Optimising mesh features in point could reconstruction

Ugo PELISSIER

April 4, 2023

---

1 Problem description

For both the surface mesh (2D) and the volume mesh (3D), build an interactive visualisation software that will:

- Allow for the boundary surface visualisation by default
- Provide a way to introspect its interior in terms of 3D cells
- Compute and visualise various criteria that have driven the reconstruction, in each case ((2) and (3))

Is there a way to have a consistent meshing for both the boundary surface and the inner domain ?

2 Mesh features

2.1 Data creation with Gmsh

At first, the generation of data with the CGAL software was not successful. It was therefore decided to generate a test case using the open source mesh software Gmsh.

We generated a geometrical shape from the union of two basic geometrical shapes: a cube and a sphere.

We generated a surface mesh and a volume mesh. The final mesh was saved in *.vtk* format to be manipulated later.

```
SetFactory("OpenCASCADE");
R = DefineNumber[ 1.4 , Min 0.1, Max 2, Step 0.01, Name "Parameters/Box dimension" ];
Rt = DefineNumber[ R*1.25, Min 0.1, Max 2, Step 0.01, Name "Parameters/Sphere radius" ];

Box(1) = {0,0,0, 1,1,1};
Sphere(2) = {1,0.5,0.5,0.5};
BooleanUnion(3) = { Volume{1}; Delete; }{ Volume{2}; Delete; };
```

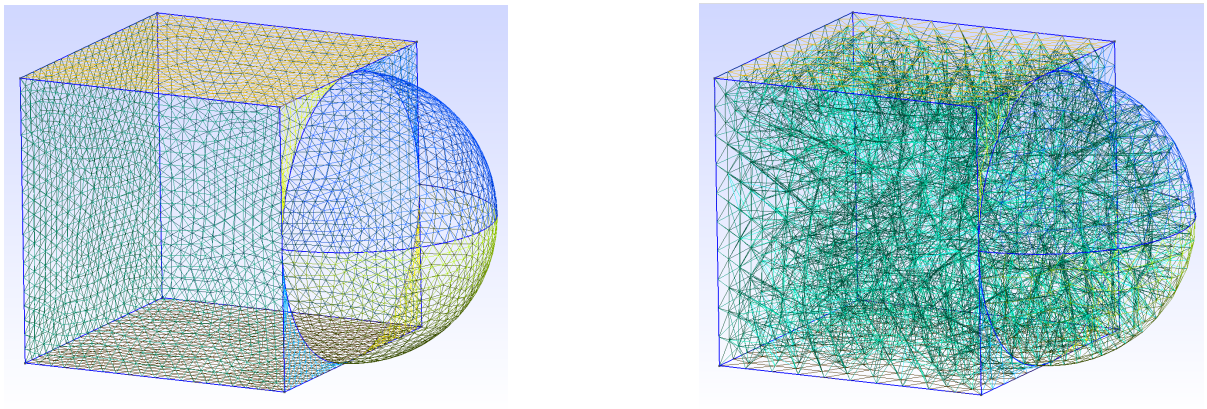


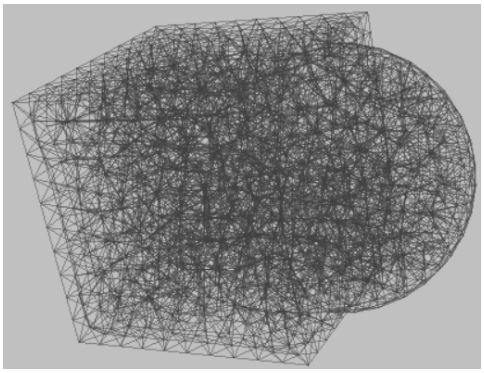
Figure 1: 2D and 3D mesh obtained with Gmsh

2.2 Visualising the boundary surface

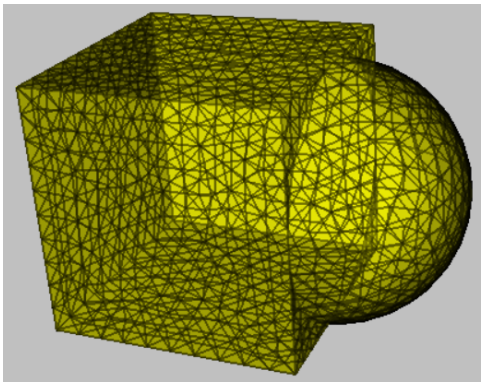
We then tried to visualise the boundary surface using VTK. First, we simply visualised the 3D mesh of the object again. using the *vtkExtractEdges* filter (fig. 2a).

Then we tried to visualise the surface of the object as well as the mesh at this surface. For that, we used the *vtkDataSetSurfaceFilter* by using a different color for the surface and the surface mesh (fig. 2b).

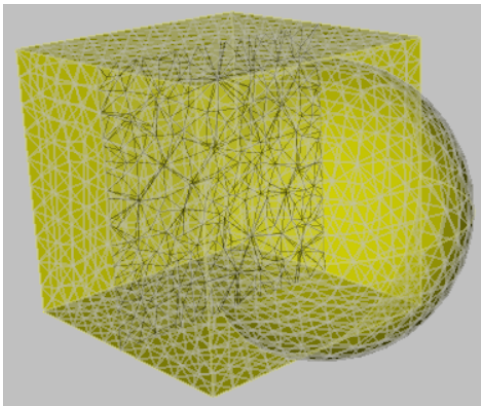
Finally, we made a cut in a longitudinal plane to observe the projection of the 3D mesh on this surface. We used the *vtkCutter* filter to do this work (fig. 2c).



(a) *vtkExtractEdges*



(b) *vtkDataSetSurfaceFilter*



(c) *vtkCutter*

Figure 2: Visualising the boundary surface

2.3 Dynamic clipping

Finally, we tried to implement a mobile cutting plane with which the user can interact. We used the *vtkImplicitPlaneWidget2* filter. This required the implementation of a callback function to handle the interaction with the user.

The interaction with the user does not work as expected, and more importantly the filter does not perform a cut even when positioned on the object (fig. 3).

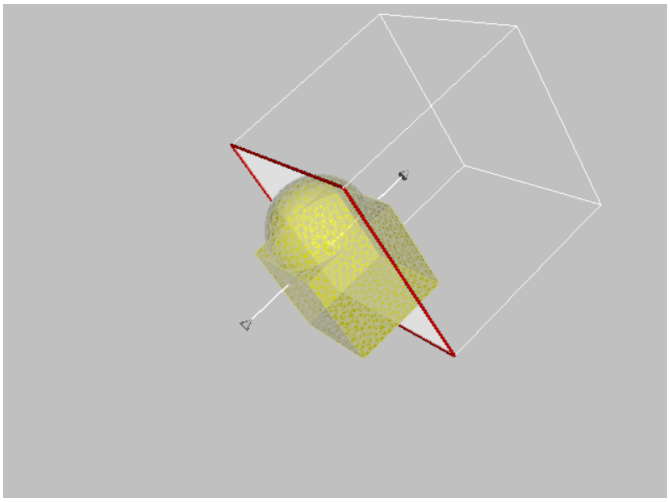


Figure 3: Dynamic clipping