A Complete 3D Frame Problem

13 solutions submitted (max: Unlimited) | View my solutions

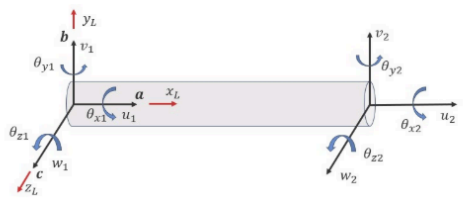

Figure 1: 3D frame elements in local axis

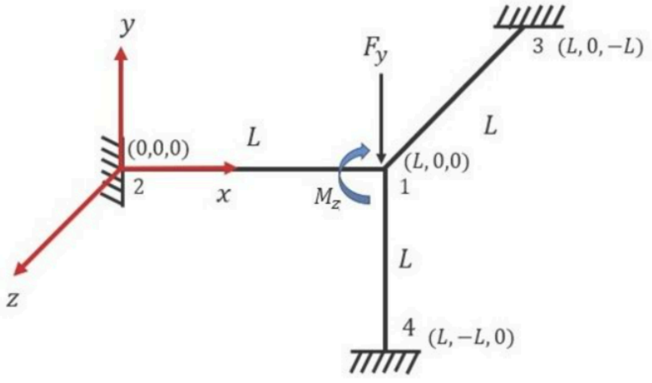

Figure 2: Structure with 3 3D frame elements

We have a structure consisting of 3 3D frame elements as shown in the figure.

We have a structure consisting of 3 3D frame elements as shown in the figure.

Frame elements have 2 nodes with 6 dofs per node. These dofs are (as shown in Figure 1) $u, v, w, \theta_x, \theta_y, \theta_z$, in that order. Thus, the 12 dofs for every element is arranged as 3 displacements and 3 rotations at each node: $\langle u_1\ v_1\ w_1\ \theta_{x1}\ \theta_{y1}\ \theta_{z1}\ u_2\ v_2\ w_2\ \theta_{x2}\ \theta_{y2}\ \theta_{z2}\rangle^T$.

The 2 noded local element is defined in the $\mathbf{a}, \mathbf{b}, \mathbf{c}$ basis (local axes are x_L, y_L, z_L) , with the element lying along $\mathbf{a}$. Bending in the $x_L - y_L$ plane causes v, θ_z, bending in the $x_L - z_L$ plane causes z, θ_y, stretching causes u and twisting causes θ_x. Thus, the 3d frame is a combination of a truss along x_L , a beam in $x_L - y_L$ plane, a beam in the $x_L - z_L$ plane and a shaft along x_L. Out of these, we have derived local stiffnesses for all except the shaft.

1. First, derive the local stiffness for a shaft element which has dofs θ_x at the nodes and can sustain torques M_x. Recall from your basic strength of materials that the torque twist relation for a shaft is $\theta_x = \frac{M_x L}{GJ}$, where G, J are the shear modulus and polar moment of inertia of the shaft.
2. Having derived the local stiffness for the shaft, put together the 12×12 local stiffness of the 3D frame in its local $x_L - y_L - z_L$ coordinate system. The moments of intertia are I_z and I_y for bending in the $x_L - y_L$ plane and I_y for $x_L - z_L$ plane. Polar moment of inertia is J. Young's modulus is E.
3. In the code that is given to you, there is a function called `frame3d` (you need to scroll down below the main code). In that function, you need to complete the matrix `Klocal` by filling in the locations marked by *.
4. You are also provided with a function `transformation` that gives you a 3×3 matrix `Q` that gives you the components of a vector in the global coordinate system to the local coordinate system. Use this matrix to construct a matrix `T` in `frame3d` that rotates `Klocal` into the global coordinate system. Keep in mind that you are dealing with a rotation that is a vector with components $\theta_x, \theta_y, \theta_z$. Again, to complete `T`, you need to complete the positions marked by *.
5. Now come back to the main code. All essential material data, nodal coordinates, element connectivity are already supplied for the problem shown. Complete the missing bits (again marked with *) in the way you have done for many assignments so far. You have to determine the destination array, assemble the global stiffness, insert the boundary conditions and find the reaction forces.

```
% material properties
E = 206.84e09; %in Pa
G = 68.9e09; %in Pa
% geometric properties of the frame elements
Iz = 4.16e-05; % in m^4
Iy = Iz;
J = Iz/2;
A = 0.00645; % in m^2
x1=[0 0 0];x2=[2.54 0 0];
% basic data
nelem = 3; %no of elements
```

```
nnode = 4; % no of nodes
L=2.54;
% nodal coordinates
coor=[L 0 0;0 0 0;L 0 -L;L -L 0];
% element connectivity
conn=[2 1;3 1;4 1];
% initialise
Kglobal=zeros(24);
Fglobal = zeros(24,1);
% boundary conditions
fixed_dof = [7:12, 13:18, 19:24] ;
% force boundary condition
Fglobal([2,6])=[-222.4 -112.98]*1e03;
% form destination array
for ielem = 1:nelem
   n1=conn(ielem,1); %1st local node
   idofs1 = [6*n1-5 6*n1-4 6*n1-3 6*n1-2 6*n1-1 6*n1];
   n2=conn(ielem,2); %2nd local node
   idofs2=[6*n2-5 6*n2-4 6*n2-3 6*n2-2 6*n2-1 6*n2];
   dest(ielem,1:12) = [idofs1 idofs2];
end
% form global stiffness matrix, element by element
for ielem = 1:nelem
   n1=conn(ielem,1);
   n2=conn(ielem,2);
   x1=coor(n1,:);
   x2=coor(n2,:);
   Klocal=frame3d(x1,x2,E,G,A,Iz,Iy,J);
   % place in the global stiffness Kglobal
   s=dest(ielem,:);
   Kglobal(s,s) = Kglobal(s,s)+Klocal;
end
% apply boundary conditions
Fprime=Fglobal;
Kgprime=Kglobal;
% Modify Kgprime and Fprime here
for d =fixed_dof
   Kgprime(d,:) = 0;  Kgprime(:,d) = 0;  Kgprime(d,d) = 1;
   Fprime(d)    = 0;
end

% solve
U = inv(Kgprime)*Fprime;
% determine reaction forces
```

```
Freac=Kglobal*U - Fprime;
% end of main code =============================================
% function to get local stiffness
function[Klocal]=frame3d(x1,x2,E,G,A,Iz,Iy,J)
% Determine the local stiffness matrix of a 3d frame element in the
global coordinate
% system. The 3d coordinates of node 1 is x1 and that of node 2 is x2.
The matrix is
% returned in Klocal, a 12X12 matrix.
L=norm(x2-x1);
Klocal=zeros(12);
k1=A*E/L;
Ktruss=[k1 -k1;-k1 k1];
k2 = (12*E*Iz)/L^3;
k3 = (6*E*Iz)/L^2;
k4 = 4*E*Iz/L;
Kbeamxy=[k2 k3 -k2 k3;k3 k4 -k3 k4/2;-k2 -k3 k2 -k3;k3 k4/2 -k3 k4];
k2 = (12*E*Iy)/L^3;
k3 = (6*E*Iy)/L^2;
k4 = 4*E*Iy/L;
Kbeamxz=[k2 k3 -k2 k3;k3 k4 -k3 k4/2;-k2 -k3 k2 -k3;k3 k4/2 -k3 k4];
k1 = A*E/L;

Kshaft=(G * J / L) * [1 -1; -1 1];
Klocal([1 7],[1 7])= Ktruss;
Klocal([2 6 8 12],[2 6 8 12]) = Kbeamxy;
Klocal([3 5 9 11],[3 5 9 11]) = Kbeamxz;
Klocal([4 10],[4 10]) = Kshaft;
Q = transformation(x1,x2);
T = zeros(12);
% form T matrix here
Q = transformation(x1, x2);
T = blkdiag(Q, Q, Q, Q);
Klocal=T'*Klocal*T;
end


% function for transforming global to local
function[Q] = transformation(c1,c2)
% c1 and c2 are 3X1 vectors denoting the coordinates of local nodes 1
and 2
% respectively. A coordinate system with base vectors a,b,c is formed so
% that a is along the vector going from x1 to x2, b = e3Xa and c = aXb,
% where e1, e2, e3 denote the global base vectors. The 3X3 matrix Q is
the
% transformation matrix, i.e. u=QU where u are the components of a
vector
% in a,b,c system while U are its components in the e1, e2, e3 system.
x1=c1(1); y1=c1(2); z1=c1(3);
x2=c2(1); y2=c2(2); z2=c2(3);


L = sqrt((x1-x2)^2 + (y1-y2)^2+ (z1-z2)^2);
l=(x2-x1)/L;m=(y2-y1)/L;n=(z2-z1)/L;
% special case: if a is parallel to e3
if l == 0 & m == 0 &  n == 1
   Q=[0 0 1;0 1 0;-1 0 0];
elseif l == 0 & m == 0 &  n == -1
   Q=[0 0 -1;0 1 0;1 0 0];
else
a=[l m n];
b=cross([0 0 1],a);
b = b/norm(b);
c = cross(a,b);
c = c/norm(c);
Q = [a;b;c];
end
end
```