3D Mesh tools in FreeFem++

J. Morice and F. Hecht

Laboratoire Jacques-Louis Lions (LJLL)

General Information

- 3D Mesh :
  - Volume Element : Tetrahedrons
  - Surface Element : Triangles
  - Line Elements (not yet implemented)

- For 3D mesh tools, Load the library " msh3" $\Rightarrow$ `load "msh3"` in top of .edp file.

- Read mesh

```
mesh3 Th =readmesh("filename") ;
mesh3 Th("filename") ;
```

- file extension : .mesh, .meshb : Mesh Format File of Medit (P. Frey LJLL)
  .msh : (see manual) $\neq$ data file of Gmsh (Mesh generator)

- Save mesh

```
savemesh(Th,"filename") ;
```

- file extension : .mesh, .meshb .msh

Simple Function

– comand change : change label of elements in a 2D/3D mesh

parameter of 3D mesh :

`reftet =` is a vector of integer that contains the old labels and the new labels of tetrahedrons.

`refface =` is a vector of integer that contains the old labels and the new labels of triangles.

Definition of these vectors

$$[V_1, ..., V_{n_l}].$$

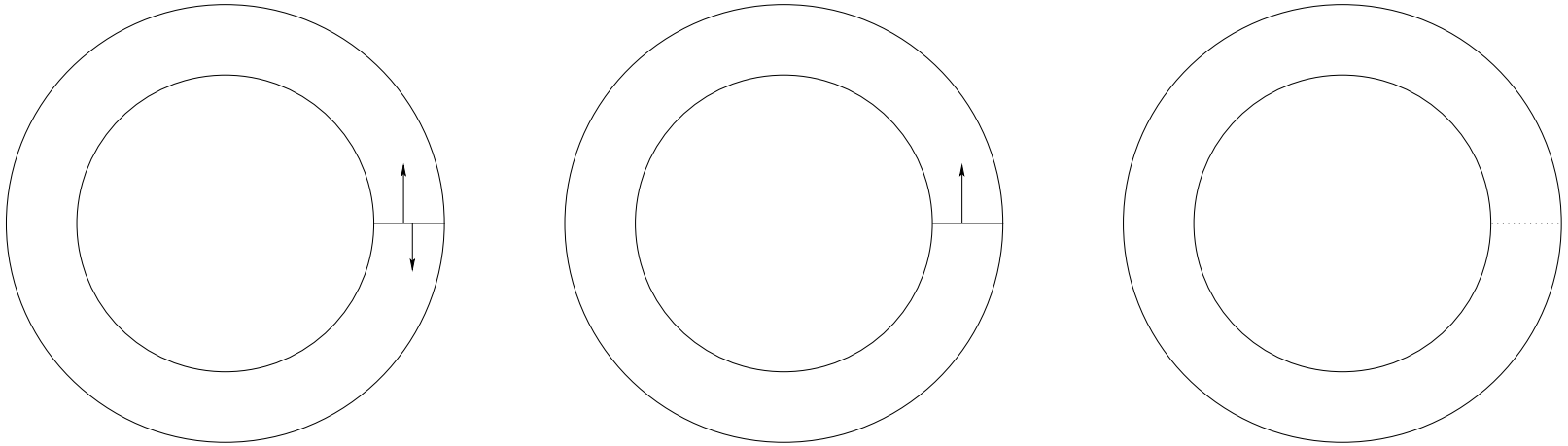
where the sub vector V_i is defined by $V_i =$ [old label of set i, new label of set i]

```
int[int] r1=[2,0] ;
Th3=change(Th3,refface=[1,135]) ; //    change the triangle of label 1 into
label 135.
```

– These parameters are using post-processing in different 3D mesh tools.

Simple Function (3)

– movemesh : move mesh a three dimensional.
parameter :
`transfo` the displacement vector of transformation `[Φ1, Φ2, Φ3]`
`reftet` and `refface` parameter to change label.
`facemerge` integer value.



facemerge=0 facemerge=1 facemerge=2

`ptmerge` criterion to define two merging points.

Simple Function (3)

– movemesh23 : tranform a 2D mesh in 3D Surface mesh.

```
mesh3 Th3 = movemesh23(Th2,transfo=[Φ1,Φ2,Φ3]) ;
```

parameter :

`transfo` the displacement vector of transformation `[Φ1, Φ2, Φ3]`

`refface` parameter to change face's label.

`orientation` orientation of the normal of the surface mesh. The normal of the initial two dimensionnal mesh is orientated in the $z > 0$. Set `orientation`= 1 if you want the surface mesh oriented by this transformed normal. Otherwise, set this parameter to -1 to obtain the opposite normal. By default, this parameter is 1.

`ptmerge` Criterion to define two merging points.

Interface with 3D mesh Generator

- Tetgen developped by Hang Si (Weierstrass Institute, Berlin). `http://tetgen.berlios.de/`
  - Delaunay tetrahelizations.
  - Construction of Convex hull for a set of points.
- Netgen developped by J. Schöberl (Univ.) (`http://sourceforge.net/projects/netgen-mesher/`.
  - Graphics inteface with all functions : surface mesh, parallel meshing, anisotropic mesh
- Gmsh developped by C. Geuzaine (Univ. Liège)/ J-F.Remacle (Univ. Louvain) `http://www.geuz.org/gmsh/`
  - Graphics Interface
  - external 3D mesh generator : Netgen/Tetgen.

Gmsh

- No Library for Interface with other software generating by Gmsh.
- Load 2D/3D mesh :

```
mesh3 Th3=  gmshload("filename.msh");
```

- Remark : Medit Format of 3D mesh can be reading by Gmsh.

Appear in the next release.

Netgen

- Library interface with software $\Rightarrow$ not update.
- Load 2D/3D mesh (standardized file format) :

```
mesh3 Th3 =  netgload("filename.stl");
```

- Internal call of Netgen :
  - Compile netgen interface (file examples++-load/netgen.cpp)
    ./ff-c++ -I/DirectoryHeaderNetgen netgen.cpp -L/DirectoryNglib/ -lnglib
  - Two functions :
    1 :Tetrahelization of 3D volume

```
mesh3 Th3 = netg(ThSurf3D, maxh=,  meshsizefilename=);
```

    2 : Load STL Geometry + Tetrahelization of 3D volume

```
mesh3 Th3 = netgstl("filename.stl", maxh=,  meshsizefilename=);
```

    parameters : $maxh$(double) maximum size of edges,
    $meshsizefilename$(string) name of the file containing mesh size.
  - example : netgencube.edp

Tetgen

- call of Tetgen : tetg, tetgtransfo, tetgreconstruction, tetgconvexhull.
- Compile tetgen interface : in directory examples++-load make tetgen.dylib (mac) / make tetgen.so (unix) / make tetgen.dll (windows)
- tetg Tetrahelization of 3D volume

```
mesh3 Th3 = tetg(ThSurf, ...);
```

parameters :

`refface` same parameters as the keyword `change`.

`switch` = string parameter switch of Tetgen.

`nbofholes`= Number of holes.

`holelist` = $[x_1, y_1, z_1, x_2, y_2, z_2, \cdots,]$ where x_i, y_i, z_i is a point inside the i^{th} hole.

`nbofregions` = Number of regions.

`regionlist` = $[x_1, y_1, z_1, Lab_1, mvol_1, x_2, y_2, z_2, Lab_2, mvol_2, \cdots]$. where x_i, y_i, z_i is a point inside the region, Lab_i and $mvol_i$ are the label and the maximum volume of domain i.

`nboffacetcl`= Number of facets constraints.

`facetcl`= $[Lab_1^{fc}, marea_1^{fc}, Lab_2^{fc}, marea_2^{fc}, \cdots]$. The i^{th} facet constraint is defined by the facet label Lab_i^{fc} and the maximum area for faces $marea_i^{fc}$.

Tetgen : example tetgenholeregion.edp

```
                                        //    file ‘‘tetgenholeregion.edp’’
load "msh3’’
load "tetgen"

mesh Th=square(10,20,[x*pi-pi/2,2*y*pi]) ;          //    ]-pi/2, pi/2[×]0, 2π[
                                        //    a parametrization of a sphere
func f1 =cos(x)*cos(y) ;
func f2 =cos(x)*sin(y) ;
func f3 = sin(x) ;

                  //    construction of an adaptative mesh for the surface
....
Th=adaptmesh(Th,m11*vv,m21*vv,m22*vv,IsMetric=1,periodic=perio) ;
```

Tetgen : example tetgenholeregion.edp(2)

```
                                    //    construction of the surface of spheres
real Rmin  = 1. ;
func f1min = Rmin*f1 ;
func f2min = Rmin*f2 ;
func f3min = Rmin*f3 ;

mesh3 Th3sph = movemesh23(Th,transfo=[f1min,f2min,f3min],orientation=1) ;

real Rmax  = 2. ;
func f1max = Rmax*f1 ;
func f2max = Rmax*f2 ;
func f3max = Rmax*f3 ;

mesh3 Th3sph2 = movemesh23(Th,transfo=[f1max,f2max,f3max],orientation=-1) ;

cout << "addition" << endl ;
mesh3 Th3 = Th3sph+Th3sph2 ;
```

Tetgen : example tetgenholeregion.edp(3)

```
real[int] domain2 = [1.5,0.,0.,145,0.001,0.5,0.,0.,18,0.001] ;
cout << " tetgen call without hole " << endl ;
mesh3 Th3fin =
tetg(Th3,switch="paAAQYY",nbofregions=2,regionlist=domain2) ;
savemesh(Th3fin,"spherewithtworegion.mesh") ;

real[int] hole = [0.,0.,0.] ;
real[int] domain = [1.5,0.,0.,53,0.001] ;
cout << "finish tetgen call with hole   " << endl ;
mesh3 Th3finhole=tetg(Th3,switch="paAAQYY",nbofholes=1,holelist=hole,
nbofregions=1,regionlist=domain) ;
savemesh(Th3finhole,"spherewithahole.mesh") ;
```

Tetgen : tetgreconstruction

– tetgreconstruction : allow to refine a 3D mesh
  parameters : tetg : switch, nbofregions, regionlist, nboffacetcl and facetcl
  change : reftet and refface
  sizevolume= a function to constraint maximum volume size.
  Example : examples++-load/refinesphere.edp

```
real[int] domain = [0.,0.,0.,145,0.01] ;
mesh3 Th3sph=tetg(Th3,switch="paAAQYY",nbofregions=1,regionlist=domain) ;

int[int] newlabel = [145,18] ;
real[int] domainrefine = [0.,0.,0.,145,0.0001] ;
mesh3 Th3sphrefine=tetgreconstruction(Th3sph,switch="raAQ",reftet=newlabel,
nbofregions=1,regionlist=domainrefine) ;

int[int] newlabel2 = [145,53] ;
func fsize = 0.01/(( 1 + 5*sqrt( (x-0.5)^2+(y-0.5)^2+(z-0.5)^2) )^3) ;
mesh3 Th3sphrefine2=tetgreconstruction(Th3sph,switch="raAQ",reftet=newlabel2
sizeofvolume=fsize) ;
```

Tetgen

– tetgtransfo $\sim$ movemesh23 + tetg

```
mesh3 Th3= tetgtransfo(Th2,transfo=[f1,f2,f3]) ;
```

equivalent to

```
mesh3 Th3="movemesh23(Th2,transfo=[f1,f2,f3)] ;
Th3= tetg(Th3) ;
```

parameters : movemesh23 + tetg

– tetgtconvexhull : construction of the convex hull of set of points and tetrahelization of this domain.

```
real[int] x(nv),y(nv),z(nv) ;                    //    nv=number of vertex
...  ...                                  //    definition of coordinate of vertex
mesh3 Th3= tetgconvexhull([x,y,z], reftet=1,refface=2) ;
```

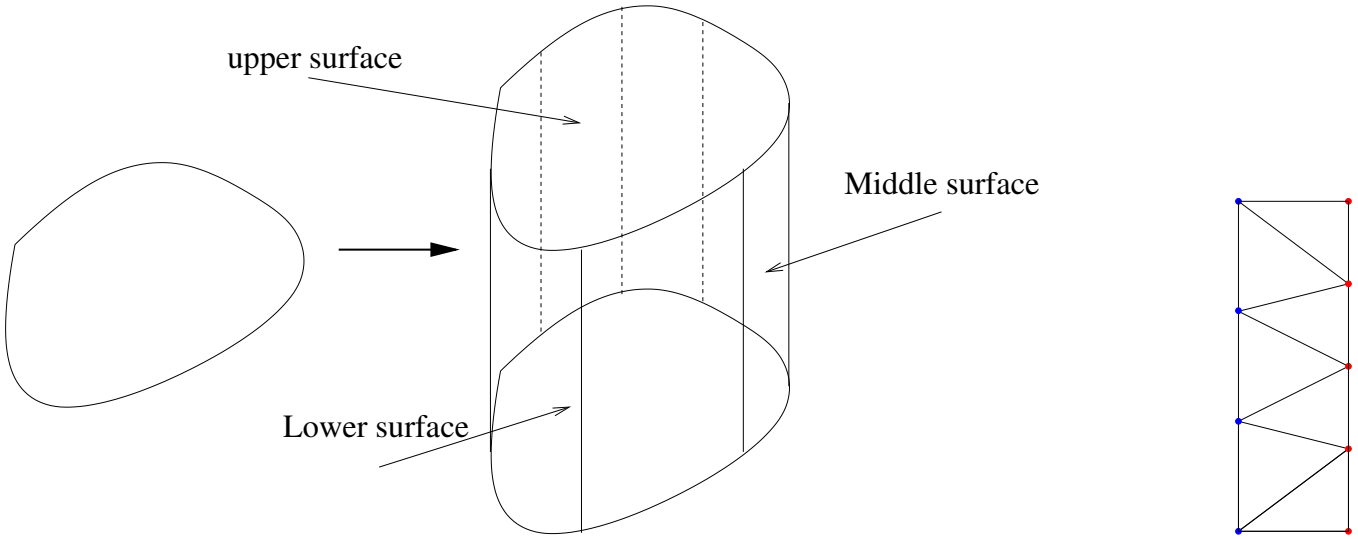
other parameters :

`switch` = The command line *switch* of Tetgen.

In the string switch, we can't used the option 'p' and 'q' of tetgen.

Layermesh

– buildlayers : Layer mesh with degenerate elements



The arguments of `buildlayers` *is a two dimensional mesh and the number of layers* M.

```
mesh3 Th3= buildlayers(Th2,M,zbound=[zmin,zmax] ,....) ;
```

parameters

`zbound` = [zmin,zmax] where zmin and zmax are the lower surface mesh and upper mesh of surface mesh.

`coef` = A function expression between [0,1]. The number of elements generated by vertex V_i^{2d} is the integer part of $coef(V_i^{2d})M$.

Layermesh

– Element generates
  Triangle of 2D mesh ⇒ Triangles of lower and upper surfaces and tetrahedrons in the domain.
  Edges of intial mesh ⇒ Triangles of middle surface mesh.

`reftet` = This vector is used to initialized the labels of tetrahedrons. This vector contains these labels and labels of triangles of the 2D mesh.

`reffacemid` = This vector is used to initialized the labels of triangles of the layermesh of middle surface mesh. This vector contains these labels and labels of edges of the 2D mesh.

`reffaceup` = This vector is used to initialized the label of triangles of the layermesh of upper surface mesh. This vector contains these labels and labels of triangles of the 2D mesh.

`reffacelow` = This vector is used to initialized the label of triangles of the layermesh of lower surface mesh. This vector contains these labels and labels of triangles of the 2D mesh.

add post processing parameters that allow to move the mesh. These parameters correspond to parameters `transfo`, `facemerge` and `ptmerge` of the command line `movemesh`.

Layermesh

```
load ‘‘msh3’’
int nn=5 ;
border cc(t=0,2*pi){x=cos(t) ;y=sin(t) ;label=1 ;}
mesh Th2 = buildmesh(cc(100)) ;
fespace Vh2(Th2,P2) ;
Vh2 ux,uy,p2 ;
int[int] rup=[0,2], rdlow=[0,1], rmid=[1,1] ;
func zmin = 2-sqrt(4-(x*x+y*y)) ;
func zmax = 2-sqrt(3.) ;

mesh3 Th = buildlayers(Th2,nn,
  coeff = max((zmax-zmin)/zmax, 1./nn),
  zbound=[zmin,zmax],
  reffacemid=rmid ;
  reffaceup=rup ;
  reffacelow=rlow) ;
medit(‘‘Lac",Th,wait=1) ;
```

Conclusion and Future Work

- Conclusions :
  - Interface with 3D generator : Tetgen, Netgen, Gmsh
  - A Layer Mesh
- Future Works :
  - Anisotropic mesh generator inside Freefem++.