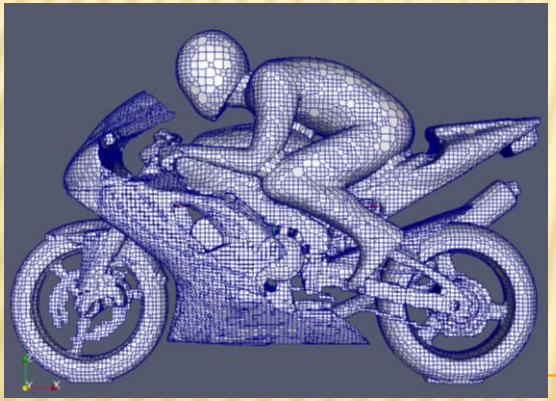
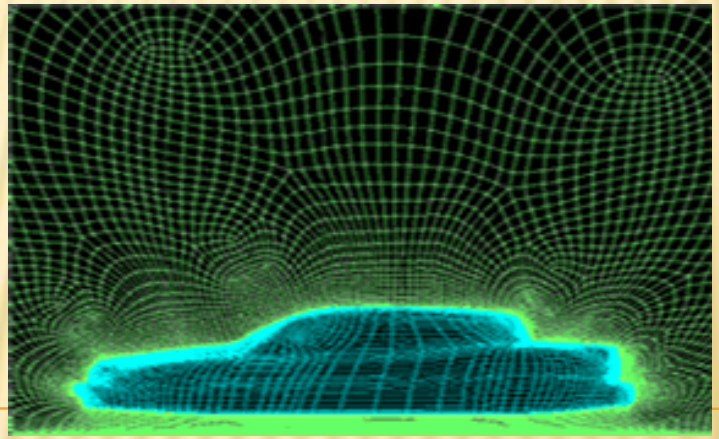
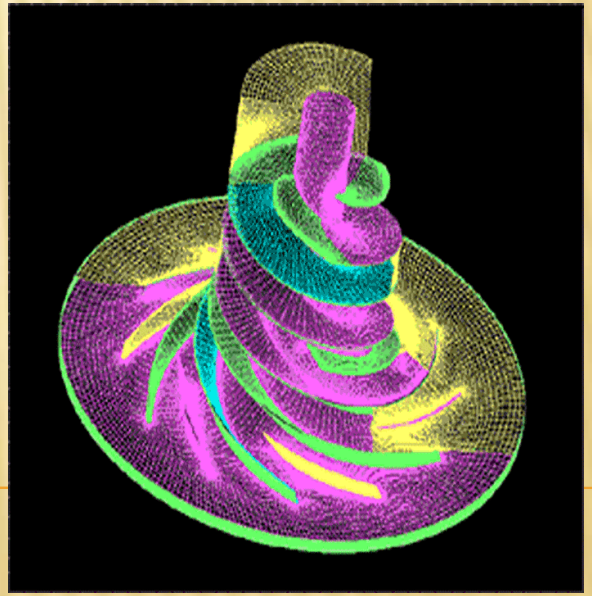
Pre-processing in openfoam, mesh generation.





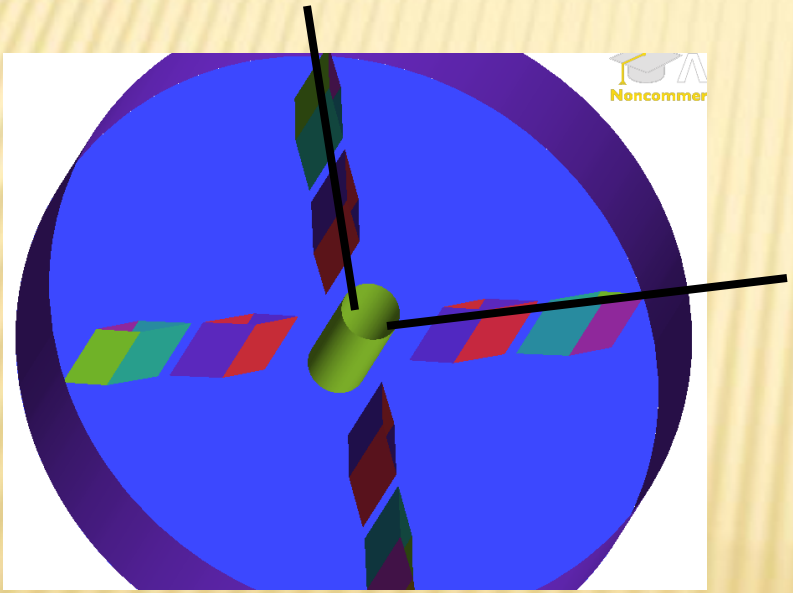


Different ways of creating the mesh. Outline

- Using **SnappyHexMesh**, an OpenFOAM mesh generation tool.
- Using **blockMesh**.
- Importing the mesh from **external software**.

A tutorial for snappyHexMesh.

- SnappyHexMesh generates a 3D mesh from a .stl file. (triangulated surface geometry)
- For this tutorial, a simplified pump geometry is chosen.



For more simplicity in computations, the symmetry of the geometry is used, and only one quarter of the pump is meshed.

A tutorial for snappyHexMesh.

- **Download** the tutorial from Håkan's webpage.
- **Source** OpenFOAM 1.6.x with alias or

  . /chalmers/sw/unsup64/OpenFOAM/OpenFOAM-1.6.x/etc/bashrc

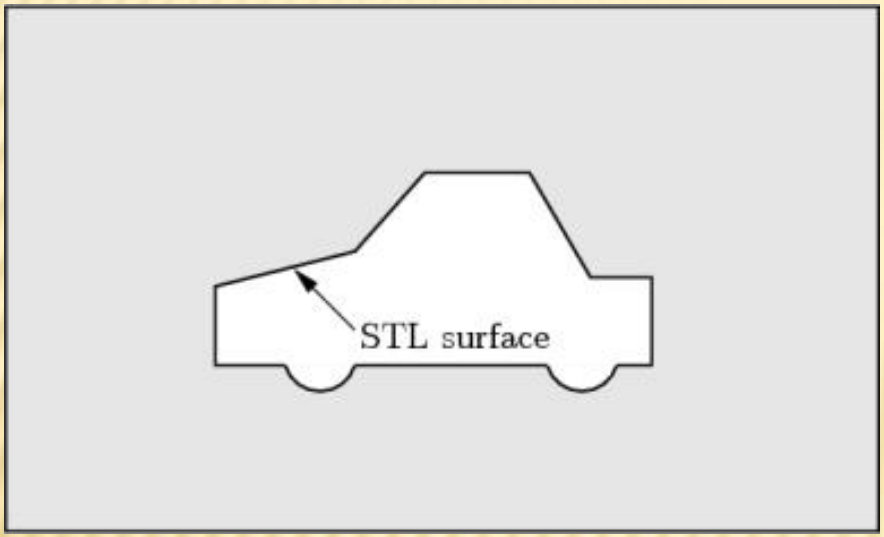
- To check if the right OpenFOAM was called:
  which SimpleFoam
  It should point to simpleFoam in OpenFOAM-1.6.x

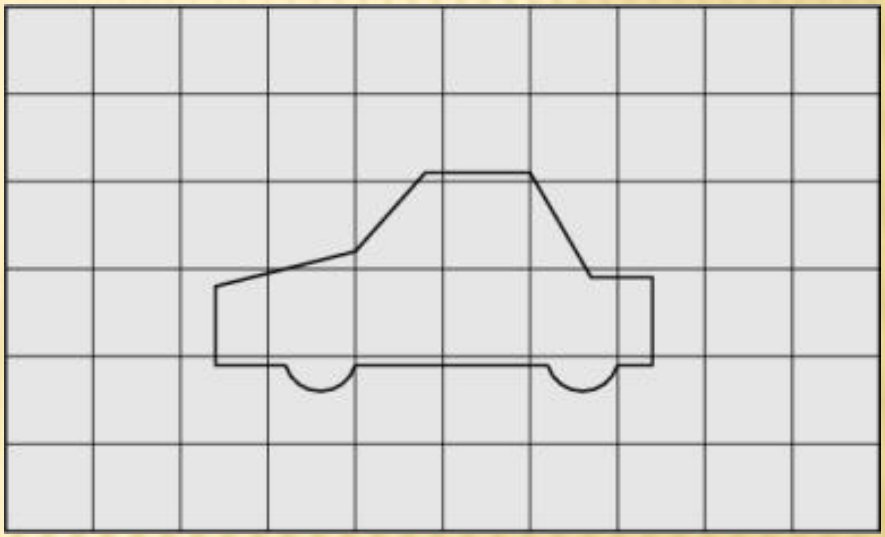
- In the tutorial case, you should find:

  1. The .stl file located in constant/triSurface.
  2. A dictionnary called snappyHexMeshDict in system/.

snappyHexMesh: step 1

- **Creation of a grid surrounding the stl surface.**



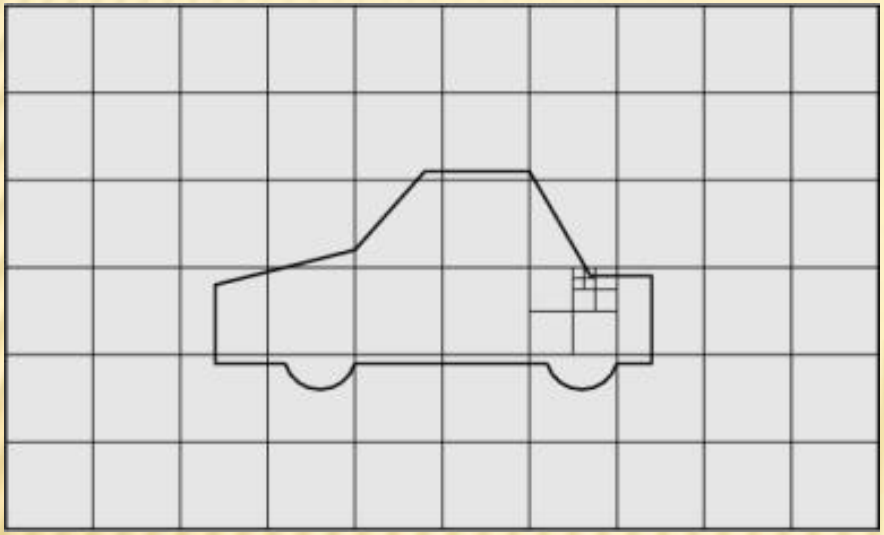


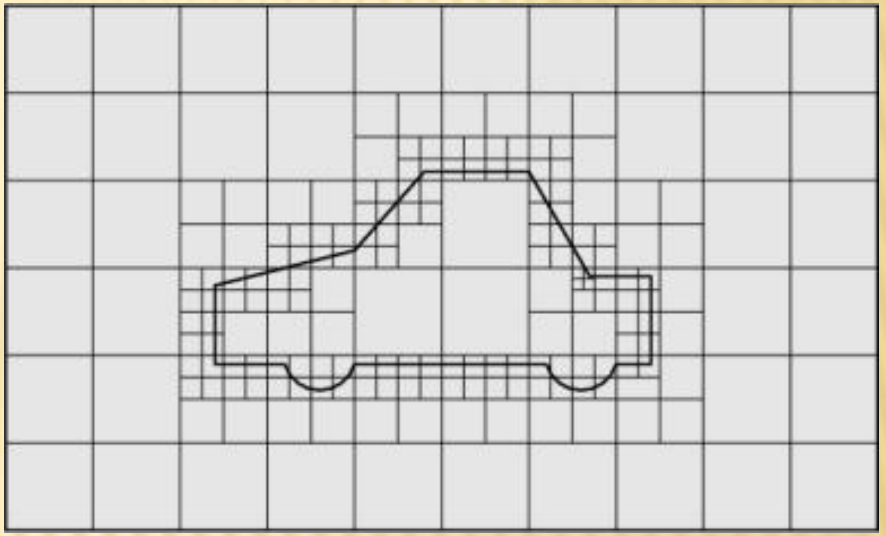
Characteristics of the grid:

- ✓ *The aspect ratio of the grid cells should be around 1.*
- ✓ *More than one cell in the z direction.*
- ✓ *At least on cell's edge should intersect the surface.*
- ✓ *There can not be empty patches, it is a 3D mesher.*

SnappyHexMesh: step 1

- **Starting the splitting process.**





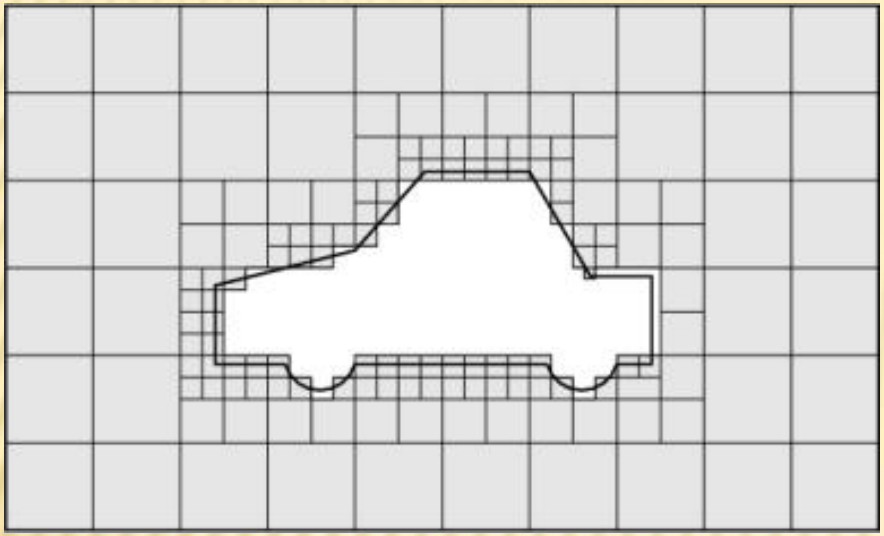
• Start the splitting from **locationInMesh** feature.
•This edge must be **inside the region to be meshed** and **must no coincide with a cell face.**

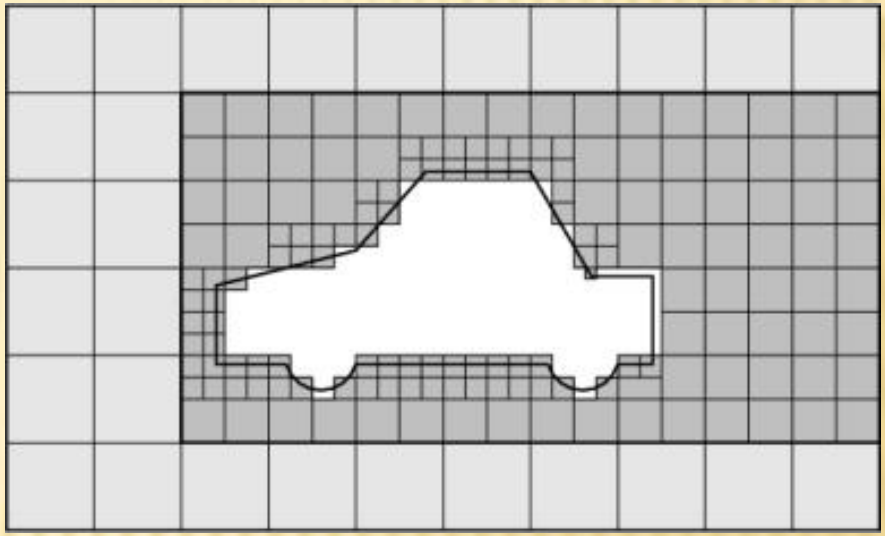
• Splitting the cells around the surface according to :

```
refinementSurfaces
 {
    file.stl
    {
        level (2 2); // default (min max) refinement for  surface
     }
 }
```

Snappyhexmesh: step 1

- **Cell removal process, and refinement of specified regions**





• Keep the side of the mesh defined by **locationInMesh.**
• Remove all cells that have above 50% of their volumes in the meshed region.

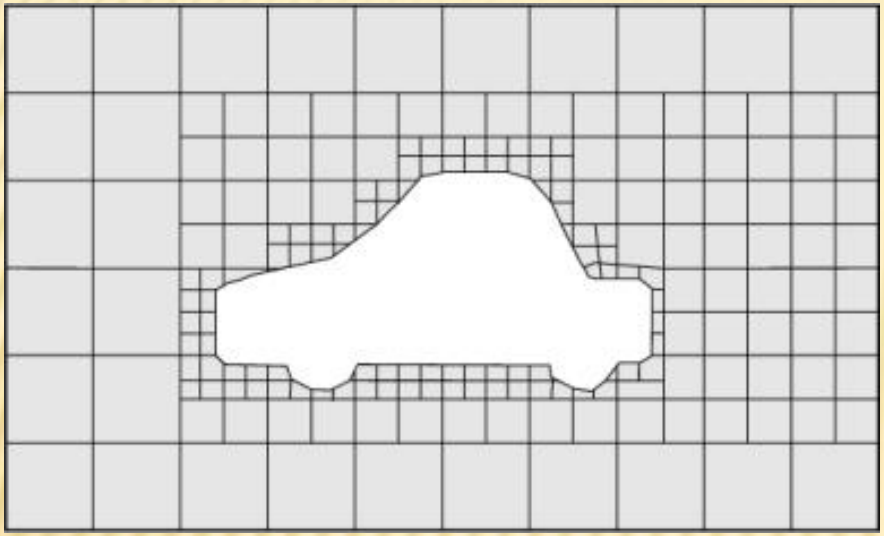
The region refined is specified by:
- **inside**: inside the volume region.
- **outside**: outside the volume region.
- **distance**: according to distance to the surface.

The region is defined first as geometry.

This first step is saved into the time folder 1.

Snappyhexmesh: step 2

- **Snapping to surface.**



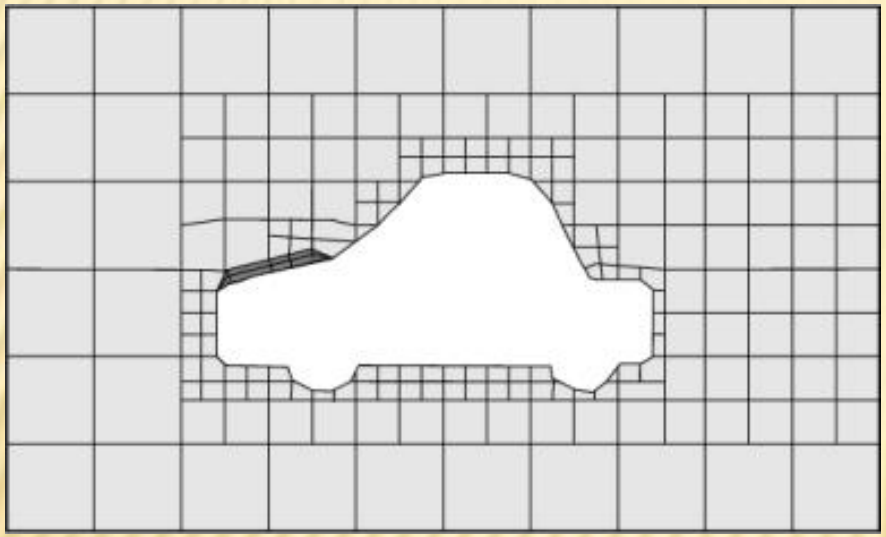
The steps to snap to surface are:

1. **Snap** the vertices onto the STL surface.
2. **Solve** for relaxation of the internal mesh.
3. **Iterate** using the snapControls in SnappyHexMeshDict.

This second step is saved into the time folder 2.

Snappyhexmesh: step 3

- **Boundary layer addition.**



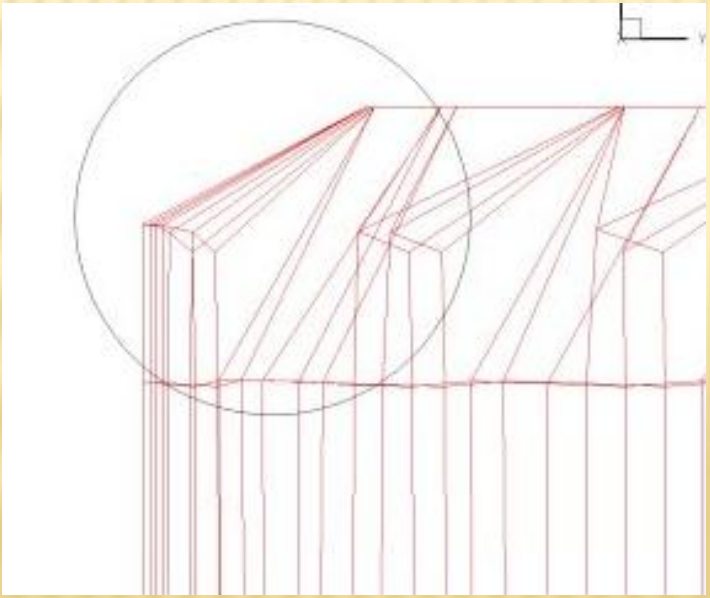
The boundary are applied on patches, not on surface!!

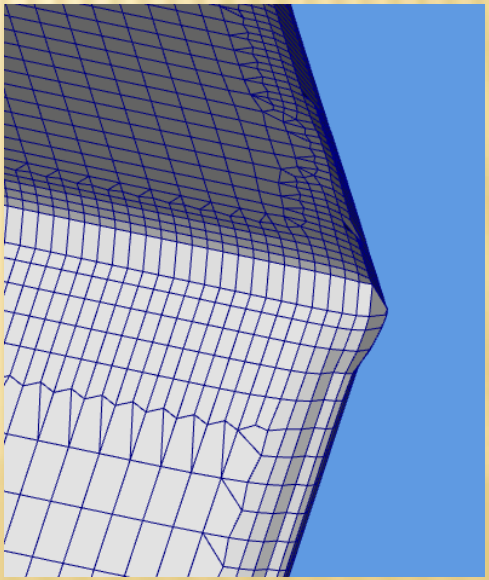
- Mesh projection back from the surface using a specified thickness normal to the surface.
- Solve for relaxation of the internal mesh with the latest projected boundary vertices.
- Check if validation criteria are validated.
- If the validation criteria can be satisfied, insert mesh layers.

This last step is saved into the time folder 3.

Snappyhexmesh: pros and cons.

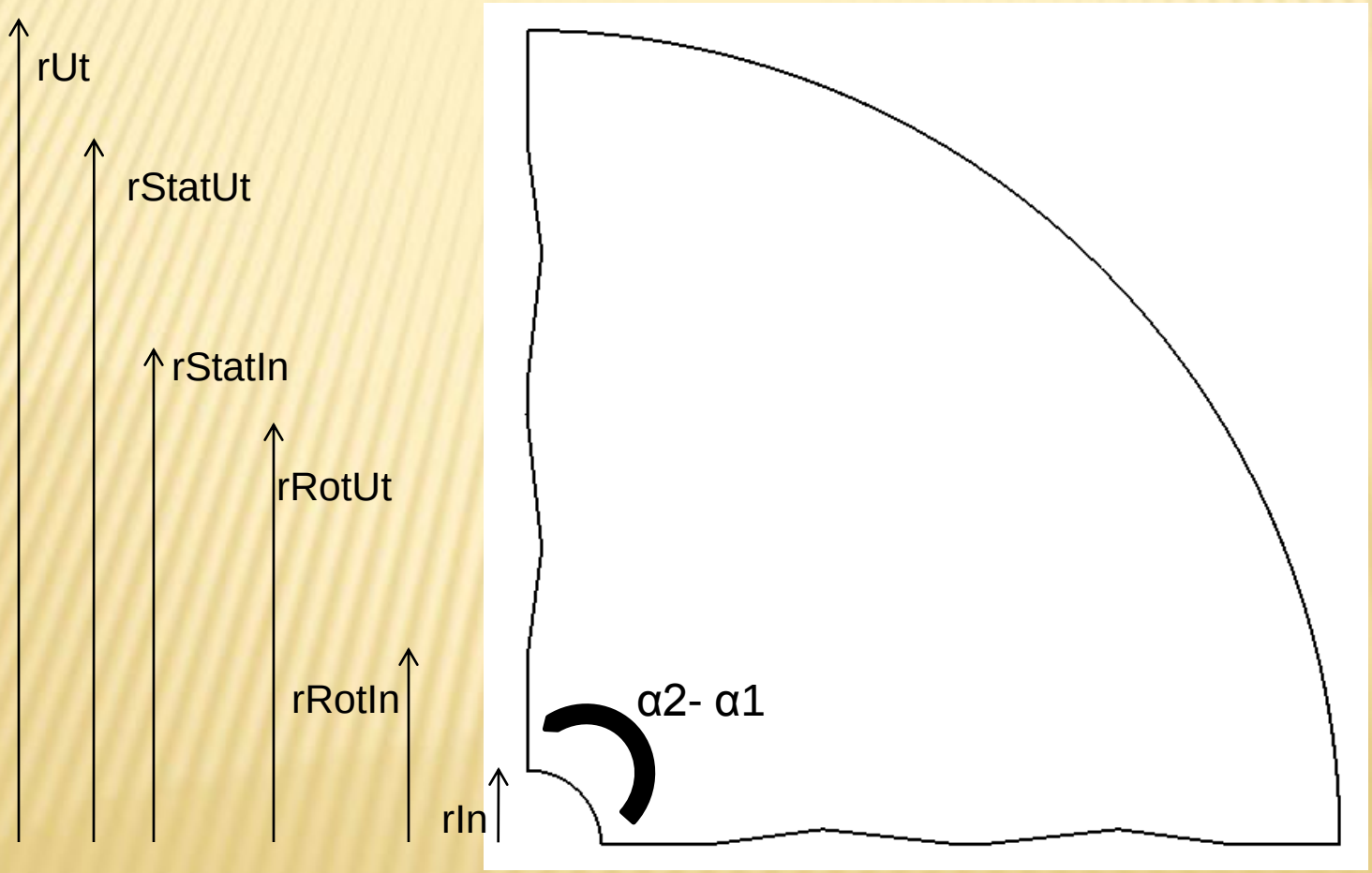
- Possibilities of multiple refinements that make it very robust.
- It runs in parallel.
- Need of a good quality STL surface, with more than one region/patch.
- The lack of geometry feature line makes snappyHexMesh not reliable on sharp edges.
- Icon developped an edge feature for OpenFOAM, but so far you have to pay to get it.





blockMesh/m4: a short tutorial.

- m4: allow a parametrization of the case, easy to change.



blockMesh/m4: pros and cons.

- Very easy way to create an simple geometry.
- The parametrization with m4 allow to create different geometry from the same m4 file.
- Not enough precision in the meshing parameters.
- Easy to go wrong on the orientation of the faces and blocks.

Import the mesh: pros and cons

- Need of an other software to create the mesh.
- Few converters:
  - **fluentMeshToFoam**, **fluent3DMeshToFoam** for Gambit mesh types.
  - **starToFoam** for STAR-CD mesh types.
  - **ideasToFoam** for I-DEAS mesh types
  - **cfx4ToFoam** for CFX mesh types.
  - **CGNSToFoam** for CGNS files (can import more than meshes), developped by users.

Conclusion

- SnappyHexMesh is an easy tool to generate a mesh, but is still young and lacks some important features.
- m4/blockMesh is a perfect tool when it comes to simple geometries, but is not enough developped to deal with real type geometries.
- The most common solution is to import a mesh from an external software.