A general way to plot the deflected shape of any 2D frame structure is to plot it element by element. This is achieved by the function Plot_ANY_Deflected_Shape(…) in MATLAB. The inputs to the function are:

1. *U_glob*: Once the converged displacements corresponding to Global DOFs for all load steps are determined, store them in a matrix *U_glob* as shown below –

$$U\_glob = \begin{bmatrix} U^{l=1}_{glob,1} & U^{l=2}_{glob,1} & \cdots & U^{l=nls}_{glob,1} \\ U^{l=1}_{glob,2} & U^{l=2}_{glob,2} & \cdots & U^{l=nls}_{glob,2} \\ \vdots & \vdots & \ddots & \vdots \\ U^{l=1}_{glob,ndof} & U^{l=2}_{glob,ndof} & \cdots & U^{l=nls}_{glob,ndof} \end{bmatrix}_{ndof \times nls}$$

   where $ndof$ is the number of DOFs and nls is the total number of load steps

2. *scale*: scale factor for the deflected shape
3. *ID*: is the ID array itself
4. *XY*: save all the nodal coordinates for all elements in a matrix *XY* as follows –

$$XY = \begin{bmatrix} ix_{e=1} & iy_{e=1} & jx_{e=1} & jy_{e=1} \\ ix_{e=2} & iy_{e=2} & jx_{e=2} & jy_{e=2} \\ \vdots & \vdots & \vdots & \vdots \\ ix_{e=n\_ele} & iy_{e=n\_ele} & jx_{e=n\_ele} & jy_{e=n\_ele} \end{bmatrix}_{n\_ele \times 4}$$

   Where,

   $ix_e =$ x – coordinate of i – node of element e

   $iy_e =$ y – coordinate of i - node of element e

   $jx_e =$ x – coordinate of j - node of element e

   $jy_e =$ y – coordinate of j - node of element e

5. *lsPlot*: vector of load step numbers to plot, e.g., [10:10:100]. You may not want to plot all load steps

6. *undefColor*: MATLAB color code (e.g., ‘r’, ‘g’, ‘b’, etc.) or RGB triplet (e.g., [1 0 0], [0 1 0], [0 0 1], etc.) for undeformed configuration
7. *undefLineWidth*: MATLAB line width for undeformed configuration
8. *defColor*: MATLAB color code (e.g., ‘r’, ‘g’, ‘b’, etc.) or RGB triplet (e.g., [1 0 0], [0 1 0], [0 0 1], etc.) for deformed configuration
9. *defLineWidth*: MATLAB line width for deformed configuration
10. *figNum*: A figure number for the MATLAB plot

The function first uses the *ID* Array to extract the **element end displacements in global coordinates** (*U_ele*) for load steps corresponding to the *lsPlot* vector (e.g., [10:10:100]). These are stored in a 3D matrix *U_ele* as follows –

$$U\_ele = \left[\left[\left[\begin{matrix} U_1^{l=10} & U_1^{l=20} & \cdots & U_1^{l=100} \\ U_2^{l=10} & U_2^{l=20} & \cdots & U_2^{l=100} \\ \vdots & \vdots & \ddots & \vdots \\ U_6^{l=10} & U_6^{l=20} & \cdots & U_6^{l=100} \end{matrix}\right]_{e=1}\right]_{e=2} \cdots \right]_{e=n\_ele}$$

where, *n_ele* is the number of elements in the model. This is achieved by the following lines of code:

```
nlsPlot = length(lsPlot);
ndf = size(ID,1);
n_ele = size(ID,2);
U_ele = zeros(ndf,nlsPlot,n_ele);

for ls = 1:nlsPlot
    for i = 1:n_ele
        for m = 1:ndf
            if ID(m,i) ~= 0
                U_ele(m,ls,i) = scale*U_glob(ID(m,i),lsPlot(ls));
            end
        end
    end
end
```

Deflected shapes corresponding to load steps in the *lsPlot* vector are then plotted using the *XY* matrix and the *U_ele* matrix as follows:

```
fig = figure(figNum);
hold on;
grid on;
box on;
% Undeformed Shape
for i = 1:n_ele
    undefPlot = ...
        plot([XY(i,1) XY(i,3)], [XY(i,2) XY(i,4)], 'LineStyle' ,'--',...
        'LineWidth', undefLineWidth, 'Color', undefColor);
end
% Deformed Shape
for ls = 1:nlsPlot
    figure(figNum)
    axis equal
    for i = 1:n_ele
        plot([XY(i,1)+U_ele(1,ls,i) XY(i,3)+U_ele(4,ls,i)],...
            [XY(i,2)+U_ele(2,ls,i) XY(i,4)+U_ele(5,ls,i)],...
            'ks','LineWidth',1,'MarkerFaceColor',[0.5 0.5 0.5]);
        defPlot = ...
            plot([XY(i,1)+U_ele(1,ls,i) XY(i,3)+U_ele(4,ls,i)],...
            [XY(i,2)+U_ele(2,ls,i) XY(i,4)+U_ele(5,ls,i)],...
            'LineStyle', '-', 'LineWidth', defLineWidth, 'Color', defColor);
    end
end
axis equal
```

Example function call:

```
Plot_ANY_Deflected_Shape(U_glob, 1, ID, XY, [1,5], 'k', 1.5, 'b', 1.5, 1)
Plot_ANY_Deflected_Shape(U_glob_lin, 1, ID, XY, [1,5], 'k', 1.5, 'r', 1.5, 1)
```