CEE6501 — Lecture 7.2

Introduction to 2D Frame Analysis

Learning Objectives

By the end of this lecture, you will be able to:

-

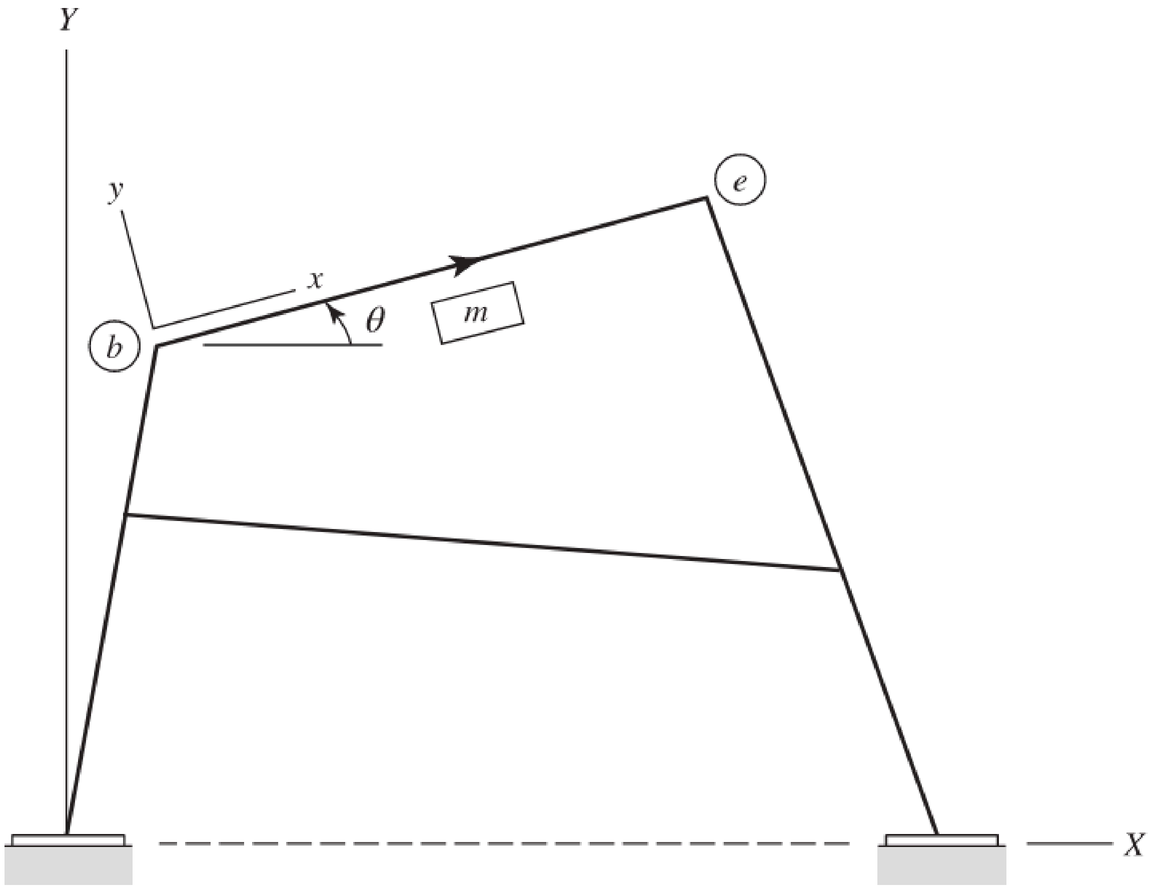
Agenda

Part 1 — Roadmap: truss → beam → 3D frame
Part 2 —

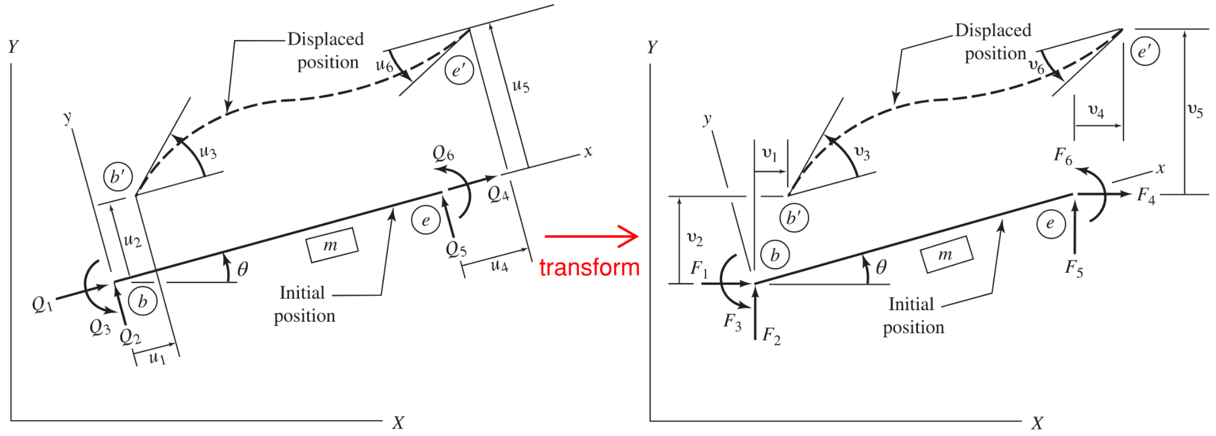
Part 1 - Local to Global Transformation

Note: See Lecture 3.2 slides (Trusses) for detaled information on this.

Generic Truss Element in a Structure



Local/Global Perspective



- **Local coordinate system (left):** forces $\boldsymbol{Q}$, displacements $\boldsymbol{u}$
- **Global coordinate system (right):** forces $\boldsymbol{F}$, displacements $\boldsymbol{v}$

Global ($\boldsymbol{F}$) $\longrightarrow$ Local ($\boldsymbol{Q}$) Force Transformation

Use trigonometric relationships to resolve global forces into the local coordinate system.

At node i:

$$\begin{aligned} Q_1 &= F_1 \cos\theta + F_2 \sin\theta \\ Q_2 &= -F_1 \sin\theta + F_2 \cos\theta \\ Q_3 &= F_3 \end{aligned}$$

The local and global z-axes are aligned (out of plane), therefore the bending moment is unchanged:

$$Q_3 = F_3$$

At node, j:

$$\begin{aligned} Q_4 &= F_4 \cos\theta + F_5 \sin\theta \\ Q_5 &= -F_4 \sin\theta + F_5 \cos\theta \\ Q_6 &= F_6 \end{aligned}$$

Global ($\boldsymbol{F}$) → Local ($\boldsymbol{Q}$) Force Transformation

- Local element forces $\boldsymbol{Q}$ are obtained by **rotating** global nodal forces $\boldsymbol{F}$ into the member's local coordinate system
- Each node contributes a 3×3 transformation block:
  - Translational DOFs → **rotated by** θ
  - Rotational DOF → **invariant** (local and global z-axes are aligned)

$$\boldsymbol{Q} = \boldsymbol{T}\,\boldsymbol{F}$$

$$\begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \end{Bmatrix} = \underbrace{\begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos\theta & \sin\theta & 0 \\ 0 & 0 & 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{T}} \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{Bmatrix}$$

Note: This is identical to the truss transformation for translational DOFs; the only addition is the rotational DOF (unchanged). See truss lecture for details on **direction cosines**.

Global ($\boldsymbol{v}$) → Local ($\boldsymbol{u}$) Displacements

- Nodal displacements are transformed using the **same rotation matrix** as forces
- Displacements and forces transform identically because they are defined along the **same directions**

$$\boldsymbol{u} = \boldsymbol{T}\boldsymbol{v}$$

Local ($\boldsymbol{Q}$) → Global ($\boldsymbol{F}$) Force

- This is the **reverse of the global → local process**
- Local member forces or displacements are **rotated back** into the global (X, Y) directions

$$\boldsymbol{F} = \boldsymbol{T}^{\top}\boldsymbol{Q}$$

Local ($\boldsymbol{u}$) → Global ($\boldsymbol{v}$) Displacements

- This is the **reverse of the global → local process**
- Local member forces or displacements are **rotated back** into the global (X, Y) directions

$$\boldsymbol{v} = \boldsymbol{T}^{\top}\boldsymbol{u}$$

Part 2 - Global Stiffness Relationship

We start from the known local relation:

$$\mathbf{Q} = \mathbf{k}\mathbf{u} + \mathbf{Q}^F$$

where:

- $\mathbf{Q}$ = local end force vector
- $\mathbf{k}$ = local stiffness matrix
- $\mathbf{u}$ = local displacement vector
- $\mathbf{Q}^F$ = local fixed-end force vector

Step 1 — Transform Forces to Global System

Forces transform as:

$$\mathbf{F} = \mathbf{T}^T\mathbf{Q}$$

Substitute the local stiffness relation:

$$\mathbf{F} = \mathbf{T}^T(\mathbf{k}\mathbf{u} + \mathbf{Q}^F)$$

Distribute:

$$\mathbf{F} = \mathbf{T}^T\mathbf{k}\mathbf{u} + \mathbf{T}^T\mathbf{Q}^F$$

Step 2 — Transform Displacements

Displacements transform as:

$$\mathbf{u} = \mathbf{T}\mathbf{v}$$

Substitute into previous equation:

$$\mathbf{F} = \mathbf{T}^T\mathbf{k}\mathbf{T}\mathbf{v} + \mathbf{T}^T\mathbf{Q}^F$$

Step 3 — Define Global Quantities

Define:

Global Member Stiffness Matrix

$$\boxed{\mathbf{K} = \mathbf{T}^T \mathbf{k} \mathbf{T}}$$

Global Fixed-End Force Vector

$$\boxed{\mathbf{F}^F = \mathbf{T}^T \mathbf{Q}^F}$$

Final Global Member Relation

Substitute definitions:

$$\boxed{\mathbf{F} = \mathbf{K}\mathbf{v} + \mathbf{F}^F}$$

Consistent with our notation, where we use u for global

$$\boxed{\mathbf{F} = \mathbf{K}\mathbf{u} + \mathbf{F}^F}$$

Part 3 - 2D Frame Element Stiffness Matrix

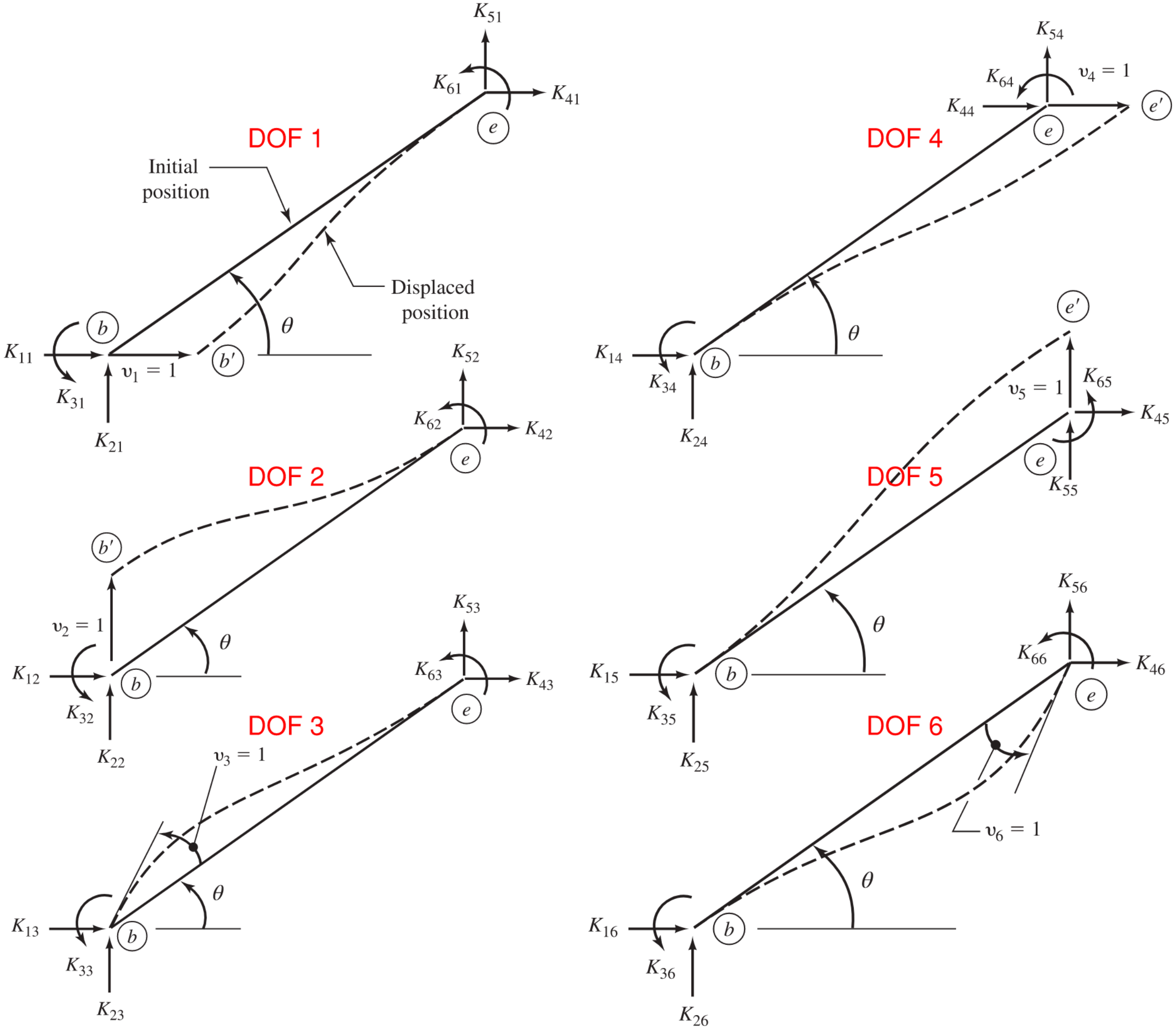
> *Global Coordinates*

Unit Displacement Method

You can derive 2D frame element from scratch using unit displacement method.

$$k_{ij} = \text{force at DOF } i \text{ due to a unit displacement at DOF } j, \\ \text{with all other DOFs fixed.}$$

LEFT END | RIGHT END



Closed Form Expression

A closed-form expression for the global member stiffness matrix **does exist** for frame elements (Section 6.4 in Kassimali)

However, unlike the truss case, the resulting expression is considerably more involved and not especially transparent.

$$
\mathbf{K} = \frac{EI}{L^3}\begin{bmatrix}
\frac{AL^2}{I}\cos^2\theta + 12\sin^2\theta & \left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & -6L\sin\theta & -\left(\frac{AL^2}{I}\cos^2\theta + 12\sin^2\theta\right) & -\left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & -6L\sin\theta \\
\left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & \frac{AL^2}{I}\sin^2\theta + 12\cos^2\theta & 6L\cos\theta & -\left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & -\left(\frac{AL^2}{I}\sin^2\theta + 12\cos^2\theta\right) & 6L\cos\theta \\
-6L\sin\theta & 6L\cos\theta & 4L^2 & 6L\sin\theta & -6L\cos\theta & 2L^2 \\
-\left(\frac{AL^2}{I}\cos^2\theta + 12\sin^2\theta\right) & -\left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & 6L\sin\theta & \frac{AL^2}{I}\cos^2\theta + 12\sin^2\theta & \left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & 6L\sin\theta \\
-\left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & -\left(\frac{AL^2}{I}\sin^2\theta + 12\cos^2\theta\right) & -6L\cos\theta & \left(\frac{AL^2}{I} - 12\right)\cos\theta\sin\theta & \frac{AL^2}{I}\sin^2\theta + 12\cos^2\theta & -6L\cos\theta \\
-6L\sin\theta & 6L\cos\theta & 2L^2 & 6L\sin\theta & -6L\cos\theta & 4L^2
\end{bmatrix}
$$

For this reason, it is generally cleaner and more systematic to just compute:

$$\mathbf{K} = \mathbf{T}^T \mathbf{k} \mathbf{T}$$

Part 4 - Member Forces

Global Coordinates

Why Need to Rotate to Global?

We need the fixed-end forces (FEFs) in **global coordinates** before we assemble the structure, because the global system solution is written and solved in the **global coordinate system**.

Mathematical Expression

The force transformation we use for fixed-end forces is:

$$\mathbf{F}^F = \mathbf{T}^T\mathbf{Q}^F$$

Using the same $\mathbf{T}$ as before:

$$\mathbf{T}^T = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

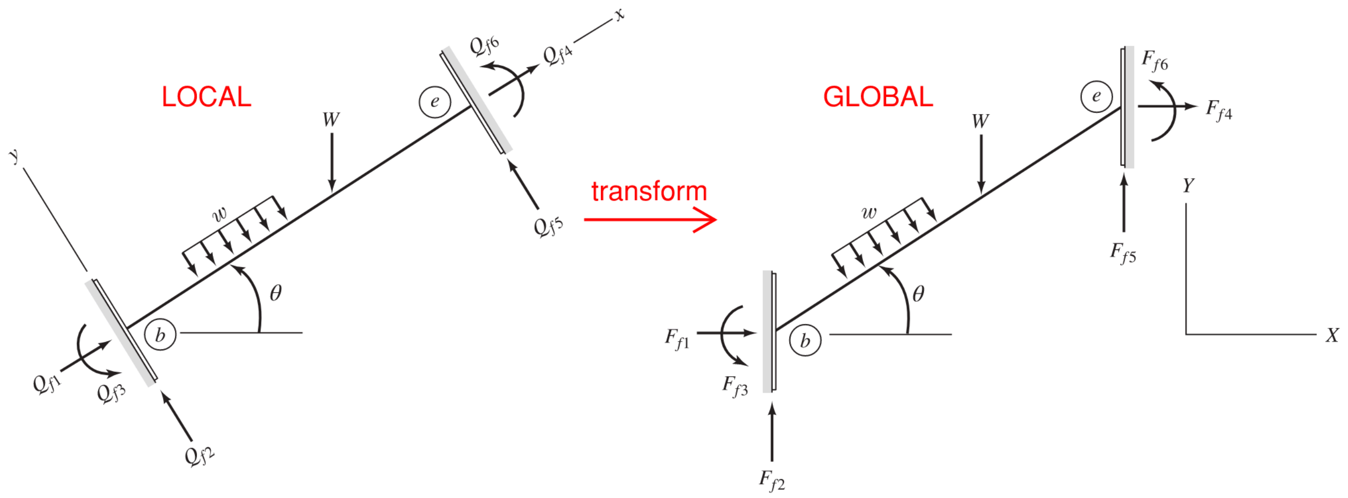
Expression for $\mathbf{F}^F$

Let $c = \cos\theta$ and $s = \sin\theta$. Then:

$$\begin{Bmatrix} F_1^F \\ F_2^F \\ F_3^F \\ F_4^F \\ F_5^F \\ F_6^F \end{Bmatrix} = \mathbf{T}^T \begin{Bmatrix} Q_1^F \\ Q_2^F \\ Q_3^F \\ Q_4^F \\ Q_5^F \\ Q_6^F \end{Bmatrix} = \begin{Bmatrix} c\,Q_1^F - s\,Q_2^F \\ s\,Q_1^F + c\,Q_2^F \\ Q_3^F \\ c\,Q_4^F - s\,Q_5^F \\ s\,Q_4^F + c\,Q_5^F \\ Q_6^F \end{Bmatrix}$$

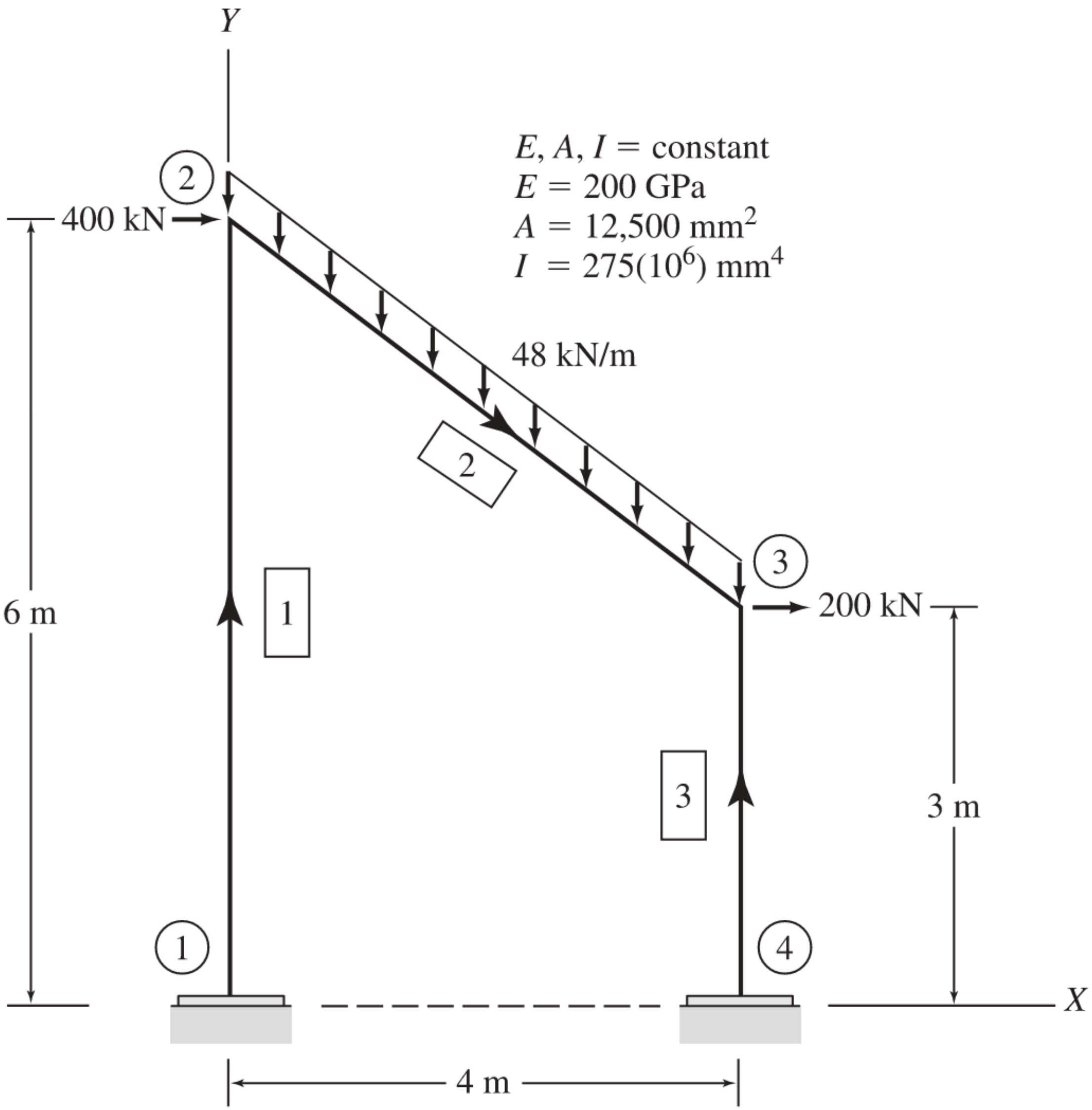
Transformation Process

1. Compute the member fixed-end force vector in **local** coordinates: $\mathbf{Q}^F$
2. Rotate it into **global** coordinates using: $\mathbf{F}^F = \mathbf{T}^T\mathbf{Q}^F$
3. Assemble $\mathbf{F}^F$ into the global force vector (same coordinate system as the global stiffness matrix)



Part 5 - Example (6.3 and 6.4 in Kassimali)

Continue with same structure.

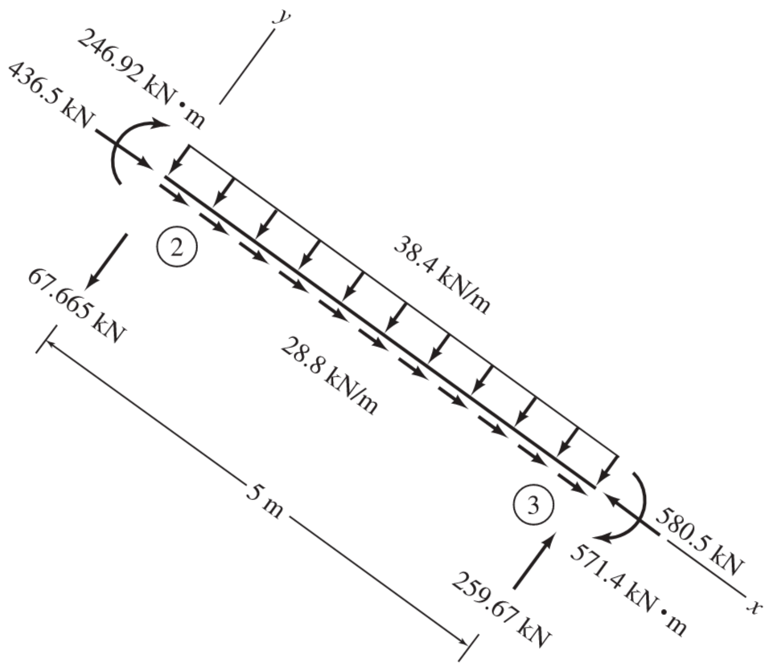


Local -> Global Element Forces

If given local element forces from earlier problem

$$\mathbf{F} = \mathbf{T}^T\mathbf{Q}$$

$$\mathbf{Q} = \begin{bmatrix} 436.5 & -67.669 & -246.929 & -580.5 & 259.669 & -571.418 \end{bmatrix}^T$$



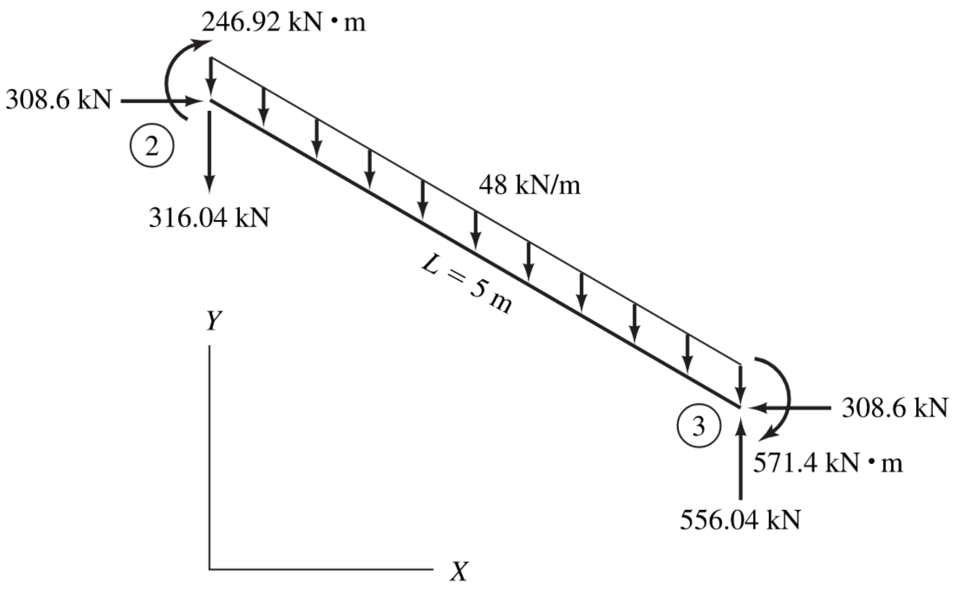
In [114]:
```
import numpy as np
np.set_printoptions(precision=3, suppress=True)

def transformation_matrix(theta):
    """
    Returns the 6x6 transformation matrix T for a 2D frame element.
    """
    theta = np.radians(theta)  # convert degrees → radians

    c = np.cos(theta)
    s = np.sin(theta)

    T = np.array([
        [ c, s,  0,  0,  0,  0],
        [ -s,  c,  0,  0,  0,  0],
        [ 0,  0,  1,  0,  0,  0],
        [ 0,  0,  0,  c, s,  0],
        [ 0,  0,  0,  -s,  c,  0],
        [ 0,  0,  0,  0,  0,  1]
    ])

    return T
```

In [115]:
```python
theta = 270 + np.degrees(np.arctan(4/3))

T = transformation_matrix(theta)
print(T)
```

```
[[ 0.8 -0.6  0.   0.   0.   0. ]
 [ 0.6  0.8  0.   0.   0.   0. ]
 [ 0.   0.   1.   0.   0.   0. ]
 [ 0.   0.   0.   0.8 -0.6  0. ]
 [ 0.   0.   0.   0.6  0.8  0. ]
 [ 0.   0.   0.   0.   0.   1. ]]
```

In [116]:
```
Q = np.array([436.5, -67.669, -246.929, -580.5, 259.669, -571.418])

F = T.T @ Q

print(F)
```

```
[ 308.599 -316.035 -246.929 -308.599  556.035 -571.418]
```

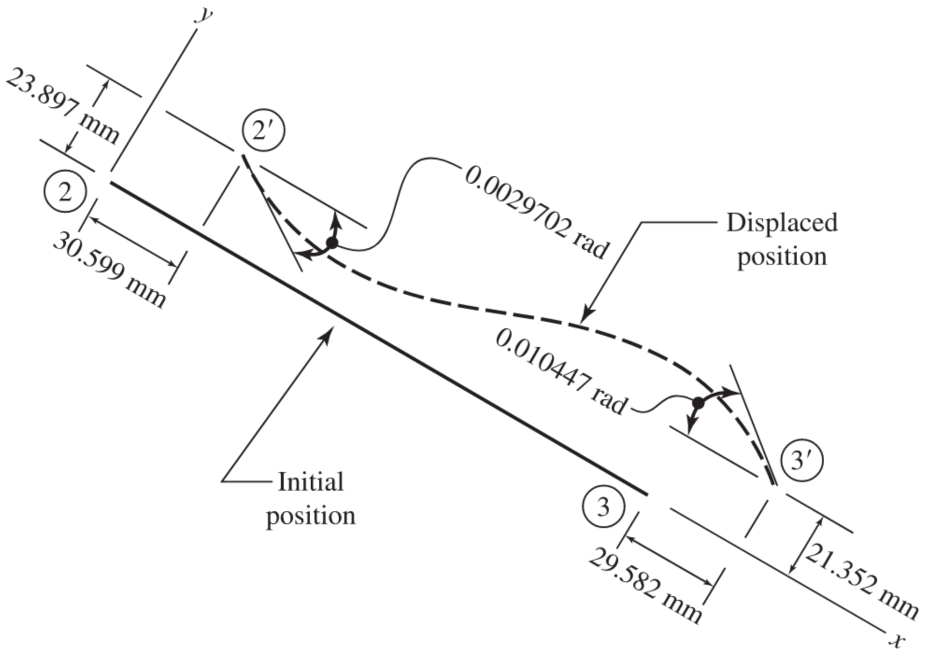


Local -> Global Element Displacements

If given local element forces from earlier problem

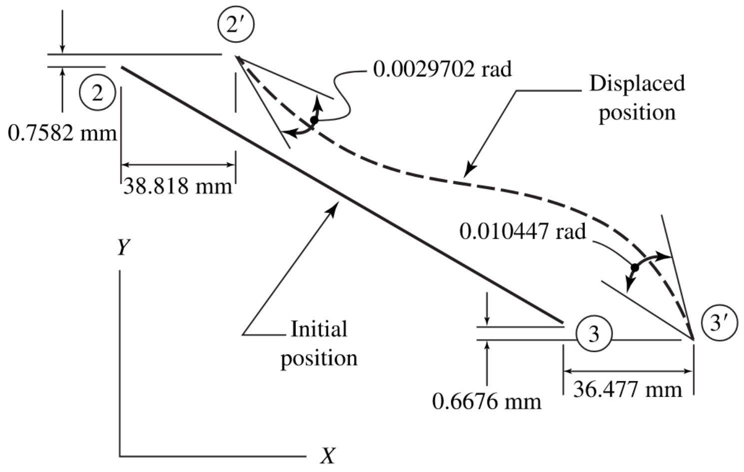
$$\mathbf{v} = \mathbf{T}^T \mathbf{u}$$

$$\mathbf{u} = \begin{bmatrix} 0.030599 & 0.023897 & -0.0029702 & 0.029582 & 0.021352 & -0.010447 \end{bmatrix}^T$$



In [117]:
```
# displacement vector (units: m, rad)
u = np.array([
    0.030599,
    0.023897,
   -0.0029702, # Clockwise rotation is negative
    0.029582,
    0.021352,
   -0.010447 # Clockwise rotation is negative
], dtype=float)

v = T.T @ u
print(v)
```

```
[ 0.039  0.001 -0.003  0.036 -0.001 -0.01 ]
```



Element Forces Using Global Stiffness Relationship

$$\mathbf{F} = \mathbf{T}^T\mathbf{k}\mathbf{T}\mathbf{v} + \mathbf{T}^T\mathbf{Q}^F$$

Given:

$$\mathbf{k} = \begin{bmatrix} 500000 & 0 & 0 & -500000 & 0 & 0 \\ 0 & 5280 & 13200 & 0 & -5280 & 13200 \\ 0 & 13200 & 44000 & 0 & -13200 & 22000 \\ -500000 & 0 & 0 & 500000 & 0 & 0 \\ 0 & -5280 & -13200 & 0 & 5280 & -13200 \\ 0 & 13200 & 22000 & 0 & -13200 & 44000 \end{bmatrix}$$

$$\mathbf{Q}^F = \begin{bmatrix} -72 & 96 & 80 & -72 & 96 & -80 \end{bmatrix}^T$$

$$\mathbf{v} = \begin{bmatrix} 0.0388174 & 0.0007582 & -0.0029702 & 0.0364768 & -0.0006676 & -0.010447 \end{bmatrix}^T$$

In [118]:
```python
k = np.array([
    [ 500000,      0,      0, -500000,      0,      0],
    [      0,   5280,  13200,       0,  -5280,  13200],
    [      0,  13200,  44000,       0, -13200,  22000],
    [-500000,      0,      0,  500000,      0,      0],
    [      0,  -5280, -13200,       0,   5280, -13200],
    [      0,  13200,  22000,       0, -13200,  44000]
], dtype=float)

K = T.T @ k @ T

print(K)
```

```
[[ 321900.8 -237465.6    7920.   -321900.8  237465.6    7920. ]
 [-237465.6  183379.2   10560.    237465.6 -183379.2   10560. ]
 [   7920.     10560.    44000.     -7920.    -10560.    22000. ]
 [-321900.8  237465.6   -7920.    321900.8 -237465.6   -7920. ]
 [ 237465.6 -183379.2  -10560.   -237465.6  183379.2  -10560. ]
 [   7920.     10560.    22000.     -7920.    -10560.    44000. ]]
```

In [119]:
```
Qf = np.array([-72, 96, 80, -72, 96, -80], dtype=float)

Ff = T.T @ Qf
print(Ff)
```

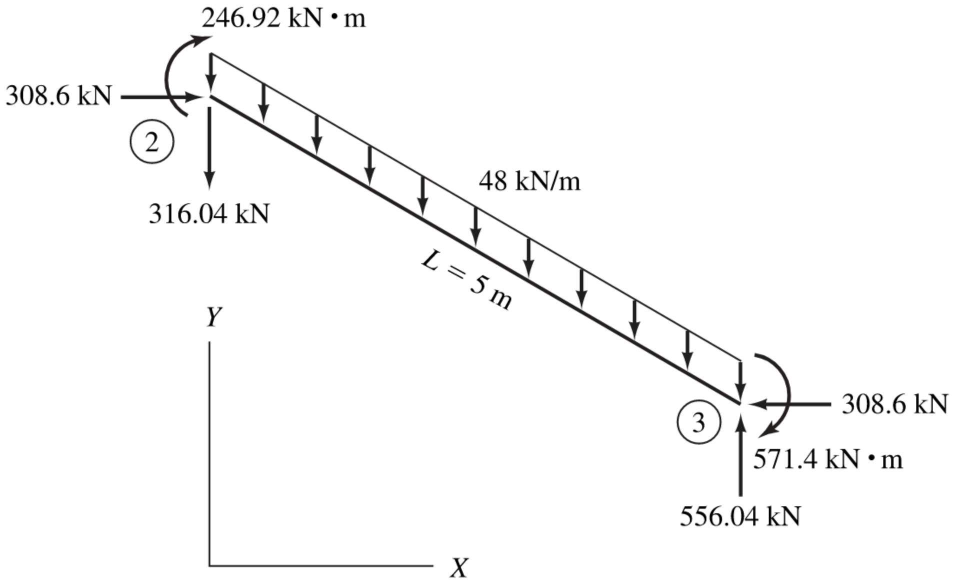
```
[   0. 120.   80.    0. 120. -80.]
```

In [120]:
```python
# Should give same result as T.T @ Qf

def transform_Qf_to_global(Qf, theta):
    """Compute F^F = T^T Q^F using the closed-form expressions."""
    theta = np.radians(theta)
    c = np.cos(theta)
    s = np.sin(theta)

    Q1, Q2, Q3, Q4, Q5, Q6 = Qf

    Ff = np.array([
        c*Q1 - s*Q2,
        s*Q1 + c*Q2,
        Q3,
        c*Q4 - s*Q5,
        s*Q4 + c*Q5,
        Q6
    ], dtype=float)

    return Ff

Ff = transform_Qf_to_global(Qf, theta)
print(Ff)
```

```
[  0. 120.  80.   0. 120. -80.]
```

In [121]:
```
v = np.array([0.0388174, 0.0007582, -0.0029702,
              0.0364768, -0.0006676, -0.010447], dtype=float)

F = K @ v + Ff
print(F)
```

```
[ 308.598 -316.036 -246.929 -308.598  556.036 -571.418]
```



Part 6 — DSM Full Procedure for Frames

Forward Pass — Structural Analysis

1. **Defining Structure**

   - Node numbering and coordinates
   - Global DOF numbering
   - Element connectivity
   - Restraints and Applied Forces

2. **Element-Level Stiffness Matrix**

   - Compute geometry: L, θ
   - Build transformation matrix, $\boldsymbol{T}$
   - Compute global element stiffness:

$$\boldsymbol{k} = \boldsymbol{T}^{\top} \boldsymbol{k}' \boldsymbol{T}$$

3. **Element-Level FEF vectors, $\boldsymbol{F^F}$**

   - Only If members loads are applied
   - identify what case, calculate closed form local FEFs, $\boldsymbol{Q^F}$
   - Compute global FEF vector (or use closed form):

$$\boldsymbol{F}^F = \boldsymbol{T}^{\top} \boldsymbol{Q}^F$$

4. **Assemble global stiffness matrix**

   - Scatter-add global element stiffness contributions into $\boldsymbol{K}$

5. **Assemble FEF load vector, $\boldsymbol{F}^{\boldsymbol{F}}$**

    - Scatter-add element-level force and moment contributions into global $\boldsymbol{F}^F$

6. **Assemble applied joint load vector, $\boldsymbol{F}$**

    - Scatter-add force and moment contributions into global $\boldsymbol{F}$
    - Only if there are loads applied at nodes

Forward Pass — Structural Analysis, cont...

5. **Apply boundary conditions**

    - Partition DOFs into free (f) and restrained (r)

6. **Solve for unknown displacements**

$$\boldsymbol{u}_f = \boldsymbol{K}_{ff}^{-1}\left(\boldsymbol{F}_f - \boldsymbol{F}_f^F - \boldsymbol{K}_{fr}\boldsymbol{u}_r\right)$$

7. **Recover support reactions**

$$\boldsymbol{F}_r = \boldsymbol{K}_{rf}\boldsymbol{u}_f + \boldsymbol{K}_{rr}\boldsymbol{u}_r + \boldsymbol{F}_r^F$$

Backward Pass — Element Recovery and Design

7. **Extract element global displacement vectors**
   - For each member, collect the relevant entries from $\boldsymbol{u}$ to form $\boldsymbol{u}_{(e)}$

8. **Transform displacements to local coordinates**

$$\boldsymbol{u}' = \boldsymbol{T}\,\boldsymbol{u}_{(e)}$$

9. **Extract element FEF vectors**
   - For each member, collect the relevant entries from $\boldsymbol{F}^{\boldsymbol{F}}$ to form $\boldsymbol{F}_{(e)}^{F}$

10. **Transform FEFs to local coordinates**

$$\boldsymbol{Q}_{(e)}^{F} = \boldsymbol{T}\,\boldsymbol{F}_{(e)}^{F}$$

11. **Compute local element end forces and moments**

$$\boldsymbol{f}' = \boldsymbol{k}'\,\boldsymbol{u}' + \boldsymbol{Q}^{F}$$

12. **Compute axial stress and bending stress (design quantities)** - With the beam end forces, you can draw shear and bending moment diagrams for each member

Wrap-Up

In this lecture, we:

Next lecture: