```
%====================================================================================
%Copyright (c) 2018 by Georgia Tech Research Corporation.
%All rights reserved.
%
%The files contain code and data associated with the paper titled 
%"A Deep Learning Approach to Estimate Stress Distribution: A Fast and
%Accurate Surrogate of Finite Element Analysis".
%
%The paper is authored by Liang Liang, Minliang Liu, Caitlin Martin, 
%and Wei Sun, and published at Journal of The Royal Society Interface, 2018.
%
%The file list: ShapeData.mat, StressData.mat, DLStress.py, im2patch.m, 
%UnsupervisedLearning.m, ReadMeshFromVTKFile.m, ReadPolygonMeshFromVTKFile.m, 
%WritePolygonMeshAsVTKFile.m, Visualization.m, TemplateMesh3D.vtk, TemplateMesh2D.vtk.
%Note: *.m and *.py files were converted to pdf files for documentation purpose.
%
%THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES,
%INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
%FOR A PARTICULAR PURPOSE.
%====================================================================================
%%
function WritePolygonMeshAsVTKFile(PolyMesh, FilePathAndName)
% save PolyMesh as vtkPolyData in *.vtk
% PolyMesh.Point: 3xN matrix
% PolyMesh.Face:  1xN cell array
% PolyMesh.PointData().Name
% PolyMesh.PointData().Data
% PolyMesh.FaceData().Name
% PolyMesh.FaceData().Data

if isempty(PolyMesh)
    error('PolyMesh is empty')    
    return
end

fid = fopen(FilePathAndName, 'W');
if fid == -1    
    error('can not open vtk file')    
    return        
end

LineStr='# vtk DataFile Version 3.0';
fprintf(fid, [LineStr '\n']);

LineStr='PolygonMesh@Matlab';
fprintf(fid, [LineStr '\n']);

LineStr='ASCII';
fprintf(fid, [LineStr '\n']);

LineStr='DATASET POLYDATA';
fprintf(fid, [LineStr '\n']);
```

```
[~, PointCount]=size(PolyMesh.Point);

LineStr=['POINTS ' num2str(PointCount) ' ' class(PolyMesh.Point)];
fprintf(fid, [LineStr '\n']);

precision=10;

for k=1:PointCount
    temp=PolyMesh.Point(:,k);
    LineStr=[num2str(temp(1), precision) ' ' num2str(temp(2), precision) ' ' num2str(temp(3), 
    precision)];
    fprintf(fid, [LineStr '\n']);
end

FaceCount=length(PolyMesh.Face);

TotalNumber=0;
for k=1:FaceCount
    temp=PolyMesh.Face{k};
    TotalNumber=TotalNumber + 1 +length(temp);
end

LineStr=['POLYGONS ' num2str(FaceCount) ' ' num2str(TotalNumber)];
fprintf(fid, [LineStr '\n']);

for k=1:FaceCount
    temp=PolyMesh.Face{k};
    temp=temp-1; % change matlab index (1-start) to c++ index (0-start)
    PointCount_k=length(temp);
    LineStr=num2str(PointCount_k);        
    for n=1:PointCount_k        
        LineStr=[LineStr ' ' num2str(temp(n), precision)];
    end
    fprintf(fid, [LineStr '\n']);
end

%PointData(n).Name
%PointData(n).Data
if isfield(PolyMesh, 'PointData')
    LineStr=['POINT_DATA ' num2str(PointCount)];
    fprintf(fid, [LineStr '\n']);    
    LineStr=['FIELD FieldData ' num2str(length(PolyMesh.PointData))];
    fprintf(fid, [LineStr '\n']);
    for n=1:length(PolyMesh.PointData)        
        LineStr=[PolyMesh.PointData(n).Name ' 1 ' num2str(PointCount) ' double']; 
        fprintf(fid, [LineStr '\n']);
        for k=1:PointCount
            temp=PolyMesh.PointData(n).Data(k);
            LineStr=num2str(temp, precision);
            fprintf(fid, [LineStr '\n']);
        end
    end
end
```

```
%FaceData(n).Name
%FaceData(n).Data
if isfield(PolyMesh, 'CellData')
    LineStr=['Cell_DATA ' num2str(FaceCount)];
    fprintf(fid, [LineStr '\n']);    
    LineStr=['FIELD FieldData ' num2str(length(PolyMesh.ElementData))];
    fprintf(fid, [LineStr '\n']);
    for n=1:length(PolyMesh.FaceData)
        LineStr=[PolyMesh.ElementData(n).Name ' 1' num2str(FaceCount) ' double'];    
        fprintf(fid, [LineStr '\n']);
        for k=1:FaceCount
            temp=PolyMesh.FaceData(n).Data(k);
            LineStr=num2str(temp, precision);
            fprintf(fid, [LineStr '\n']);
        end
    end
end

fclose(fid);
```