Meshing

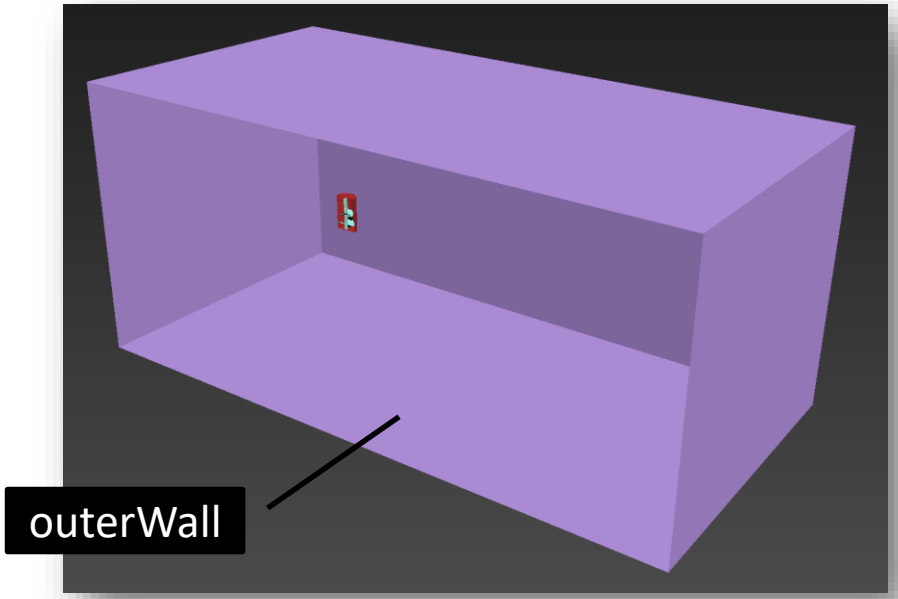
Advanced course

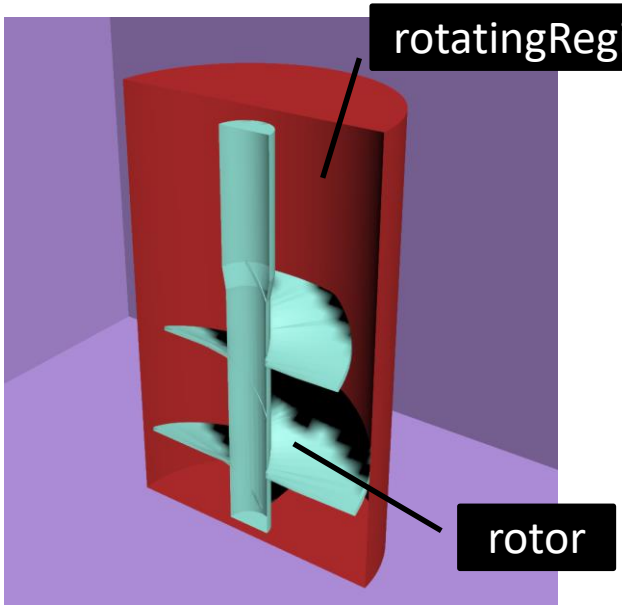
Content

- 3D environment
- Export to STL
- blockMesh
- snappyHexMesh
  - Castellated mesh
  - Snap
  - addLayers
- Multiple reference frame - MRF
- Arbitrary mesh interface - AMI

3D environment

- 3D geometry must be closed (no leak holes)
- Model both real geometry (rotor, domain wall, obstacles, ...) and virtual geometry (rotating region, porous media, refinement regions, ...)



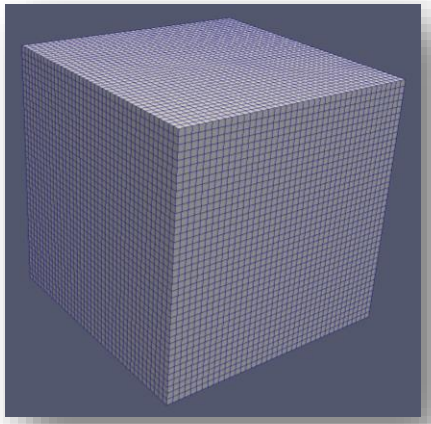


Export to STL

- One by one export: each part into a separate STL file
- Use a high resolution for the export:
  - STL file facets are not the same as the final mesh elements: 2 different things!
  - STL file functions as the snap-to surface for the mesh → mesh is at least as rough as the stl file
  - High-res mesh can be built on a low-res stl and vice versa
  - High-res stl does not necessarily increase solver time
- Use ASCII format if possible
- Place STL files under \constant\triSurface

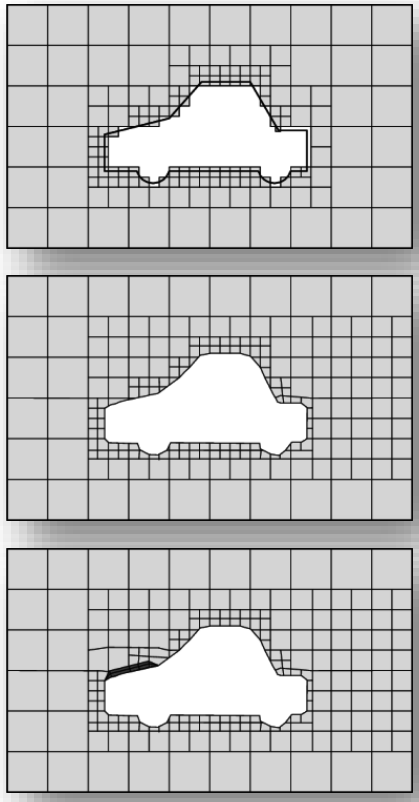
blockMesh

- Location: \constant\polyMesh\blockMeshDict
- Command: blockMesh
- This is the general grid of the domain
- Refinements will be made relative to this grid
- blockMesh must be at least as large as the domain
- Objects must intersect with the blockMesh to be spotted by snappyHexMesh
- Patches can be defined
- Different zones possible
- Orthogonal mesh
- Reference files: Z-axis is vertical



snappyHexMesh

- Location: \system\snappyHexMeshDict
- Command: snappyHexMesh (-overwrite)
- Procedure:
  - Step 0 - optional: extract features (edges) from stl files as input for refinement
  - Step 1: create the castellated mesh
    - local refinement of the blockMesh blocks into smaller blocks
    - Refinement based on features, surfaces or regions
  - Step 2: snap the mesh to surfaces/edges
  - Step 3 – optional: add layers



snappyHexMeshDict

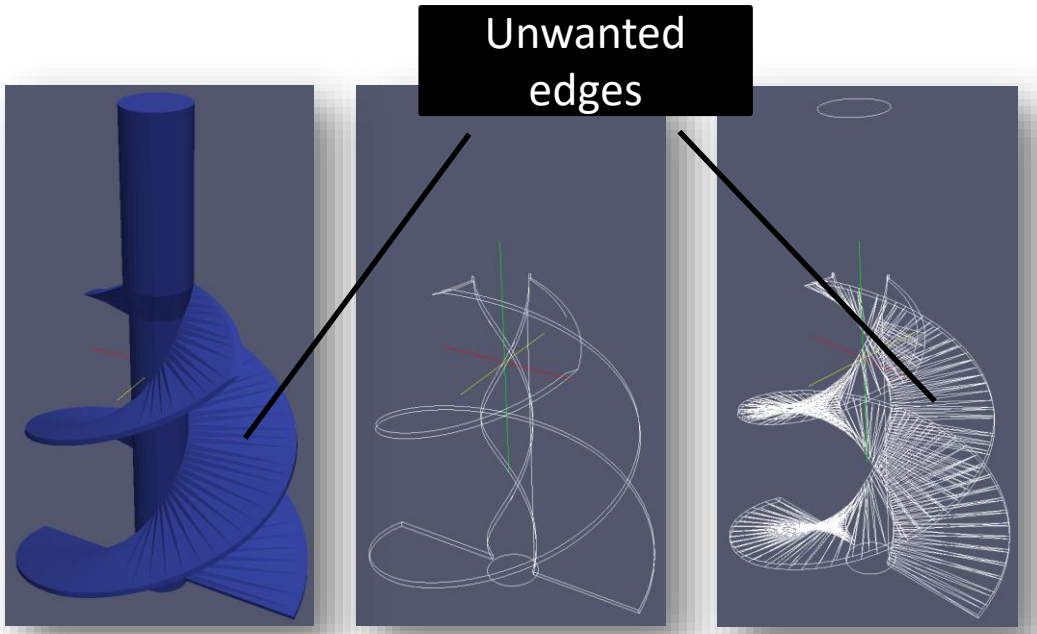
- Location: \system\surfaceFeatureExtractDict
- Command: surfaceFeatureExtract (-overwrite)
- Include angle: typically 150

```
outerWall.stl
{
    #include "surfaceFeatureExtractDictDefaults"
}


rotor.stl
{
    #include "surfaceFeatureExtractDictDefaults"
}


rotatingRegion.stl
{
    #include "surfaceFeatureExtractDictDefaults"
}
```

```
// How to obtain raw features (extractFromFile || extractFromSurface)
extractionMethod    extractFromSurface;

extractFromSurfaceCoeffs
{
    // Mark edges whose adjacent surface normals are at an angle less
    // than includedAngle as features
    // - 0  : selects no edges
    // - 180: selects all edges
    includedAngle   100;
}

trimFeatures
{
    // Remove features with fewer than the specified number of edges
    minElem         10;
}

writeObj                yes;
```

snappyHexMeshDict

- Location eMesh files: \constant\triSurface
- Location obj files: extendedFeatureEdgeMesh
- Example: ***.edgeMesh.obj shows the extracted edges
- STL quality matters!



snappyHexMeshDict

- Determine which of the steps needs to be run

```
// Which of the steps to run
castellatedMesh true;
snap            true;
addLayers       false;
```

snappyHexMeshDict

- Define the geometry
  - Input: stl/obj files or manual definition (box, cylinder, sphere, ...)
  - Scale factor can be applied
  - Define a name

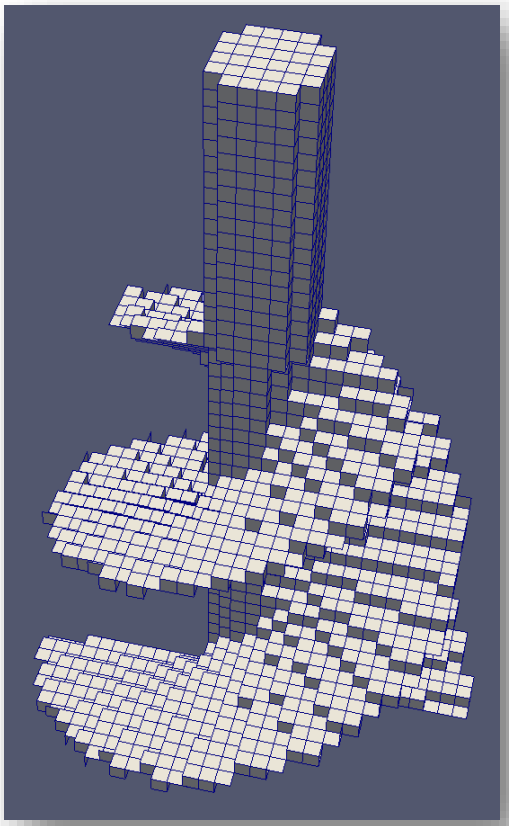
```
geometry
{
    rotor.stl
    {
        type        triSurfaceMesh;
        scale 0.001;
        name        rotor;
    }
    rotatingRegion.stl
    {
        type        triSurfaceMesh;
        scale 0.001;
        name        rotatingRegion;
    }
    outerWall.stl
    {
        type        triSurfaceMesh;
        scale 0.001;
        name        outerWall;

    }
}
```

```
outerWall
{
    type searchableCylinder;
    point1 (0 0 -0.8);
    point2 (0 0 0.5);
    radius 1.1;
}
```

castellatedMeshControls

- Refinement parameters

```
// Refinement parameters
// ~~~~~~~~~~~~~~~~~~~~

// If local number of cells is >= maxLocalCells on any processor
// switches from from refinement followed by balancing
// (current method) to (weighted) balancing before refinement.
maxLocalCells 100000;

// 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.
maxGlobalCells 2000000;

// 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)
minRefinementCells 0;

// 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.
maxLoadUnbalance 0.10;


// Number of buffer layers between different levels.
// 1 means normal 2:1 refinement restriction, larger means slower
// refinement.
nCellsBetweenLevels 1;
```



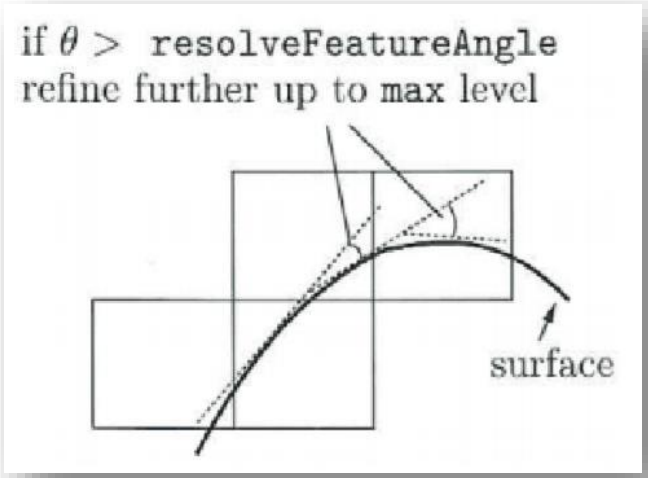
castellatedMeshControls

- Explicit feature edge refinement
  - Refers to features extracted from the stl files (during optional surfaceFeatureExtract step)

```
// Explicit feature edge refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies a level for any cell intersected by its edges.
// This is a featureEdgeMesh, read from constant/triSurface for now.
features
(
    {
        file        "rotatingRegion.eMesh";
        level       6;
    }
    {
        file        "outerWall.eMesh";
        level       1;
    }
    {
        file        "rotor.eMesh";
        level       6;
    }

);
```

castellatedMeshControls

- Surface based refinement
  - Refine cells inside a volume (min & max)
  - Create face- & cellZones for later use




```
// Surface based refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~

// 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.

refinementSurfaces
{
    rotatingRegion
    {
        level       (4 4);

        faceType    boundary;
        cellZone    rotatingRegion;
        faceZone    rotatingRegion;
        cellZoneInside  inside;
    }
    outerWall
    {
        level       (0 0);
    }
    rotor
    {
        level       (4 5);
    }
}

// Resolve sharp angles
resolveFeatureAngle 30;
```

castellatedMeshControls

- Region-wise refinement

```
// Region-wise refinement
// ~~~~~~~~~~~~~~~~~~~~~~

// 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.

refinementRegions
{

    rotatingRegion
    {
        mode        inside;
        levels      ((1E15 4));
    }
}
```

castellatedMeshControls

- Mesh selection → check every time!

```
// Mesh selection
// ~~~~~~~~~~~~~~

// 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.
locationInMesh (0 0 -2);
```

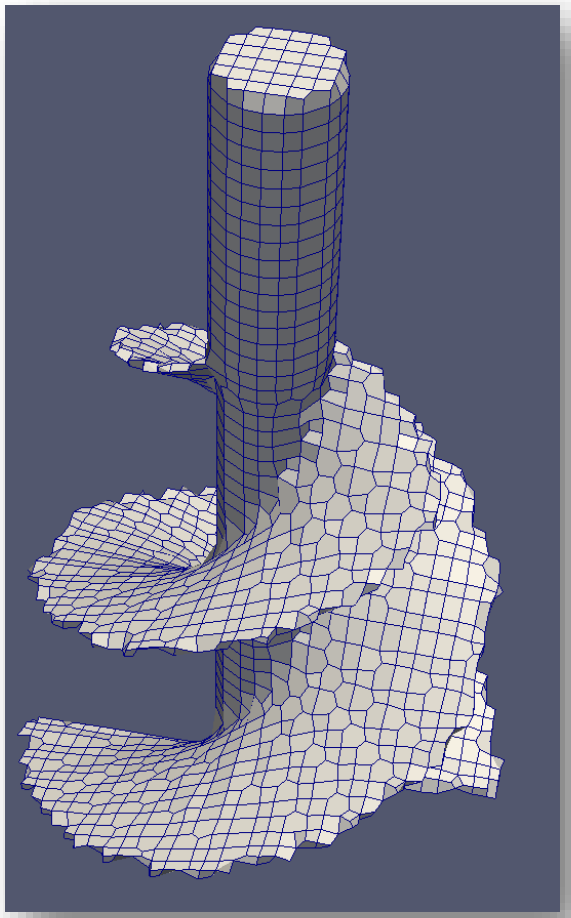
- faceZones

```
// Whether any faceZones (as specified in the refinementSurfaces)
// are only on the boundary of corresponding cellZones or also allow
// free-standing zone faces. Not used if there are no faceZones.
allowFreeStandingZoneFaces false;
```

snapControls

- General controls

```
//- Number of patch smoothing iterations before finding correspondence
//  to surface
nSmoothPatch 3;

//- Relative distance for points to be attracted by surface feature point
//  or edge. True distance is this factor times local
//  maximum edge length.
tolerance 4.0; // 1.0;

//- Number of mesh displacement relaxation iterations.
nSolveIter 300;

//- Maximum number of snapping relaxation iterations. Should stop
//  before upon reaching a correct mesh.
nRelaxIter 5;
```



http://cfd.direct/openfoam/user-guide/snappyhexmesh/

snapControls

- Feature snapping
  - Implicit: let snappy detect features
  - Explicit: use extracted features (more control)

```
// Number of feature edge snapping iterations.
// Leave out altogether to disable.
nFeatureSnapIter 10;

// Detect (geometric only) features by sampling the surface
// (default=false).
implicitFeatureSnap true;

// Use castellatedMeshControls::features (default = true)
explicitFeatureSnap false;

// Detect features between multiple surfaces
// (only for explicitFeatureSnap, default = false)
multiRegionFeatureSnap true;
```

Other (advanced)

- addLayersControl
  - Settings for adding layers of cells perpendicular to a surface
- meshQualityControls
  - Define criteria for cell quality
  - Will be used as criteria during meshing process