Overview

This is a documentation for executable file Quad2Spline.exe, which is used to generate BEXT file for surface mesh.

I/O

Input: manifold unstructured quad mesh in the **vtk format.**

- If it is a closed surface mesh, at least **3-ring neighborhood** apart for any two extraordinary points (EPs).
- If it is an open surface mesh, any EP needs to be **4-ring neighborhood** apart from the boundary, and 3-ring neighborhood apart from another EP.

Output

- Bezier extraction for surface mesh into BEXT file format for LS-DYNA (XXX_BEXT.txt).

Usage of executable file:

User can run the executable file "Quad2Spline.exe" through command line.

Here, we will use the following file structure to explain the usage of the program:

Quad2Spline/

Quad2Spline.exe

example/

pipe_junction_global0/

pipe_junction_global1/

The options to run the code are explained as follow:

- "-h" or "--help": User can use this option to check the help information
  Example: Quad2Spline.exe -h

```
PS C:\Users\LAR\Dropbox\GEM\Software Package\Shell> .\Quad2Spline.exe -h
CMU Surface Quad To Spline BEXT Generation
Usage:
  C:\Users\LAR\Dropbox\GEM\Software Package\Shell\Quad2Spline.exe [OPTION...]

General options:
  -h, --help            Print help
  -g, --globalref arg   Set the level of global refinement, default is 0
  -l, --localref arg    Set the level of local refinement, default is 0
  -i, --input arg       Input mesh file (.vtk format)
```

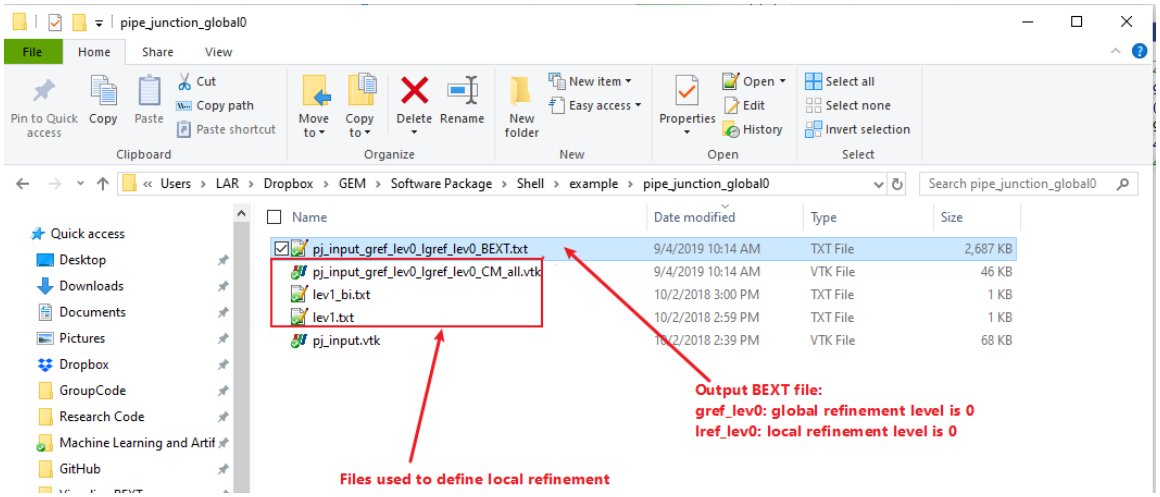
- "-i" or "--input" arg: User need to set the input mesh file using this option:
  Example: Quad2Spline.exe -i .\example\pipe_junction_global0\pj_input.vtk

```
PS C:\Users\LAR\Dropbox\GEM\Software Package\Shell> .\Quad2Spline.exe -i .\example\pipe_junction_global0\pj_input.vtk
1100 750Bezier extracting...
Writing file...
500 End of writing!
```

  The program will use the folder that includes the input mesh as the work directory and output two files here:

  XXX_BEXT.txt: Bezier extraction information for constructed surface spline.
  XXX_CM_all.vtk: User needs this file to define local refinement.

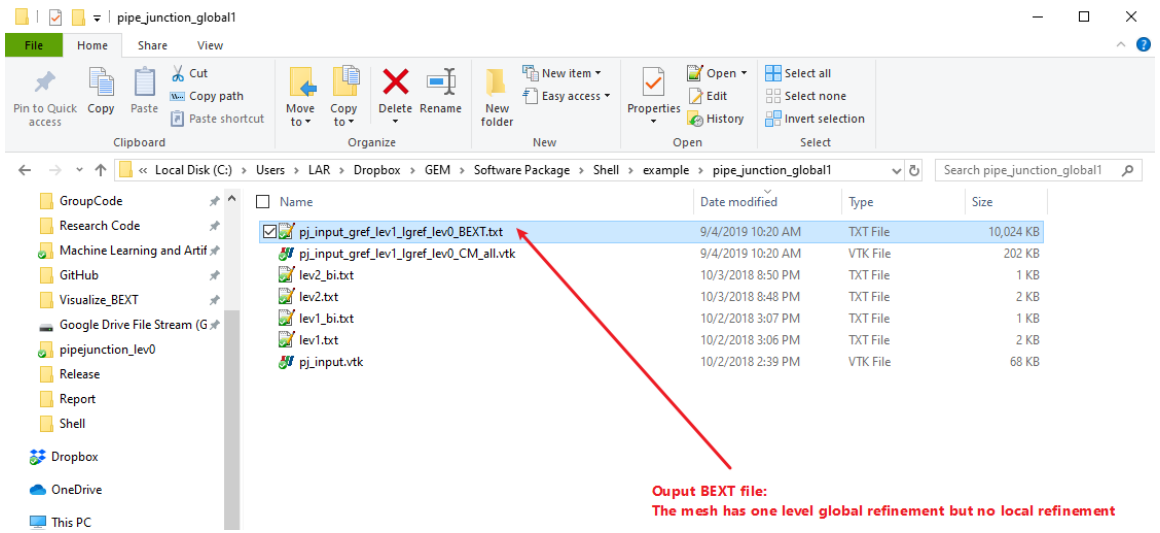
The output using this command is shown below:



- "-g" or "--globalref" arg: User can use this option to set the level of global refinement. The default level is 0, which means that no global refinement is implemented.
Example: Quad2Spline.exe -i .\example\pipe_junction_global1\pj_input.vtk -g 1

```
PS C:\Users\LAR\Dropbox\GEM\Software Package\Shell> .\Quad2Spline.exe -i .\example\pipe_junction_global1\pj_input.vtk -g 1
1100 750Bezier extracting...
Writing file...
500 1000 1500 2000 2500 3000 End of writing!
```

The program will output "XXX_gref_lev1_lref_lev0_BEXT.txt" file, which means that we apply global refinement once. The output using this command is shown below:

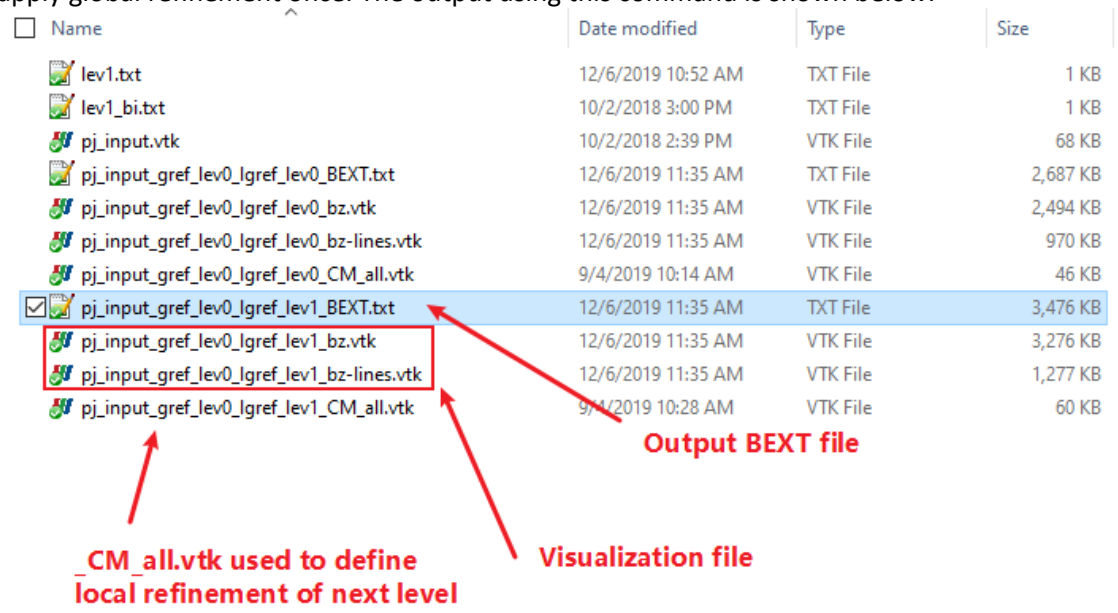


- "-l" or "--localref" arg: User can use this option to set the level of local refinement. The default level is 0, which means that no local refinement is implemented. Note that the program needs another two input files: **levX.txt** and **levX_bi.txt** to define the local refinement region. The definition of local refinemet elements is explained in next chapter.

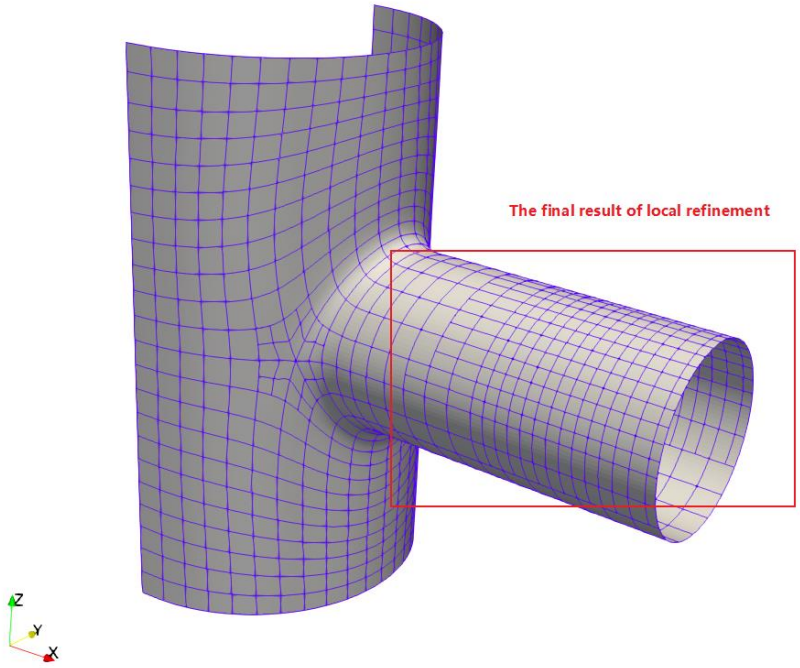
Example: Quad2Spline.exe -i .\example\pipe_junction_global0\pj_input.vtk -g 0 -l 1

```
PS C:\Users\LAR\Dropbox\GEM\Software Package\Shell> .\Quad2Spline.exe -i .\example\pipe_junction_global0\pj_input.vtk -g 0 -l 1
1100 750Bezier extracting...
Writing file...
500 1000 End of writing!
```

The program will output "XXX_gref_lev0_lref_lev1_BEXT.txt" file, which means that we apply global refinement once. The output using this command is shown below:



The local refinement result can be visualized by using two VTK files ends with "_bz.vtk" and "_bz-lines.vtk", the visualization is shown below:

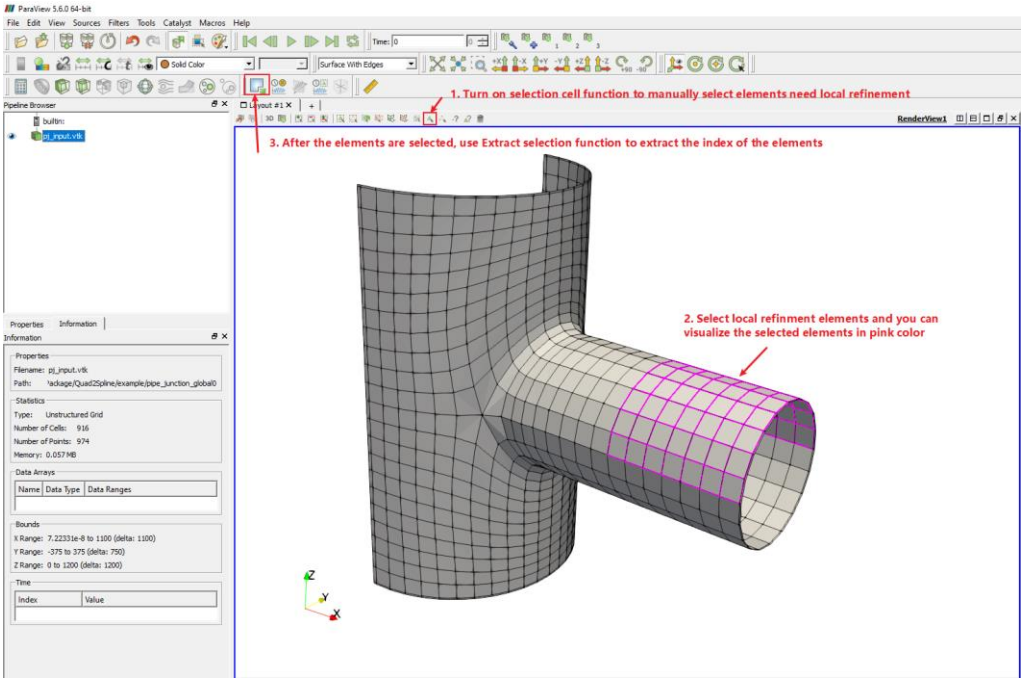


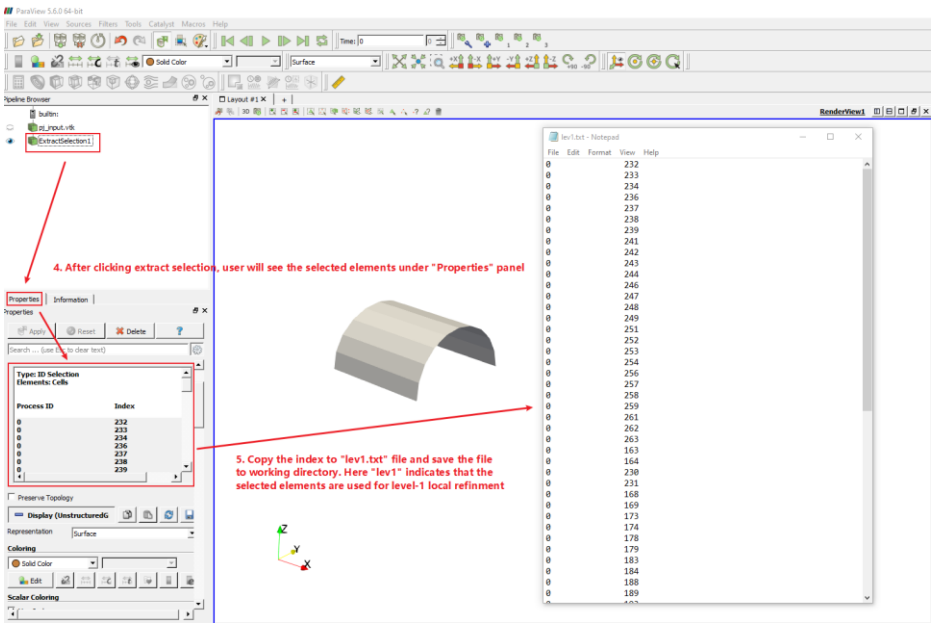
Manual selection of elements to be refined

An element can be subdivided into 4 or 2 subelements, referred to as Type-4 and Type-2 refinement, respectively. Type-4 refinement involves elements desired to be refined whereas Type-2 refinement is aimed to resolve intersections of T-junction extensions.

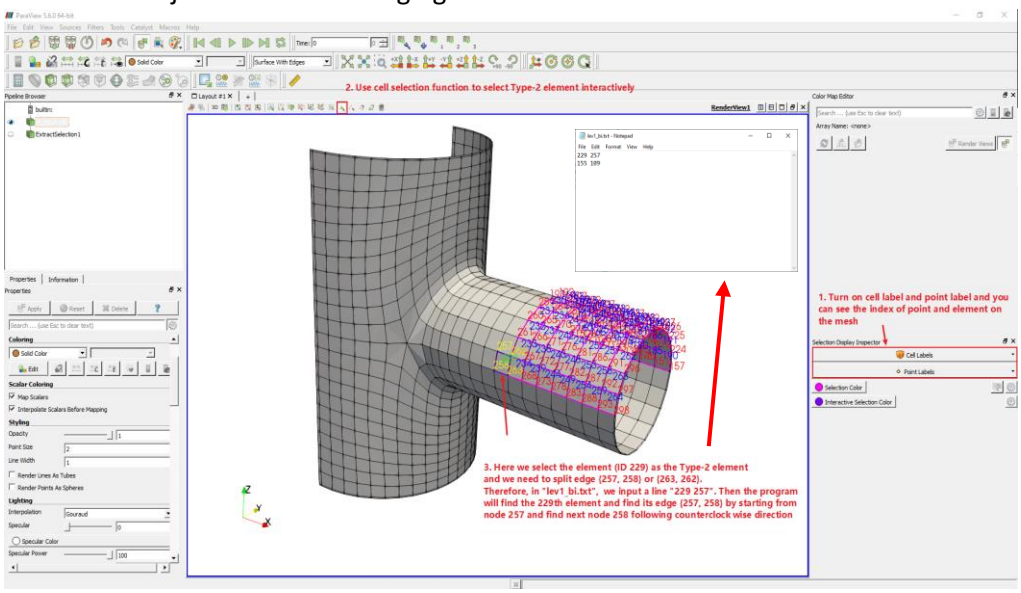
Here are several important implications.

- No boundary element should be selected.
- Use XXX_CM_all.vtk to get the element indices, which includes all the elements in the T-mesh, no matter being active (act==1) or passive (act==0). The active elements and passive elements can be visualized with different colors in Paraview. Note that only active elements will be used to represent geometry or perform simulation.
- Manually select elements to be refined (Type-4 refinement) using e.g., Paraview, and list their indices in a file named “levX.txt” (“X” indicates a specific level number). The procedures are shown in the figures below:





- Manually select and list element indices and directions of Type-2 refinement in “levX_bi.txt”. The direction of Type-2 refinement is specified by a corner vertex index. For example, an element indexed as “A” consists of four corner vertices, with indices written in a counterclockwise manner as {i,j,k,l}. If the element is to be bisected along across the edge {i,j}, then write in “levX_bi.txt” as “A i”. Similarly, when split across the edge {j,k}, then write “A j”. See the following figure for more details.



- To define local refinement after global refinement, user needs to run global refinement first and define the local refinement use output XXX_CM_all.vtk.
  Example: Quad2Spline.exe -i .\example\pipe_junction_global1\pj_input.vtk -g 1 -l 1
- To define multiple levels of local refinement, user needs to create input files **levX.txt** and **levX_bi.txt** level by level using output XXX_CM_all.vtk from previous level.
  Example: Quad2Spline.exe -i .\example\pipe_junction_global0\pj_input.vtk -g 0 -l 2

References

[1] W. Wang, Y. Zhang, G. Xu, T. J. R. Hughes. **Converting an Unstructured Quadrilateral/Hexahedral Mesh to a Rational T-Spline.** *Computational Mechanics*, 50(1):65-84, 2012.

[2] X. Wei, Y. J. Zhang, L. Liu, T. J. R. Hughes. **Truncated T-splines: Fundamentals and Methods**. *Computer Methods in Applied Mechanics and Engineering Special Issue on Isogeometric Analysis*, 316:349-372, 2017.

[3] Y. Lai, Y. J. Zhang, L. Liu, X. Wei, E. Fang, J. Lua. **Integrating CAD with Abaqus: A Practical Isogeometric Analysis Software Platform for Industrial Applications**. *A Special Issue of HOFEIM 2016 in Computers and Mathematics with Applications*, 74(7):1648-1660, 2017.

[4] H. Casquero, X. Wei, D. Toshniwal, T. J. R. Hughes, J. Kiendl, Y. J. Zhang, **Seamless integration of design and Kirchhoff-Love shell analysis using analysis-suitable unstructured T-splines.** *Submitted to CMAME*.