Mesh Generation in OpenFOAM: snappyHexMesh

Crash Course

Hrvoje Jasak

`h.jasak@wikki.co.uk`

Wikki Ltd, United Kingdom

Introduction

`snappyHexMesh`: Automatic Complex Geometry Mesher in OpenFOAM

- `snappyHexMesh` utility is available as a part of OpenFOAM
- High quality hex and split-hex meshes generated from STL files
- Mesh generation fully automatic with many parameters allowing for the mesh quality control, local refinement and layered meshes
- `snappyHexMesh` is suitable for mesh generation in both internal and external flow simulations. For conjugate simulations, run `snappyHexMesh` twice!
- The mesh generation algorithm proceeds in stages
  1. Creation of background (initial mesh)
  2. Cell splitting at edges and surfaces, cell removal, and local refinement
  3. Snapping to surfaces and layer creation
- Mesh generation algorithm is capable of handling complex surfaces
- Local feature detection (edges and surfaces) control is available: good representation of the original geometry
- Easy to control the transition from coarse to fine mesh zones: minimizing the cell count
- Algorithm runs in parallel: creation of large meshes

Mesh Generation Process

Stages of Mesh Generation Process

- Creation of background (initial mesh)
- Cell splitting at edges and surfaces, cell removal, and local refinement
- Snapping to surfaces and layer creation

In order to generate mesh, a number of files that control various aspects of the mesh generation process are required

- Directory structure of the example directory

  `example/0  example/constant / example/system`
- Directory structure of constant subdirectory

  `example/constant/polyMesh example/constant/triSurface`
- `polyMesh` directory contains `blockMeshDict` needed for initial mesh creation
- `triSurface` directory contains feature description and/or STL file
- System directory contains `controlDict`, `snappyHexMeshDict`, `fvSolution` and `fvSchemes` files
- The overall structure is very similar to a typical OpenFOAM case directory structure

Mesh Generation Process

Stages of mesh generation process

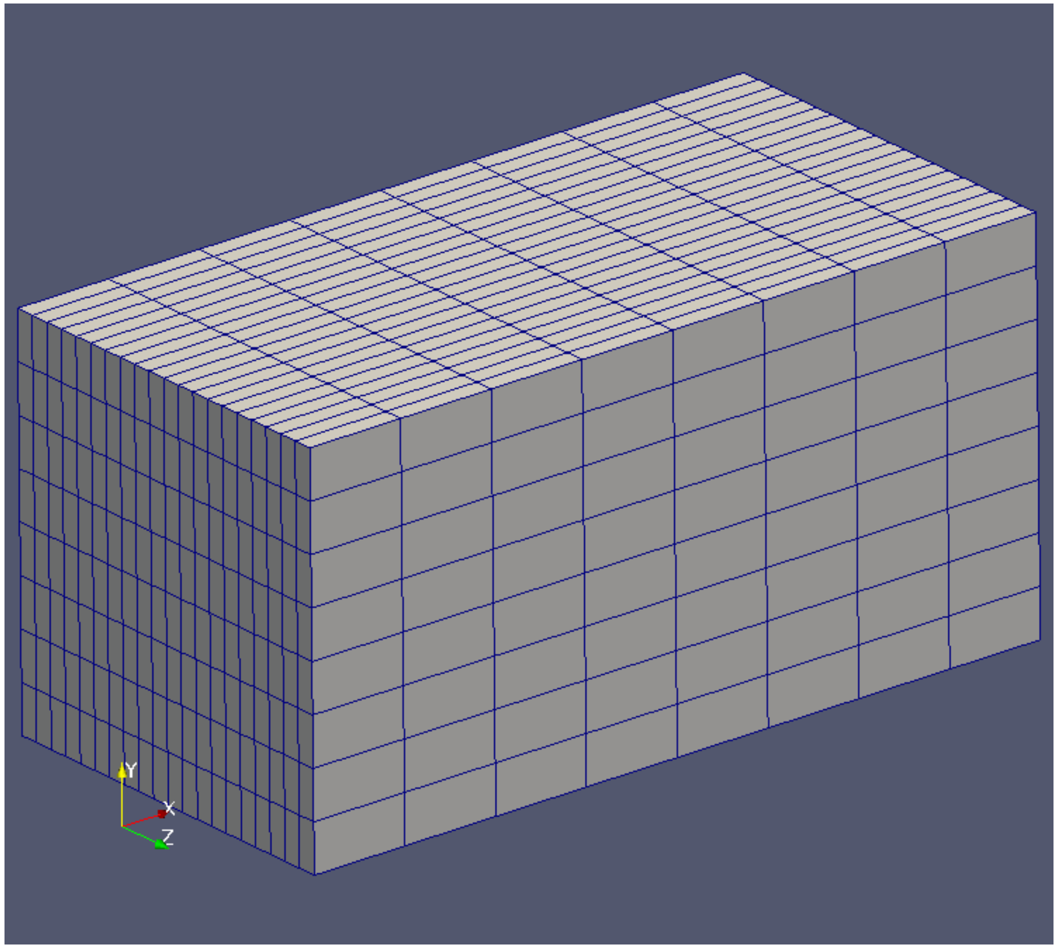
- `blockMeshDict` file is responsible for specifying the initial mesh parameters

```
convertToMeters 1;
vertices
(
    (-5 -2.5 0) (5 -2.5 0) (5  2.5 0) (-5  2.5 0)
    (-5 -2.5 5) (5 -2.5 5) (5  2.5 5) (-5  2.5 5)
);
blocks
(
    hex (0 1 2 3 4 5 6 7) (8 8 20) simpleGrading (1 1 1)
);
edges
();
patches
(
    patch in ( (0 1 2 3) )   patch out ( (7 6 5 4) )
    wall right ( (2 6 5 1) ) wall left ( (3 7 4 0) )
    wall up ( (2 6 7 3) )    wall down ( (4 5 1 0) )
);
mergePatchPairs
();
```

Mesh Generation Process

Initial Background Mesh

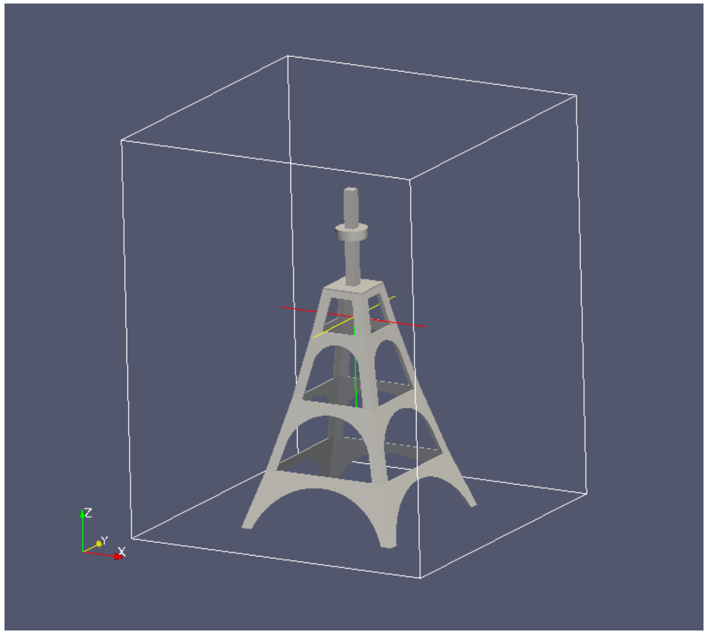
- `blockMesh` utility produces the following output



Mesh Generation Process

Initial Background Mesh

- Initial background mesh does not need to be very fine
- There should be at least one cell edge that intersects with a surface that needs to meshed
- Therefore, one cell background mesh will not work
- snappyHexMesh utility uses OpenFOAM search and intersection algorithms to create a mesh
- Background mesh must encompass the geometry fully



Mesh Generation Process

`snappyHexMeshDict` Structure

```
geometry
{
    ...
}
castellatedMeshControls
{
    ...
}
snapControls
{
    ...
}
addLayersControls
{
    ...
}
meshQualityControls
{
    ...
}
```

Mesh Generation Process

`snappyHexMeshDict` Structure

- Global parameters

```
castellatedMesh true;
snap            true;
addLayers       true;
debug 0;
mergeTolerance 1e-6;
```

Mesh Generation Process

Geometry Section

- Describes various surfaces and regions within the mesh
- Can describe geometry using an STL file or through set of tessellated primitives
- It serves as an input for other stages of mesh generation
- It is up to user to decide how surfaces and regions are used
- the type of the surface helps to select the appropriate algorithm for detection of surfaces and refinement regions: searchableCylinder, triSurfaceMesh, triSurfaceMeshWithGaps.
- Entities defined in STL file can be named for meshing purposes using regions keyword

Mesh Generation Process

WIKKI

`snappyHexMeshDict` Structure

- Geometry section

```
geometry
{
    pipe
    {
        type searchableCylinder;
        point1 (0.5 0.5 -1);
        point2 (0.5 0.5 6);
        radius 0.5;
    }

    innerPipe
    {
        type searchableCylinder;
        point1 (0.5 0.5 0);
        point2 (0.5 0.5 5);
        radius 0.1;
}
```

Mesh Generation Process

`snappyHexMeshDict` Structure

- Geometry section

```
geometry
{
    boat.stl
    {
        type triSurfaceMesh;
        name boat;
    }

    refinementBox
    {
        type searchableBox;
        min (-8000 -4000 -3000);
        max ( 8000  4000 3000);
    }
}
```

Mesh Generation Process

`snappyHexMeshDict` Structure

- Geometry section

```
geometry
{
    car.stl
    {
        type triSurfaceMeshWithGaps;
        gap  0.005;
        regions
        {
            chasis
            {
                name chasis;
            }
        }
    }
}
```

Mesh Generation Process

`castellatedMeshControls`

- This section specifies various parameters that control surface intersection and region refinement algorithms
- Geometrical entities specified in the geometry section are available here to be used for surface intersection and refinement algorithms
- Each geometric entity can be associated with a particular keyword thus indicating whether the entity is to be used for geometrical or refinement considerations
- In addition, load balancing and the stopping criteria are defined in this section

Mesh Generation Process

`snappyHexMeshDict` Structure

- `castellatedMeshControls`

```
castellatedMeshControls
{
    maxLocalCells 1000000;
    maxGlobalCells 2000000;
    minRefinementCells 0;
    maxLoadUnbalance 0.10;
    nCellsBetweenLevels 1;
    features ();
    refinementSurfaces
    {
        pipe{ level (2 1);}
    }
    refinementRegions
    {
        innerPipe { mode inside; levels ((1 2));}
    }
    resolveFeatureAngle 20;
    locationInMesh (0.54 0.52 0.16);
}
```

Mesh Generation Process

`castellatedMeshControls` Parameters

- `maxLocalCells`: If local number of cells is >= maxLocalCells on any processor switches from from refinement followed by balancing (current method) to (weighted) balancing before refinement
- `maxGlobalCells`: Overall cell limit (approximately). Refinement will stop immediately upon reaching this number so a refinement level might not complete. Note that this is the number of cells before removing the part which is not 'visible' from the keepPoint. The final number of cells might actually be a lot less
- `minRefinementCells`: The surface refinement loop might spend lots of iterations refining just a few cells. This setting will cause refinement to stop if <= minimumRefine are selected for refinement. Note: it will at least do one iteration (unless the number of cells to refine is 0)
- `maxLoadUnbalance`: Allow a certain level of imbalance during refining (since balancing is quite expensive) Expressed as fraction of perfect balance (= overall number of cells / nProcs). 0=balance always.
- `nCellsBetweenLevels`: Number of buffer layers between different levels. 1 means normal 2:1 refinement restriction, larger means slower refinement.

Mesh Generation Process

`castellatedMeshControls` Parameters

- `features`: Specifies a level for any cell intersected by its edges. This is a `featureEdgeMesh`, read from constant/triSurface

```
features
(
    {
        file "someLine.eMesh";
        level 2;
    }
);
```

- `refinementSurfaces`: Specifies two levels for every surface. The first is the minimum level, every cell intersecting a surface gets refined up to the minimum level. The second level is the maximum level. Cells that 'see' multiple intersections where the intersections make an angle > resolveFeatureAngle get refined up to the maximum level
- resolveFeatureAngle: Resolve sharp angles

Mesh Generation Process

`castellatedMeshControls` Parameters

- `refinementRegions`: Specifies refinement level for cells in relation to a surface. One of three modes
  - `distance`: ’levels’ specifies per distance to the surface the wanted refinement level. The distances need to be specified in descending order.
  - `inside`: ’levels’ is only one entry and only the level is used. All cells inside the surface get refined up to the level. The surface needs to be closed for this to be possible.
  - outside. Same but cells outside.
- `locationInMesh`: After refinement patches get added for all refinementSurfaces and all cells intersecting the surfaces get put into these patches. The section reachable from the locationInMesh is kept. NOTE: This point should never be on a face, always inside a cell, even after refinement.

Mesh Generation Process

`snappyHexMeshDict` Structure

- `snapControls`

```
snapControls
{
    nSmoothPatch 6;

    tolerance 10.0;

    nSolveIter 60;

    nRelaxIter 10;
}
```

Mesh Generation Process

`snapControls` Parameters

- `nSmoothPatch`: Number of patch smoothing iterations before finding correspondence to surface
- `tolerance`: Relative distance for points to be attracted by surface feature point or edge. True distance is this factor times local maximum edge length
- `nSolveIter`: Number of mesh displacement relaxation iterations
- `nRelaxIter`: Maximum number of snapping relaxation iterations. Should stop before upon reaching a correct mesh

Mesh Generation Process

`snappyHexMeshDict` Structure

- `addLayersControls`

```
addLayersControls
{
    relativeSizes true;
    layers{ pipe_region0 { nSurfaceLayers 3;} }
    expansionRatio 2.0;
    finalLayerThickness 0.5;
    minThickness 0.05;
    nGrow 1;
    featureAngle 60;
    nRelaxIter 5;
    nSmoothSurfaceNormals 1;
    nSmoothNormals 3;
    nSmoothThickness 10;
    maxFaceThicknessRatio 0.5;
    maxThicknessToMedialRatio 0.3;
    minMedianAxisAngle 130;
    nBufferCellsNoExtrude 0;
    nLayerIter 50;
}
```

Mesh Generation Process

`addLayersControls` Parameters

- relativeSizes: Are the thickness parameters below relative to the undistorted size of the refined cell outside layer (true) or absolute sizes (false)
- layers: Per final patch (so not geometry!) the layer information

```
layers
 {
     minZ
     {
         nSurfaceLayers 1;
     }
     motorBike_frt-fairing:001%1
     {
         nSurfaceLayers 1;
     }
 }
```

- expansionRatio: Expansion factor for layer mesh
- finalLayerThickness: Wanted thickness of final added cell layer. If multiple layers is the thickness of the layer furthest away from the wall. See relativeSizes parameter

Mesh Generation Process

`addLayersControls` Parameters

- minThickness: Minimum thickness of cell layer. If for any reason layer cannot be above minThickness do not add layer. Relative to undistorted size of cell outside layer
- nGrow: If points get not extruded do nGrow layers of connected faces that are also not grown. This helps convergence of the layer addition process close to features
- featureAngle: When not to extrude surface. 0 is flat surface, 90 is when two faces make straight angle
- nRelaxIter: Maximum number of snapping relaxation iterations. Should stop before upon reaching a correct mesh
- nSmoothSurfaceNormals: Number of smoothing iterations of interior mesh movement direction

Mesh Generation Process

`addLayersControls` Parameters

- nSmoothThickness: Smooth layer thickness over surface patches
- maxFaceThicknessRatio: Stop layer growth on highly warped cells
- maxThicknessToMedialRatio: Reduce layer growth where ratio thickness to medial distance is large
- minMedianAxisAngle: Angle used to pick up medial axis points
- nBufferCellsNoExtrude: Create buffer region for new layer terminations
- nLayerIter: Overall max number of layer addition iterations

Mesh Generation Process

`snappyHexMeshDict` Structure

- `meshQualityControls`

```
meshQualityControls
{
    maxNonOrtho 65;
    maxBoundarySkewness 20;
    maxInternalSkewness 4;
    maxConcave 80;
    minFlatness 0.5;
    minVol -1e-13;
    minArea -1;
    minTwist 0.05;
    minDeterminant 0.001;
    minFaceWeight 0.05;
    minVolRatio 0.01;
    minTriangleTwist -1;
    nSmoothScale 4;
    errorReduction 0.75;
}
```

Mesh Generation Process

`meshQualityControls` Parameters

- maxNonOrtho: Maximum non-orthogonality allowed. Set to 180 to disable
- maxBoundarySkewness: Max skewness allowed. Set to <0 to disable
- maxInternalSkewness: Max skewness allowed. Set to <0 to disable
- maxConcave: Max concaveness allowed. Is angle (in degrees) below which concavity is allowed. 0 is straight face, <0 would be convex face. Set to 180 to disable
- minFlatness: Minimum projected area v.s. actual area. Set to -1 to disable
- minVol: Minimum pyramid volume. Is absolute volume of cell pyramid. Set to a sensible fraction of the smallest cell volume expected. Set to very negative number (e.g. -1E30) to disable
- minArea: Minimum face area. Set to <0 to disable

Mesh Generation Process

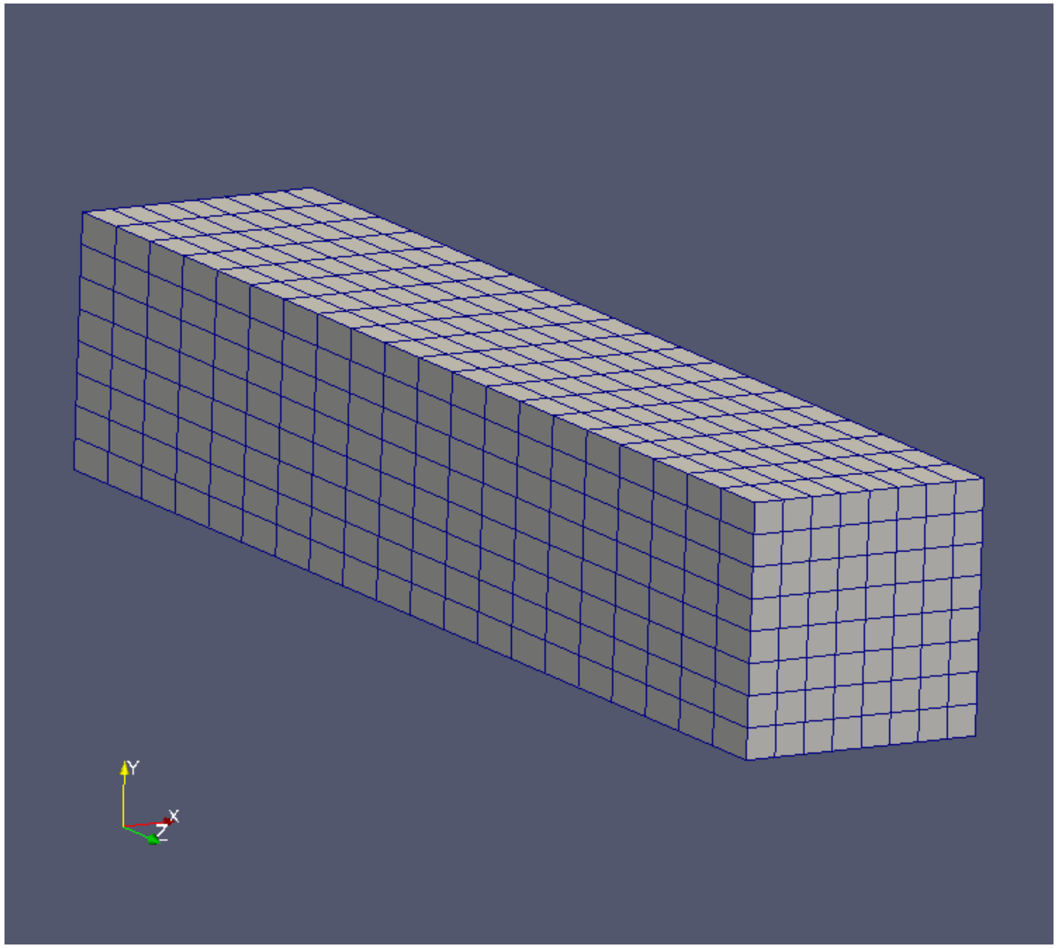
`meshQualityControls` Parameters

- minTwist: Minimum face twist. Set to <-1 to disable. dot product of face normal and face center triangles normal
- minDeterminant: minimum normalized cell determinant 1 = hex, <= 0 = folded or flattened illegal cell
- minFaceWeight: minFaceWeight (0 -> 0.5)
- minVolRatio: must be >0 for Fluent compatibility
- nSmoothScale: Number of error distribution iterations
- errorReduction: Amount to scale back displacement at error points

Mesh generation in OpenFOAM

Stages of Mesh Generation

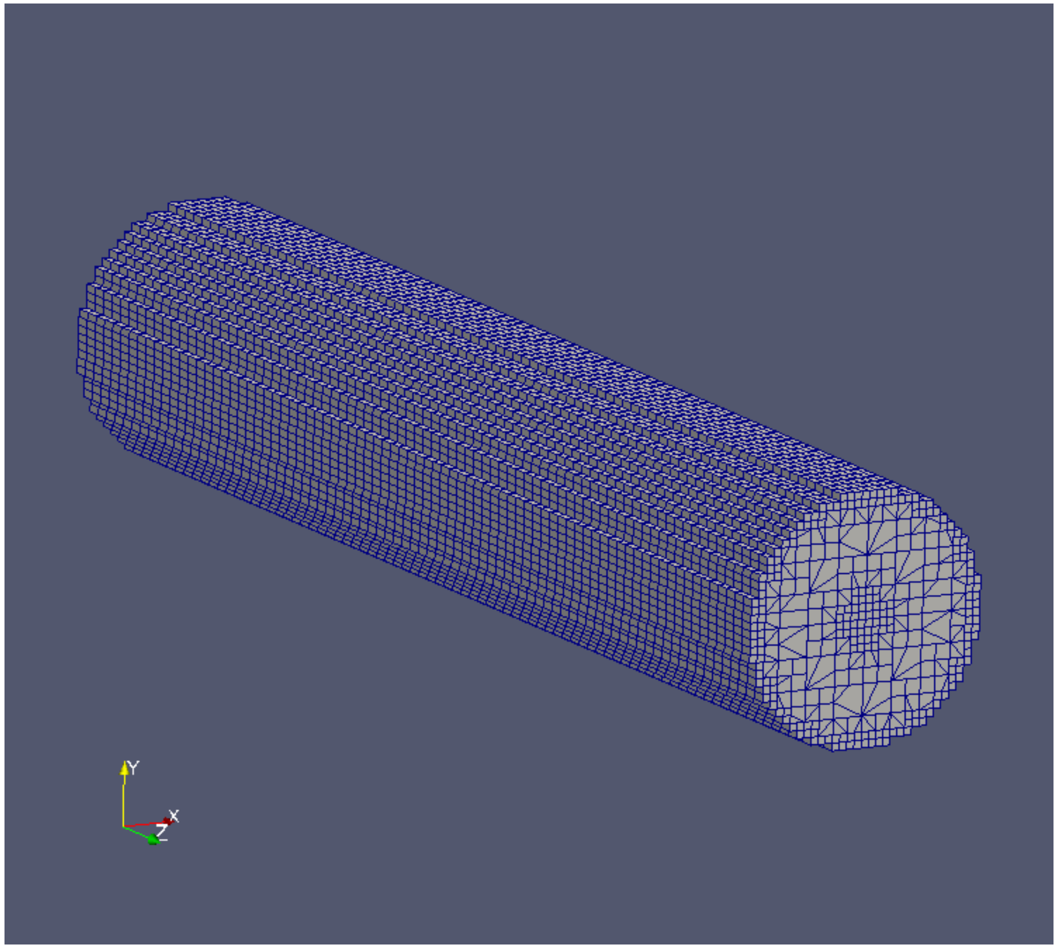
- Background mesh



Mesh generation in OpenFOAM

Stages of Mesh Generation

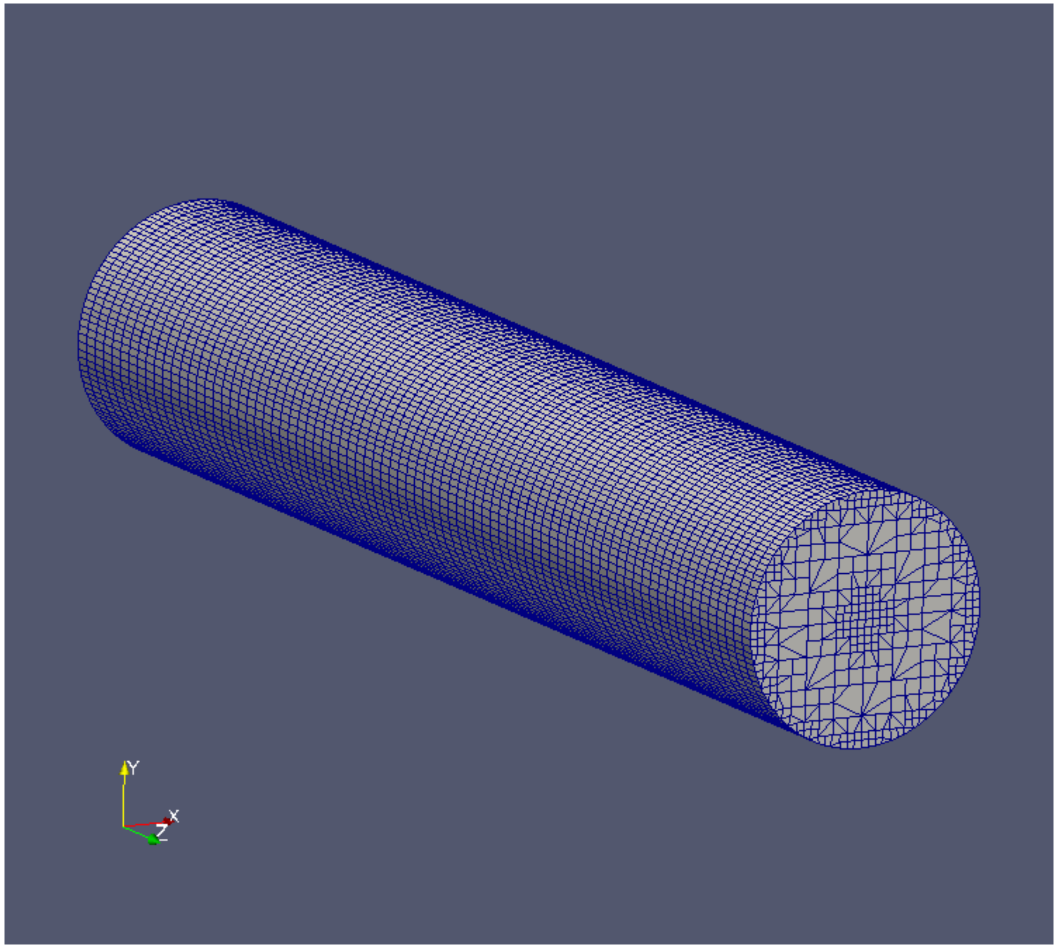
- Surface intersection



Mesh generation in OpenFOAM

Stages of Mesh Generation

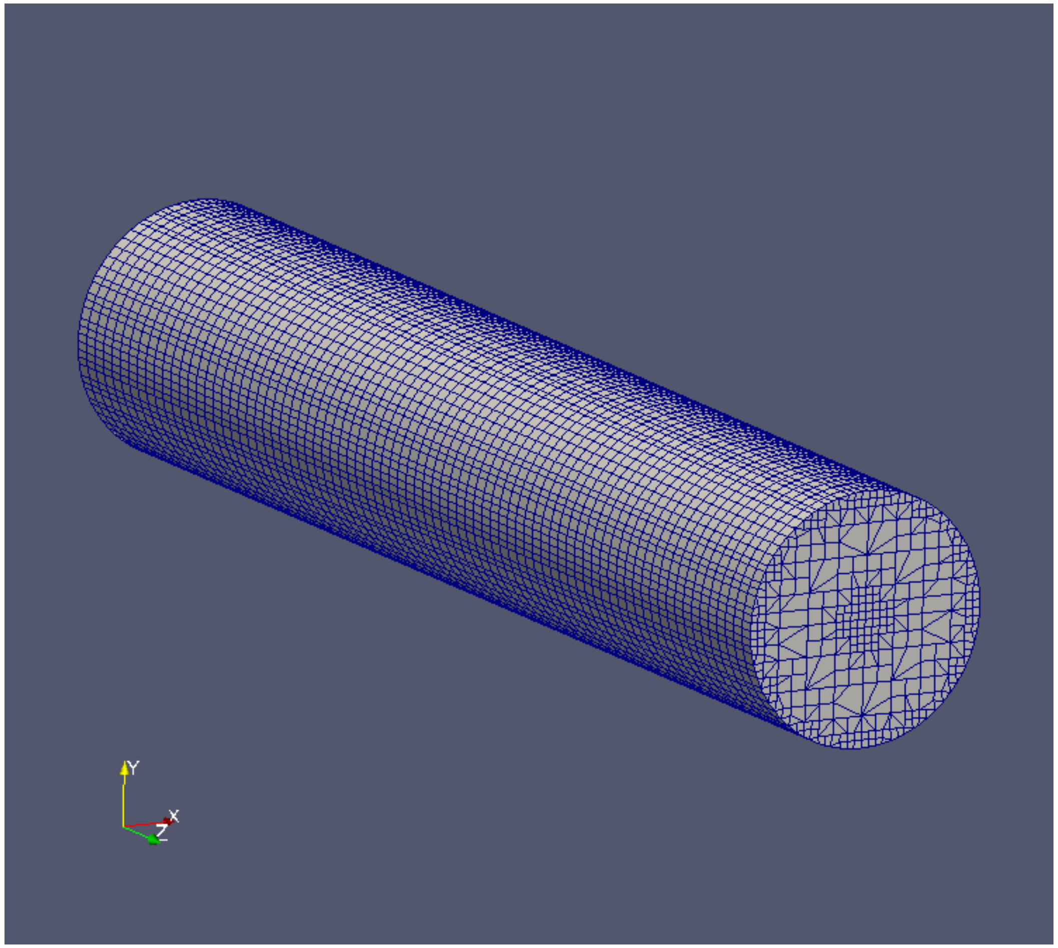
- Mesh snapping



Mesh generation in OpenFOAM

Mesh generation stages

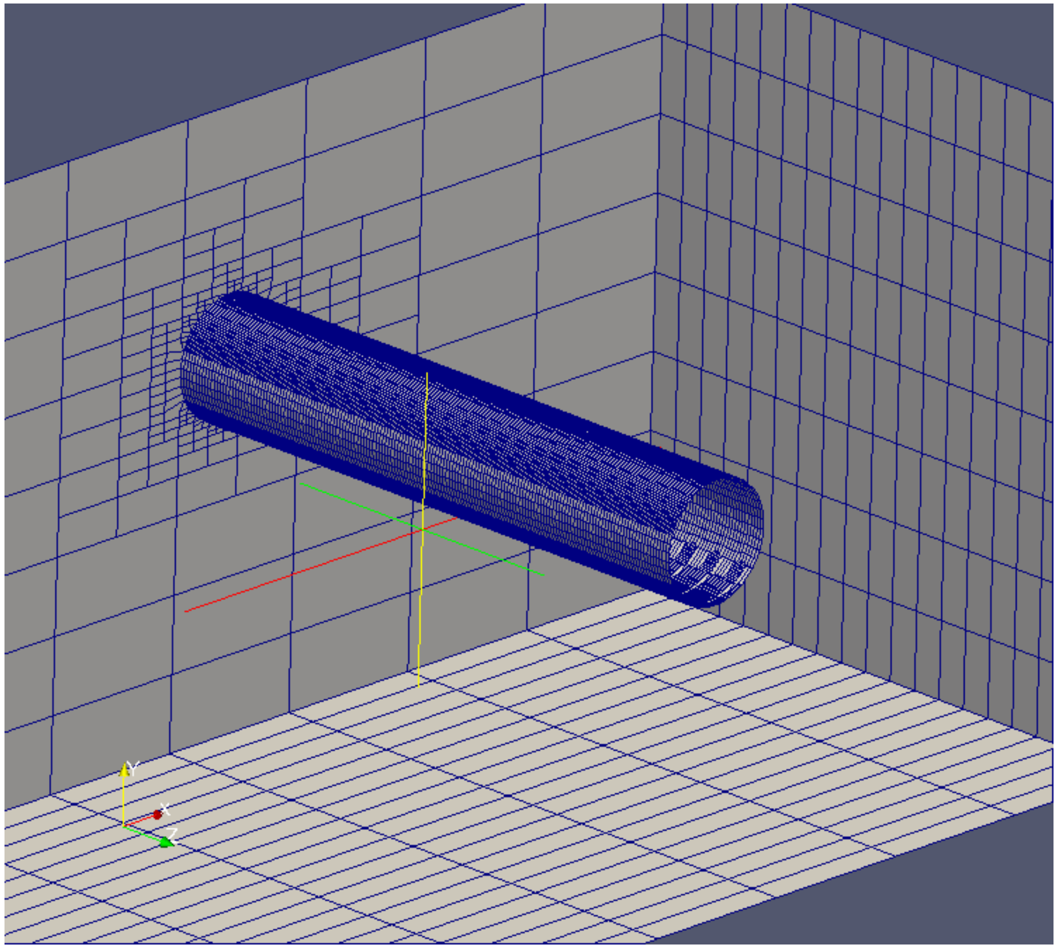
- Refinement, smoothing and layering



Mesh generation in OpenFOAM

Cylinder in Cross-Flow: Coarse Mesh

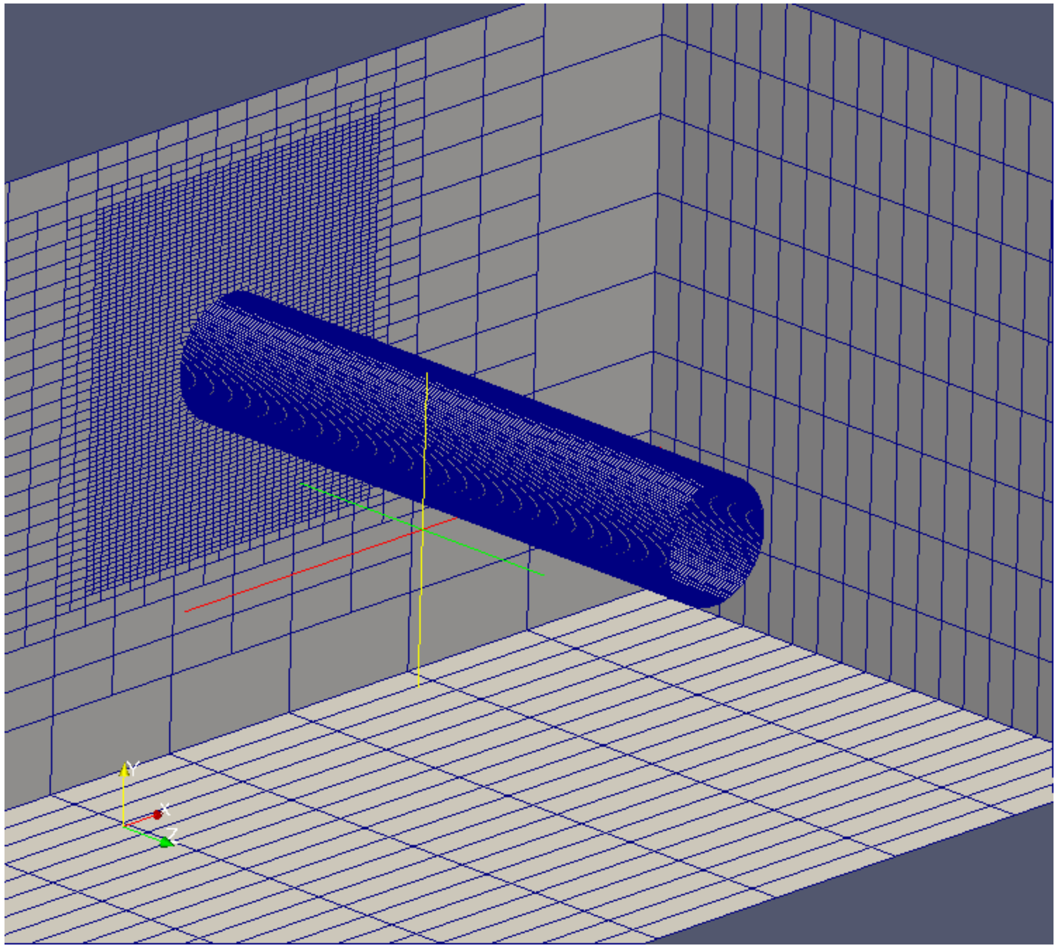
- Coarse mesh without local refinement



Mesh generation in OpenFOAM

Cylinder in cross-flow: fine mesh

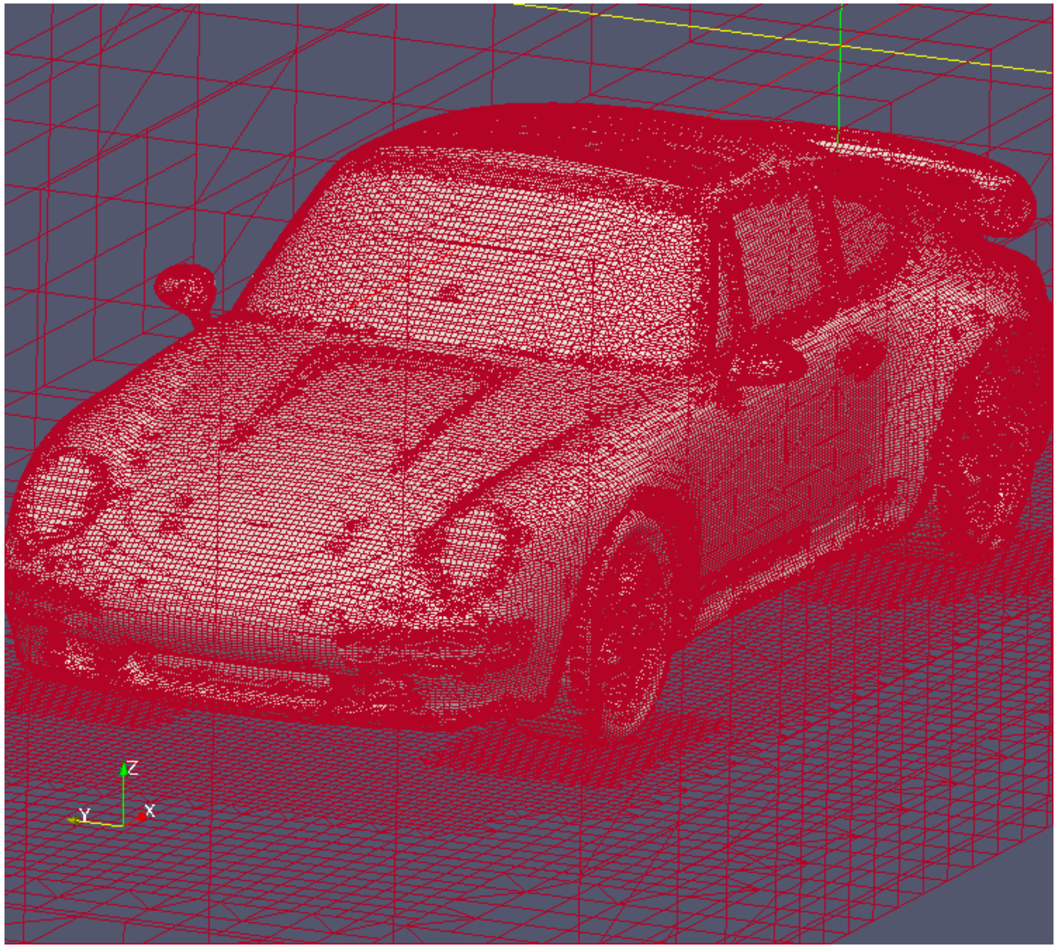
- Coarse mesh with local refinement



Mesh generation in OpenFOAM

An example of more realistic geometry

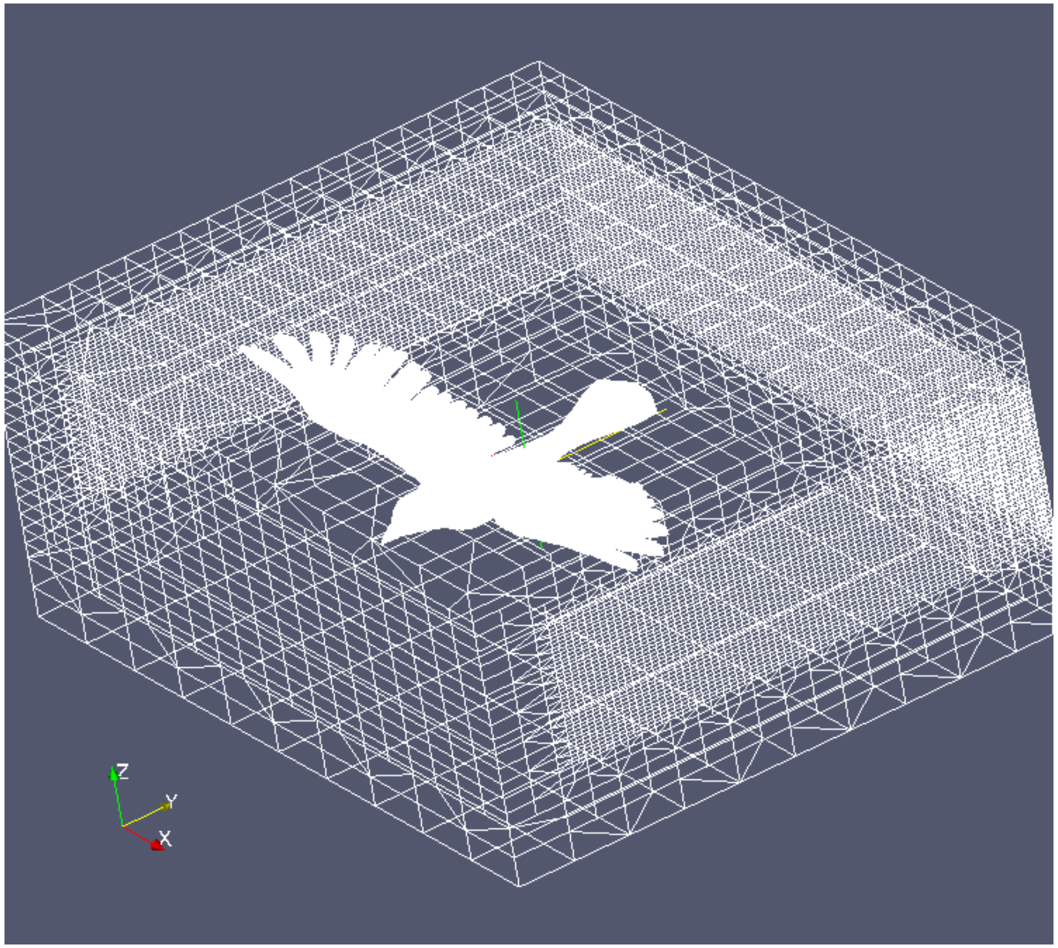
- Mesh for external simulation of the flow around a car



Mesh generation in OpenFOAM

An example of a very complex STL surface

- Mesh of a dove with many fine features



Mesh generation in OpenFOAM

An example of a very complex STL surface

- Wing detail