. $(\mathbf{m}_1^k, \mathbf{m}_2^k, \mathbf{t}^k)$ with $k = 1, \ldots, N-1$] at node $\mathbf{x}_k$ is the signed angle from $P_{k-1}^k(\mathbf{m}_1^{k-1})$ to $\mathbf{m}_1^k$ about the axis $\mathbf{t}^k$.

The word *integrated*, which may often be omitted, is used to distinguish this dimensionless measure of twist from the conventional definition of twist with unit of 1/length.

The integrated twist can be divided by the *Voronoi length* of the node to obtain the conventional twist, where the Voronoi length $\bar{l}_k = \left(|\mathbf{e}^{k-1}| + |\mathbf{e}^k|\right)/2$ and $\bar{(\,)}$ represents evaluation in undeformed state.



Discrete Elastic Twisting Energy



$$E_t = \sum_{k=2}^{N-1} E_{t,k}$$

where $E_{t,k} = \frac{1}{2}\frac{GJ}{\bar{l}_k}(\tau_k)^2$

- G = shear modulus = $\frac{\text{Young's modulus}}{2(1+\text{Poisson ratio})}$
- J = polar moment of area ($= \frac{\pi r_0^4}{2}$ if circular)
- τ_k = signed angle from $P_{k-1}^k(\mathbf{m}_1^{k-1})$ to $\mathbf{m}_1^k$ about the axis $\mathbf{t}^k$

Recall: From a Beam to a Rod

- Rod
  - 3D configuration (Beam: 2D configuration)
  - Bending energy
  - Stretching energy
  - Twisting energy (Beam does not twist)

Module 19

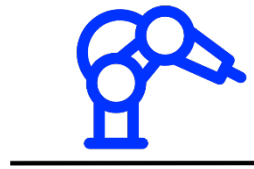
Space Parallel Reference Frame

- Definition of twist using space parallel reference frame

Goal

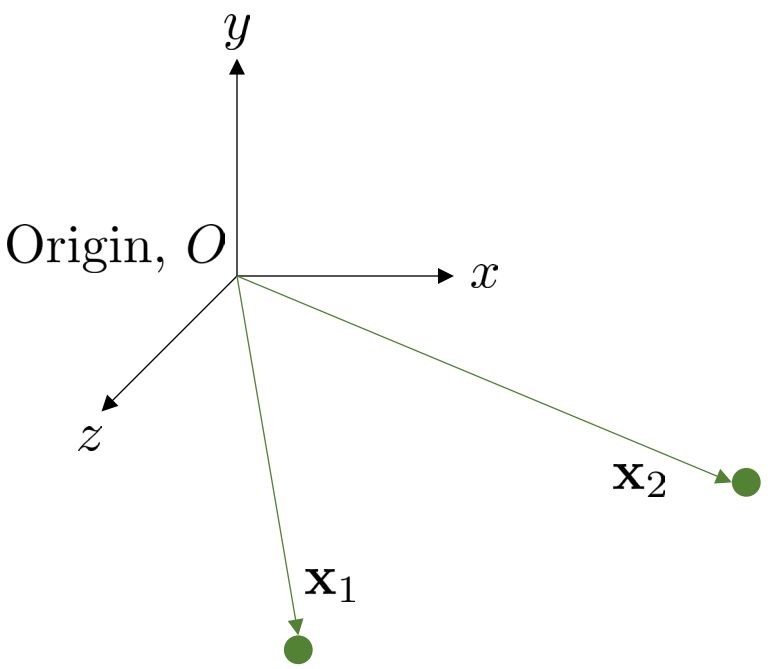
- Formulate a reference frame on the rod centerline using parallel transport along the arc-length (space parallel transport)
- Compute the twist along the rod using space parallel reference frame
- At the end of this module, you should be able to compute the twisting energy of a rod given its centerline and material frame

Resources



- Read *“Chapter 6: Discrete Twist”*
- Review Bergou, Miklós, et al. “Discrete elastic rods” *ACM SIGGRAPH (2008)* 1-12.

Relative Position

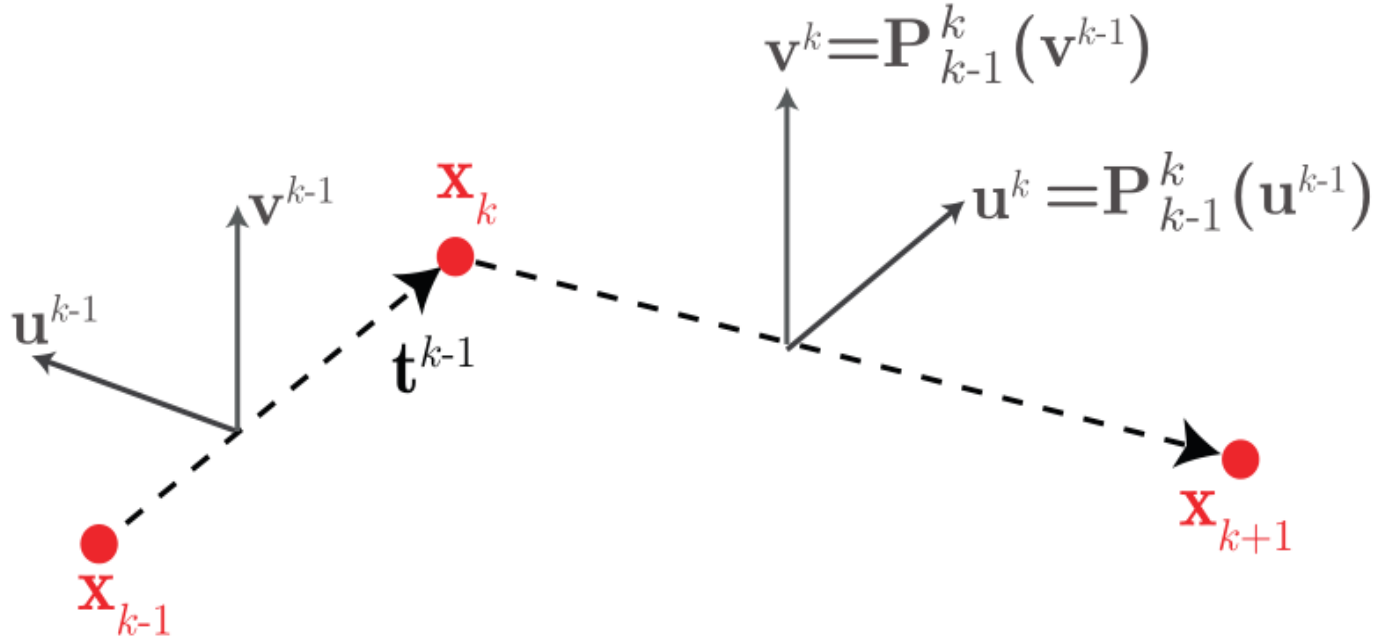


When describing the nodal coordinates, e.g. $\mathbf{x_1}$, we implicitly assume a reference frame with origin O.

Relative Rotation - Twist

When describing the rotation of the material frame, we need a reference frame

- How to compute the reference frame, $(\mathbf{u}^k, \mathbf{v}^k, \mathbf{t}^k)$?
- Known quantities: the nodal coordinates, $\mathbf{x}_k$



Reference Frame

- Since the positions of the nodes are known, the tangent on each edge is known:

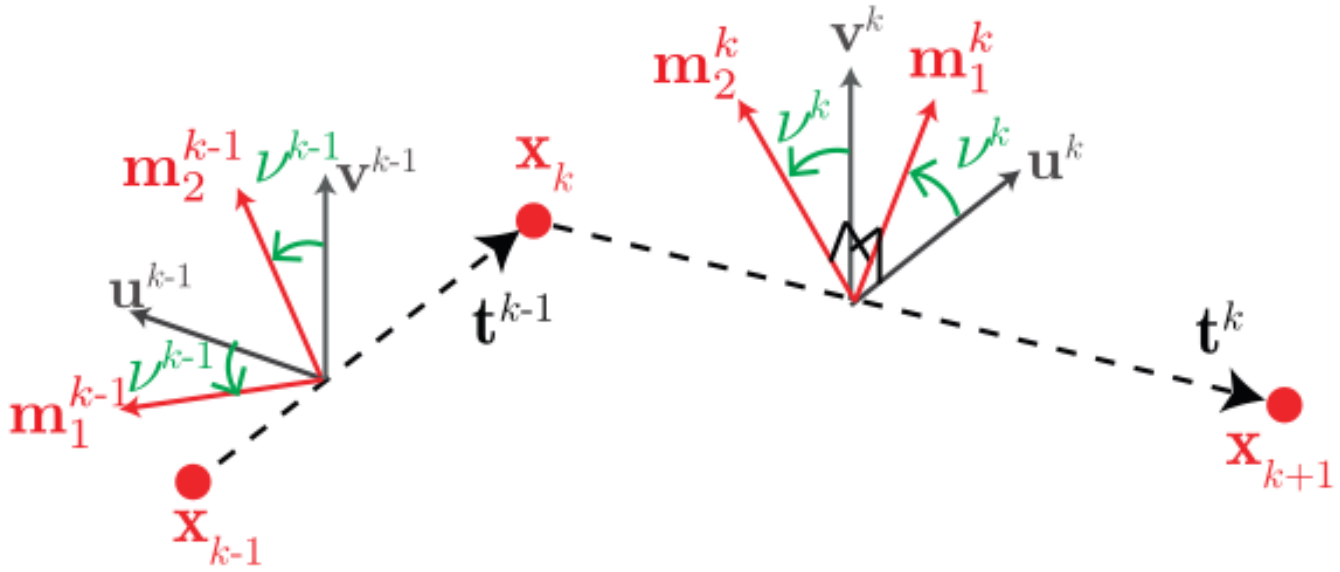
$$\mathbf{t}^k = \frac{\mathbf{e}^k}{\|\bar{\mathbf{e}^k}\|} = \frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{\|\bar{\mathbf{e}^k}\|}$$

- Start with an arbitrary (but adapted) orthonormal frame at the first edge, $\mathbf{e}^1$. For that purpose, choose any unit vector $\mathbf{u}^1$ that is orthogonal to $\mathbf{t}^1$. The second director is $\mathbf{v}^1 = \mathbf{t}^1 \times \mathbf{u}^1$ and the frame is $(\mathbf{u}^1, \mathbf{v}^1, \mathbf{t}^1)$.

- Sequentially compute the reference frames for the subsequent edges using $\mathbf{u}^k = P_{k-1}^k(\mathbf{u}^{k-1}) \equiv \texttt{parallel\_transport}(\mathbf{u}^{k-1}, \mathbf{t}^{k-1}, \mathbf{t}^k)$ and $\mathbf{v}^k = \mathbf{t}^k \times \mathbf{u}^k$.

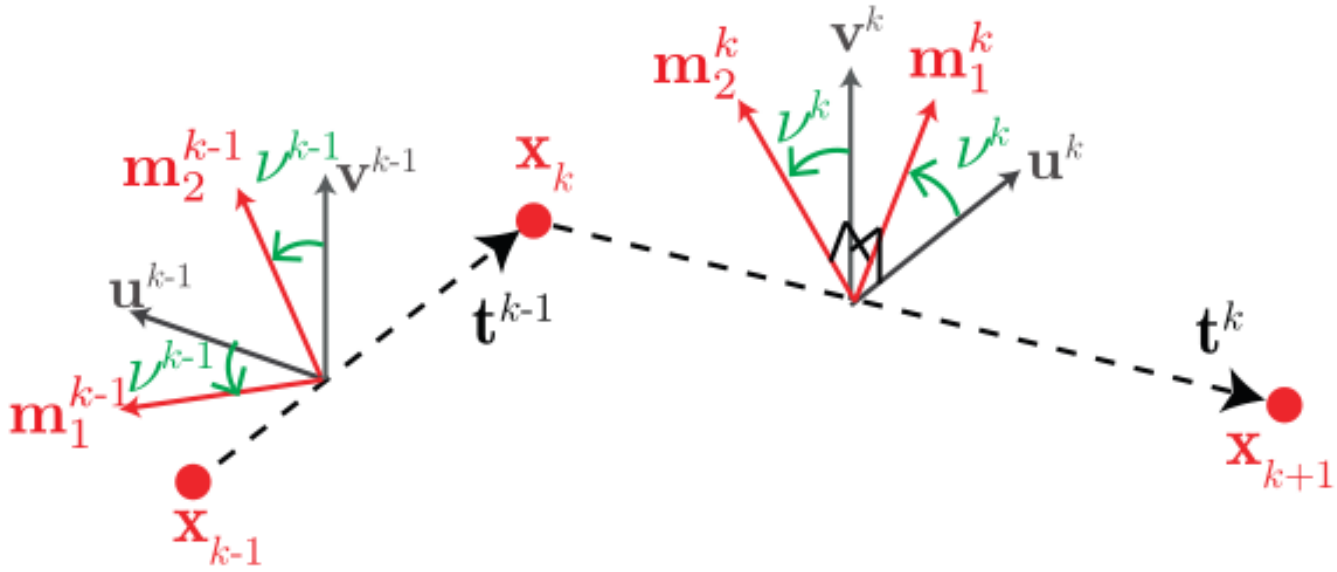
How to compute the discrete integrated twist?

- Known quantities: configuration of the rod
  - Nodal coordinates, x_k
  - Material frame on each edge, (m_1^k, m_2^k, t^k)
  - We learned how to compute the reference frame, (u^k, v^k, t^k), using space parallel transport



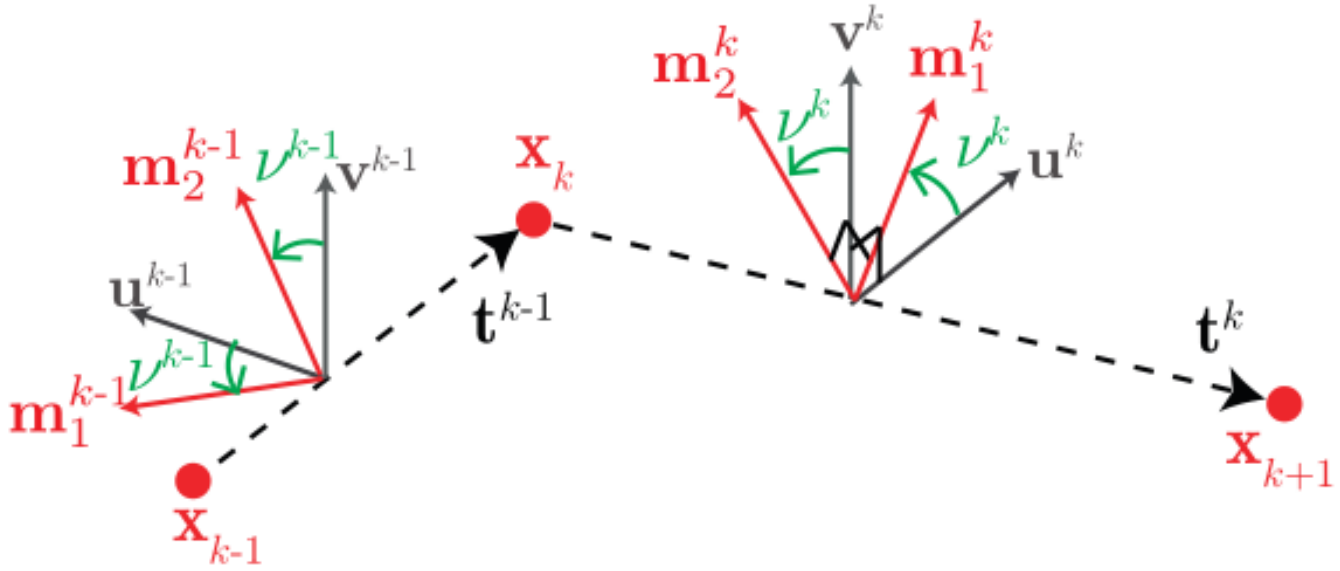
How to compute the discrete integrated twist?

- Steps to compute the twist at the node, x_k
  1. Compute the signed angles from the reference frame to the material frame, ν^k
  2. Discrete integrated twist is the difference between the signed rotation angles of two consecutive edges: $\tau_k = \nu^k - \nu^{k-1}$



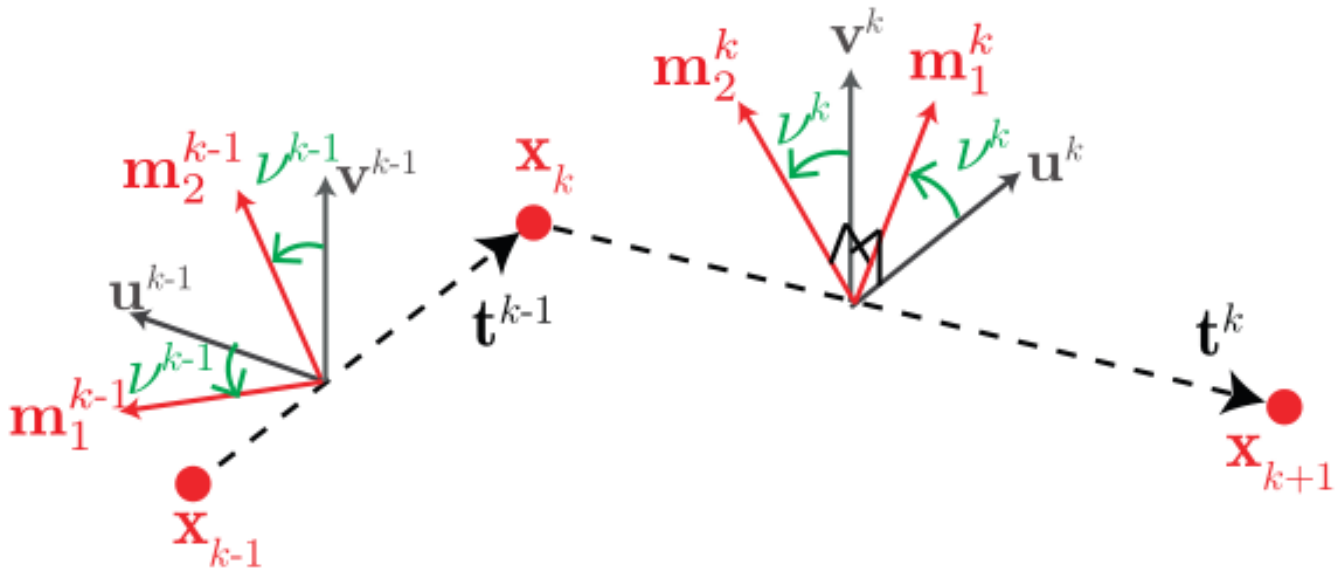
Degrees of Freedom of a Rod

- If only the nodal coordinates, x_k, and twist angles, ν_k, are known, the complete configuration of the rod can be determined
  - First, determine the space parallel reference frame
  - Rotate the reference frame director, $\boldsymbol{u^k}$, by an angle ν_k to get the material frame director, $\boldsymbol{m_1^k}$. Recall that $\boldsymbol{m_2^k = t^k \times m_1^k}$.

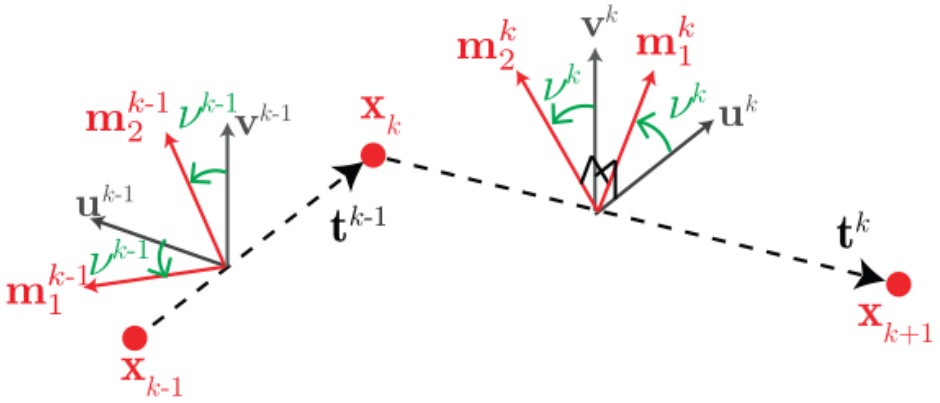


Degrees of Freedom (DOF) of a Rod

- A rod with N nodes has $(4N\text{-}1)$ DOFs
  1. $3N$ DOFs associated with the positions of N nodes, $\boldsymbol{x_k}$ $(1 \leq k \leq N)$
  2. $(N\text{-}1)$ DOFs for the twist angles of the edges, $\boldsymbol{\nu^k}$ $(1 \leq k \leq N-1)$



Discrete Elastic Twisting Energy



$$E_t = \sum_{k=2}^{N-1} E_{t,k}$$

where $E_{t,k} = \frac{1}{2}\frac{GJ}{\bar{l}_k}(\tau_k)^2 = \frac{1}{2}\frac{GJ}{\bar{l}_k}\left(\nu^k - \nu^{k-1}\right)^2$

- G = shear modulus = $\frac{\text{Young's modulus}}{2(1+\text{Poisson ratio})}$
- J = polar moment of area ($= \frac{\pi r_0^4}{2}$ if circular)
- τ_k = signed angle from $P_{k-1}^k(\mathbf{m}_1^{k-1})$ to $\mathbf{m}_1^k$ about the axis $\mathbf{t}^k$

Future Plan

- Bergou et al. *SIGGRAPH* (2008) used space-parallel reference frame
- In 2010 (Bergou et al. *SIGGRAPH* (2010)), the authors formulated another formulation of reference frame: time-parallel reference frame
  - This leads to significant speed-up in computation
- In this class, we will follow the formulation in the 2010 version and discuss time-parallel reference frame in next module