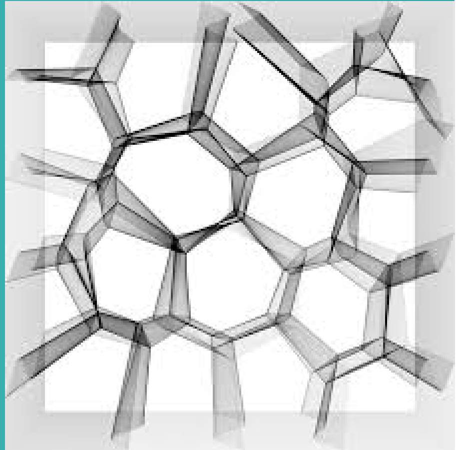
Mesh Generation Using The Voronoi Method

NUMGRID 2018 – Winter school
Dec 1st 2018



Stefano Paoletti
Siemens Industry Software
Via Durban 4 00144 Roma
Italy

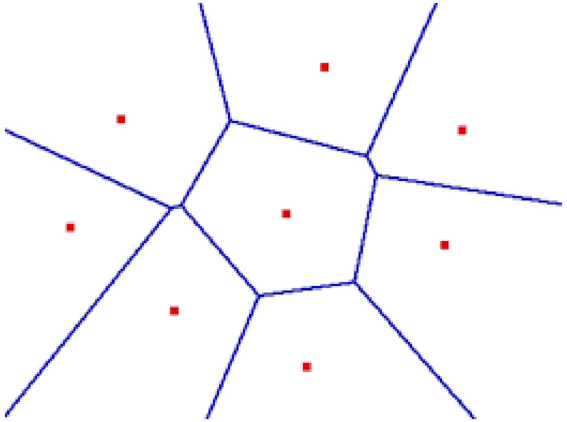
CONTENTS

Siemens Ingenuity for life

Introduction to Voronoi graph Properties

SIEMENS
Ingenuity for life

Definition of Voronoi cell:



$$R_k = \{x \in X \mid d(x, P_k) \le d(x, P_j) \text{ for all } j \neq k\}$$

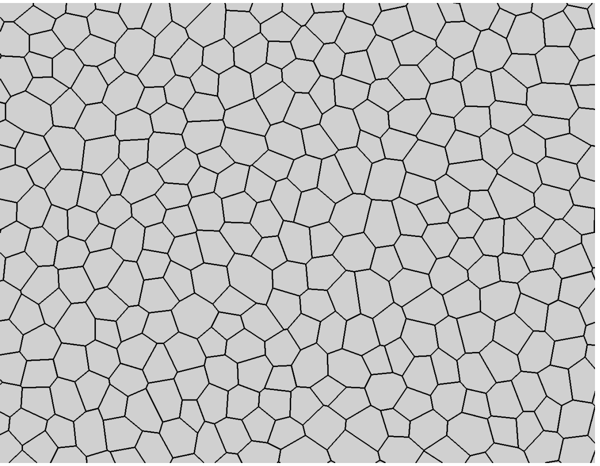
A Voronoi cell R_k is then the set of points that are closer (according to some distance function *d*(x,P)) to its generator P_k than to any other generator P_j

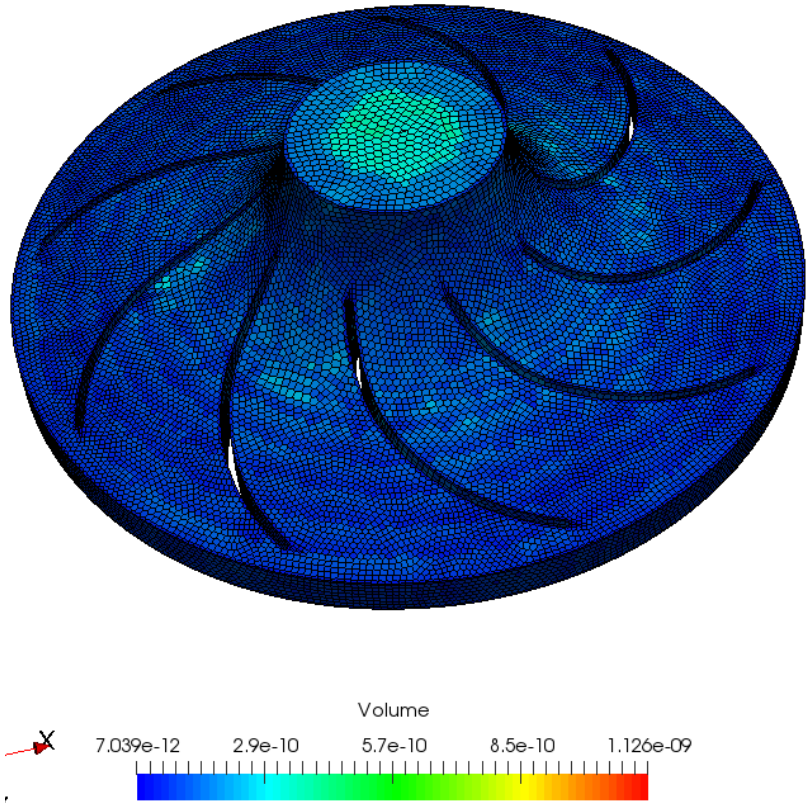
If the distance function is the usual euclidean distance then the definition of a Voronoi cell leads to several interesting and useful properties, in particular for the mesh generation.

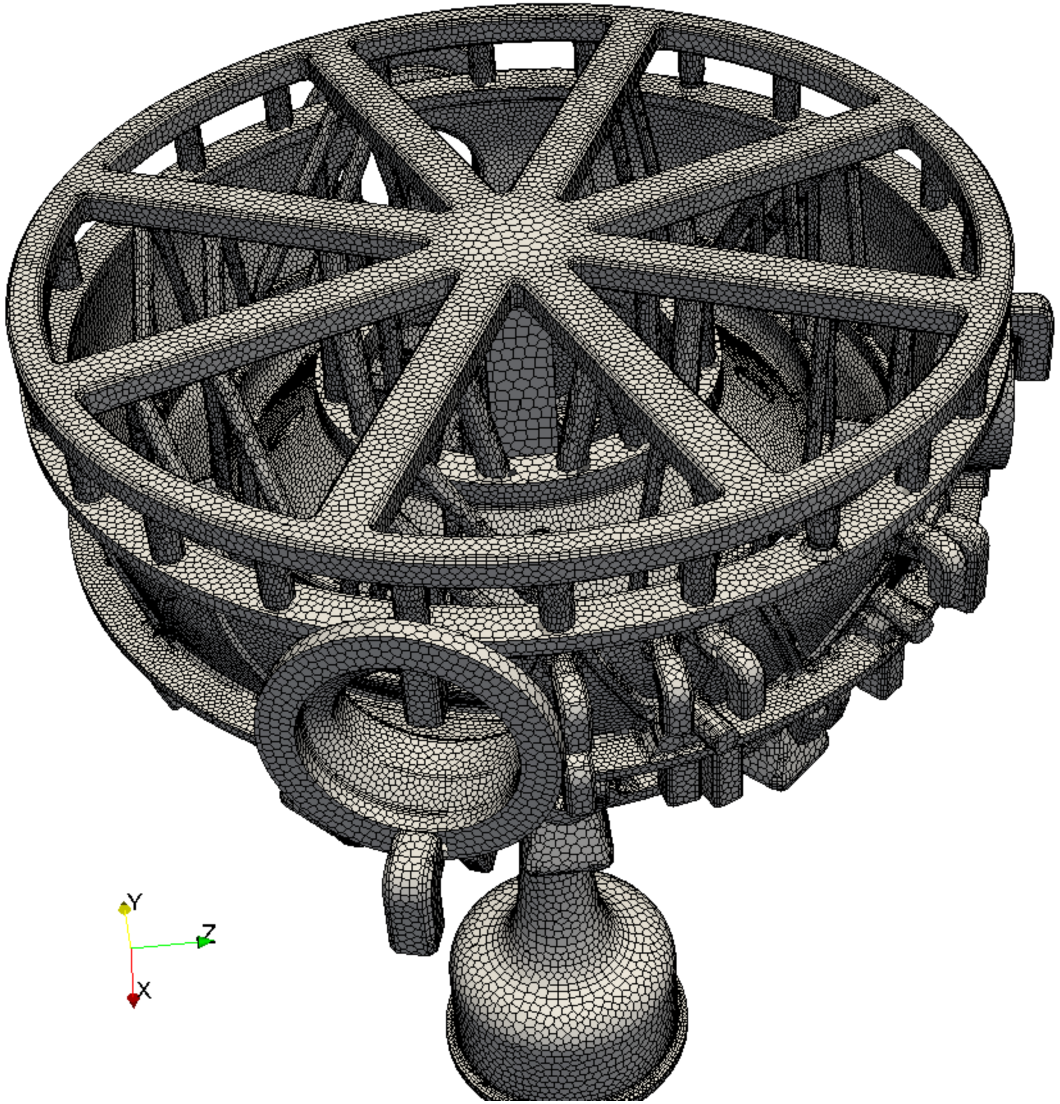
Introduction to Voronoi graph Properties

SIEMENS
Ingenuity for life

Given a set of generators, the Voronoi diagram is then the set of all the Voronoi cells that can be built, each around its own generator.







Introduction to Voronoi graph Properties

SIEMENS
Ingenuity for life

A Voronoi cell is particularly useful for solving partial differential equations using the Finite Volume Cell Centered Method.

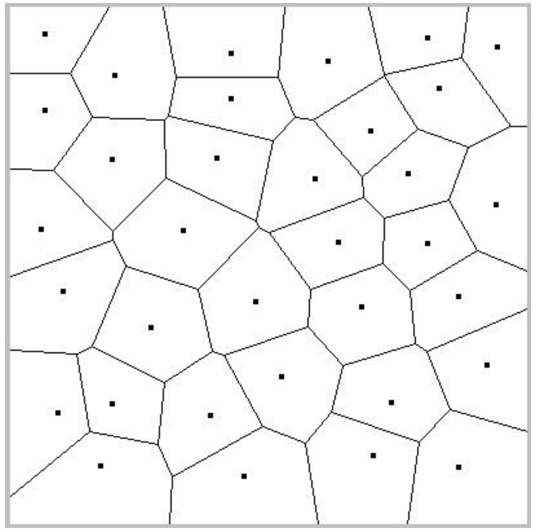
It is quite natural to identify the generators (or seeds) with the cell centers and the Voronoi cells with the Finite Volume Cells defined as a collection of faces forming a polyhedron.

Voronoi cells are general polytopes

A Voronoi cell has the following useful peculiarities:

- ***convex*** **shape (except possibly at the boundary)**
- **Has** ***positive volume***
- ***orthogonal*** **(i.e. the edge connecting the generators of two adjacent cells is orthogonal to the common face (in 3d)**
- **Its faces are** ***planar***

All of the above are highly desirable properties for a Finite Volume Cell



Voronoi 3D Mesh Generation methods

Indirect method:

- **By dualization of delaunay tedrahedrization**

Direct methods:

- **Incremental**
- **Cuboid**
- **(Fortune‘s 2d)**
- **etc.**

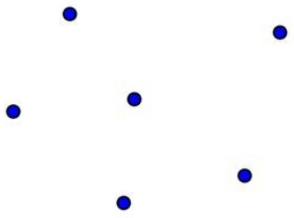
Voronoi Meshes by Indirect Method

SIEMENS
Ingenuity for life

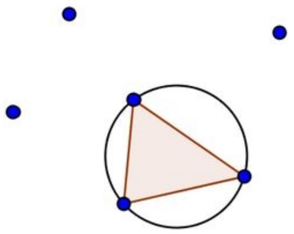
- **The dual of a Delaunay Tetrahedral mesh is a Voronoi diagram formed by Voronoi cells**

There are two reasons:

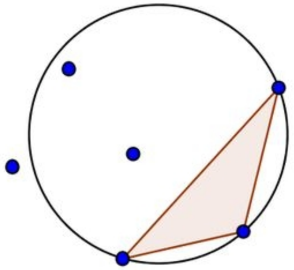
- **Tetrahedral meshers are already available**
- **Dualization can be generalized to non tetrahedral cells**



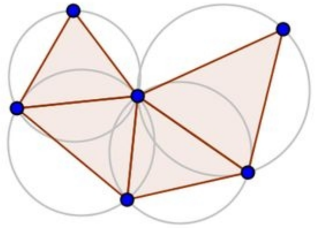
A collection of points



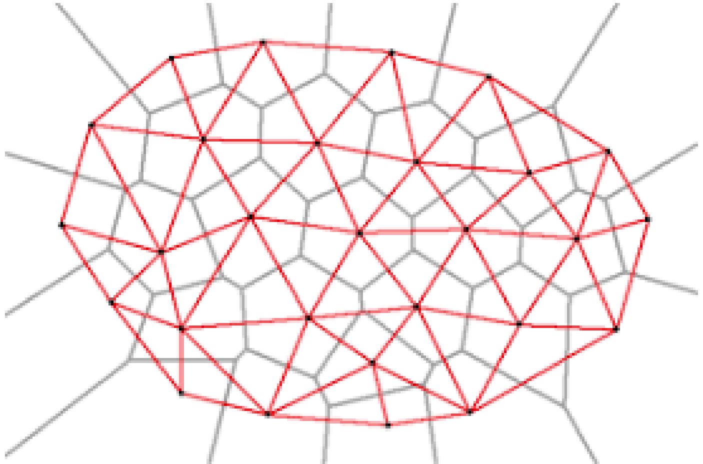
A Delaunay triangle



A non-Delaunay triangle



A Delaunay triangulation

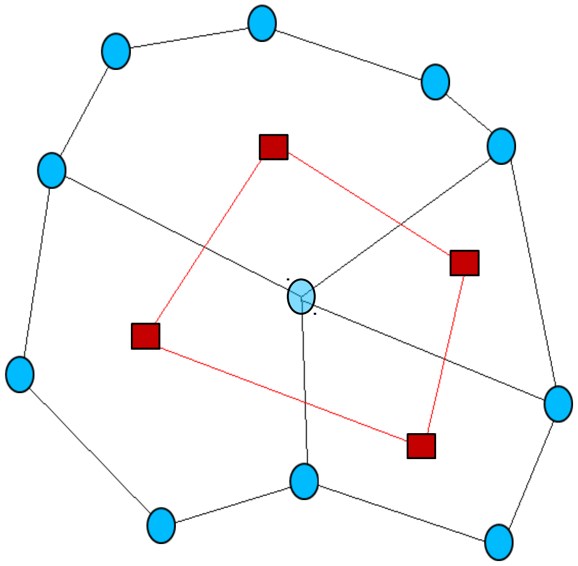


Duality

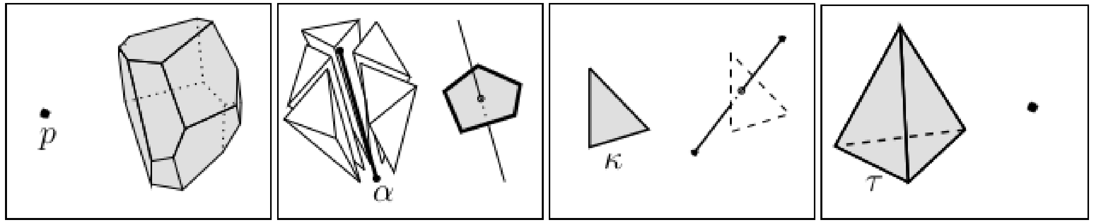
Using the following conversion tables it is practically possible to dualize any mesh both in 2D and in 3D.

C=Cell
F=Face
E=Edge
V=Vertex

Primal 2D	Dual 2D
F	V
E	E
V	F



Primal 3D	Dual 3D
C	V
F	E
E	F
V	C



(a) (b) (c) (d)

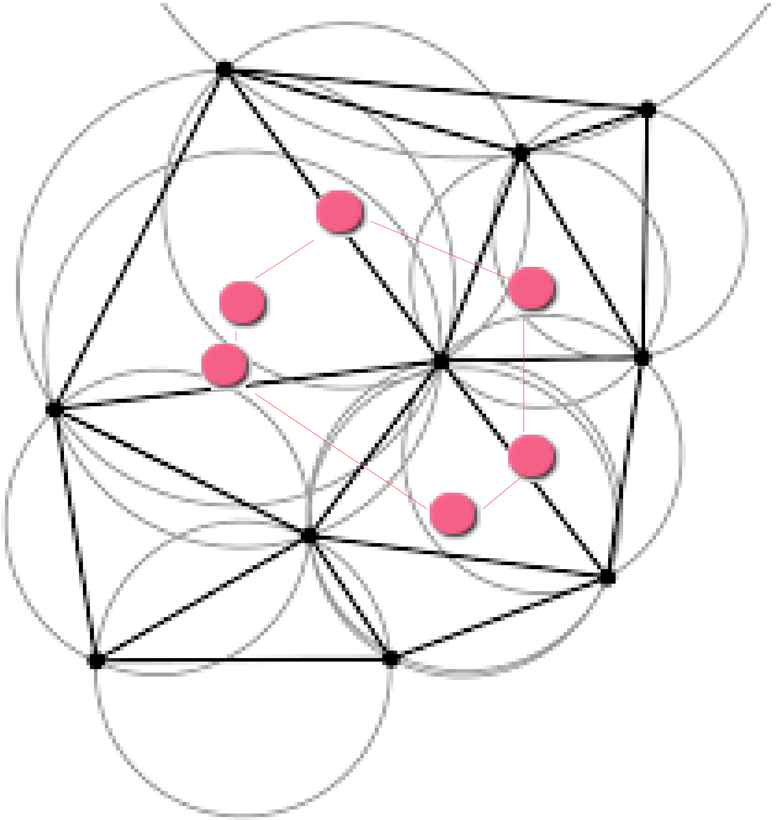
Dualization

SIEMENS
Ingenuity for life

In order to build a voronoi mesh from a delaunay mesh it would be mandatory to create the dual vertex of a cell/face as the circumcenter of the primal tetrahedron/triangle

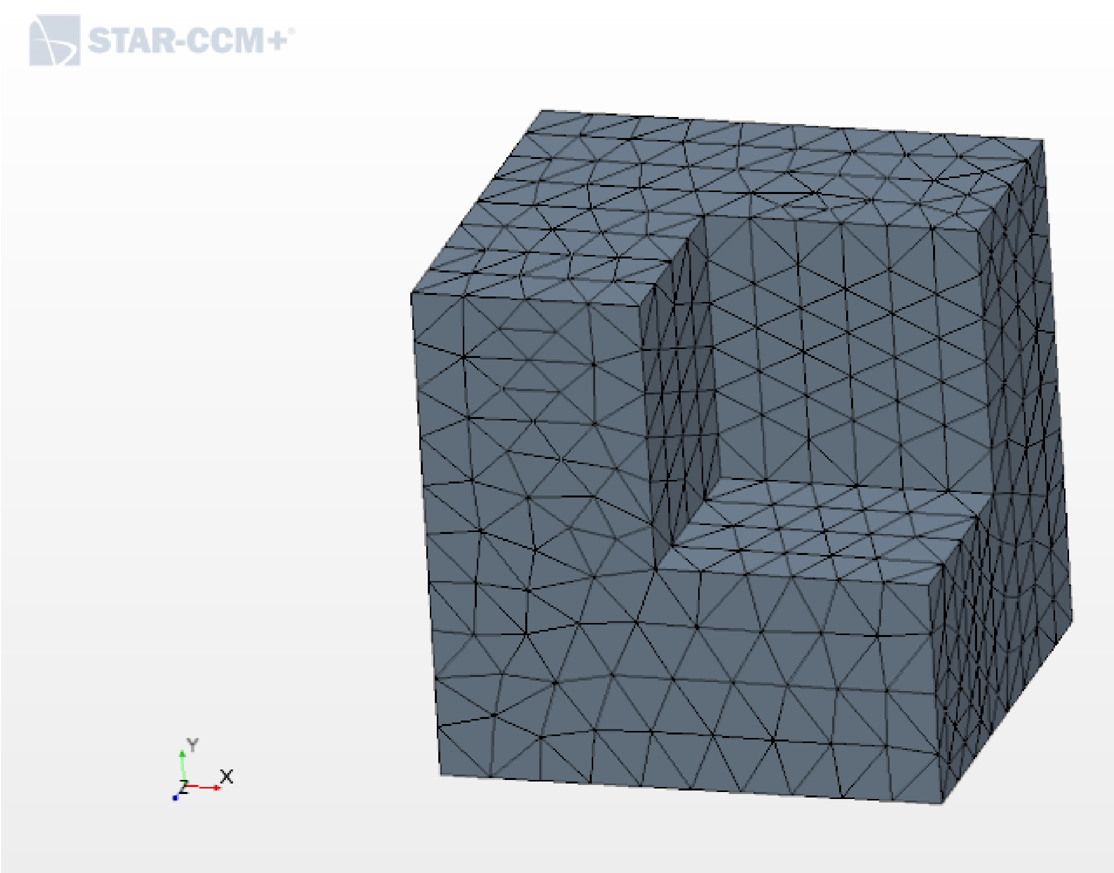
Primal 2D	Dual 2D
F	V
E	E
V	F

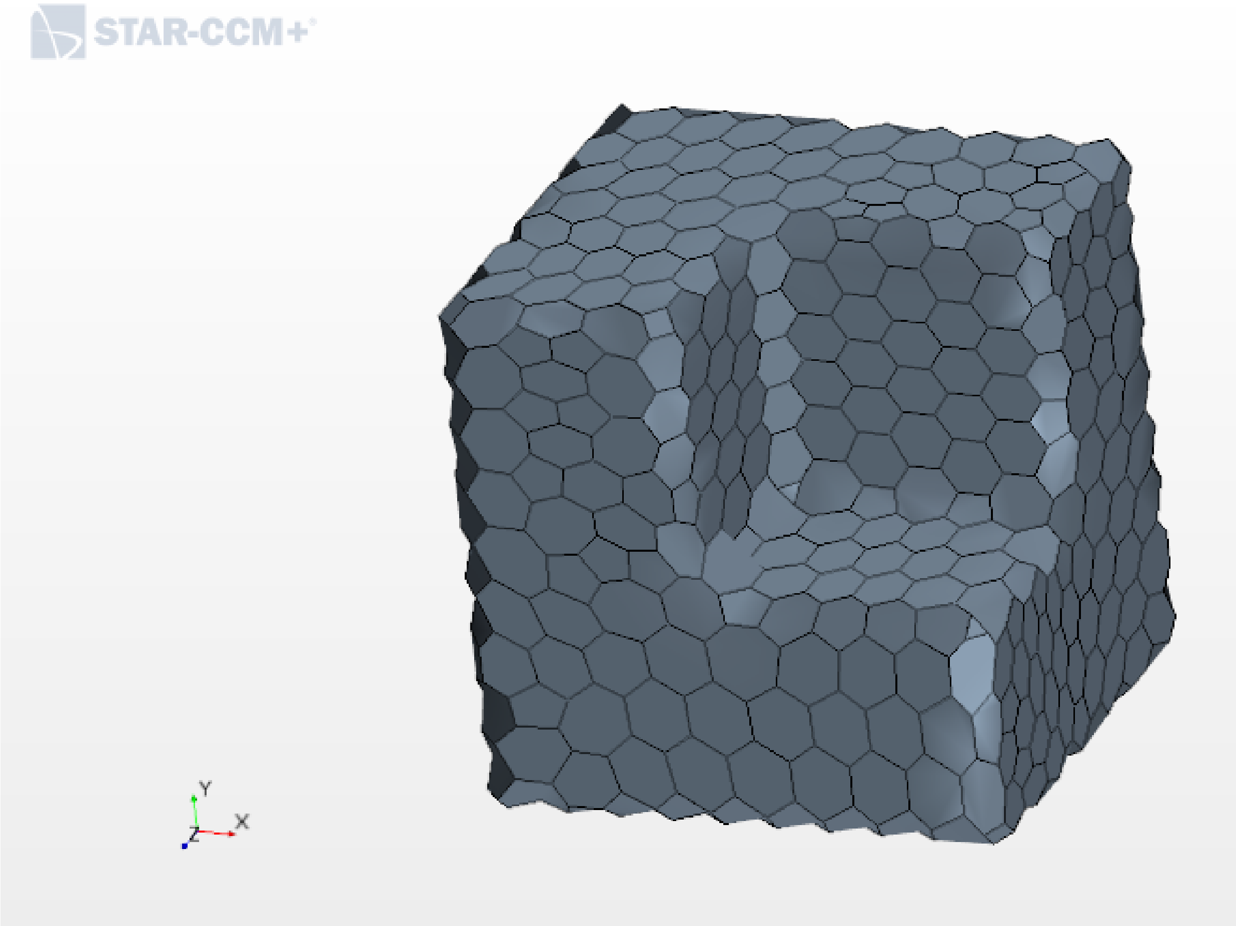
Primal 3D	Dual 3D
C	V
F	E
E	F
V	C



Recovering the feature edges and corners

Using the “raw” dualization method none of the primal entities is preserved in the dual mesh.





Recovering the feature edges and corners

However in the overwhelming majority of the cases the primal mesh has:

- **Sharp edges**
- **Edges that limit surface zones where the user wants to apply a boundary condition**
- **Etc.**

Recovering the feature edges and corners

A primal mesh possesses features that need to be honored in the dual mesh:

- **Edges of the primal mesh that need to be preserved in the dual mesh (in an approximate way)**
- **Corners: vertices of the primal mesh that need to be preserved**

Feature corners could be identified as:

- **those primal vertices where 3 or more feature edges meet.**
- **those primal vertices where 2 feature edges meet at a sharp angle**

Recovering the feature edges

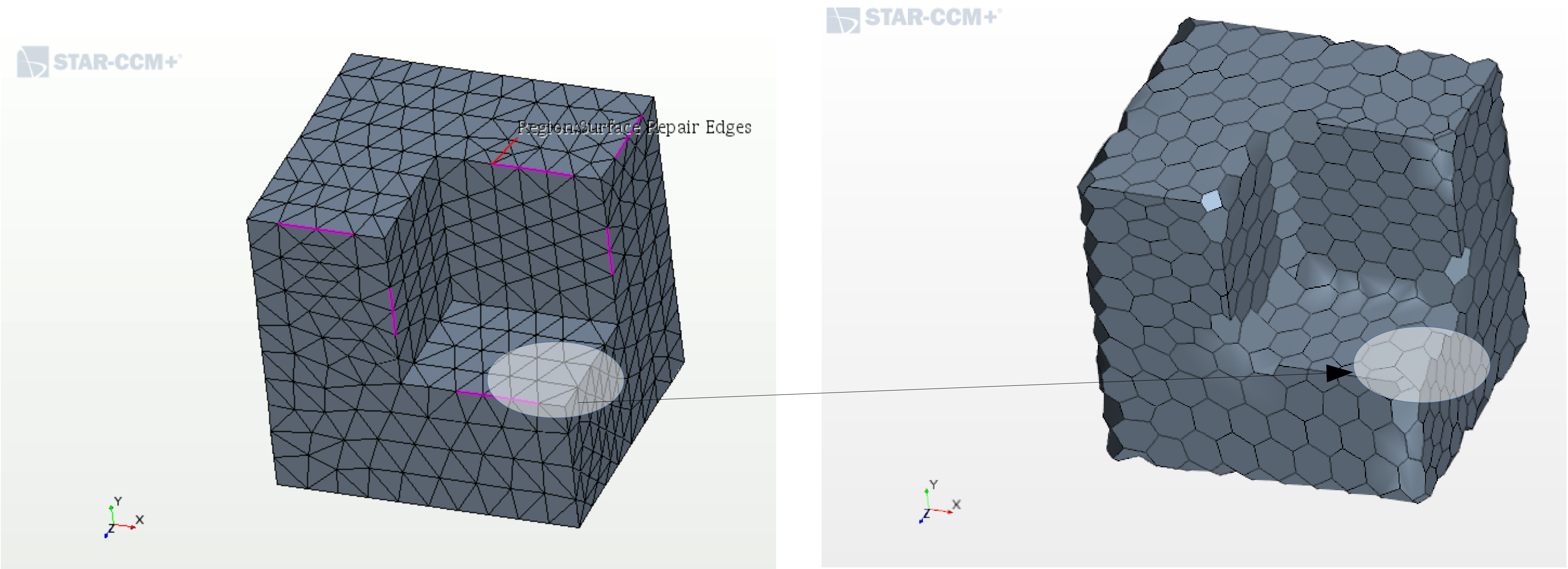
SIEMENS
Ingenuity for life

Feature edges recovery

The primal vertex is shared by 2 feature edges.

The dual face of the primal boundary boundary vertex needs to be modified by:

- **Splitting the dual edge of the 2 primal feature edges with their midpoint**
- **Splitting the dual face with an edge connecting the 2 newly created dual vertices**



Recovering the corners

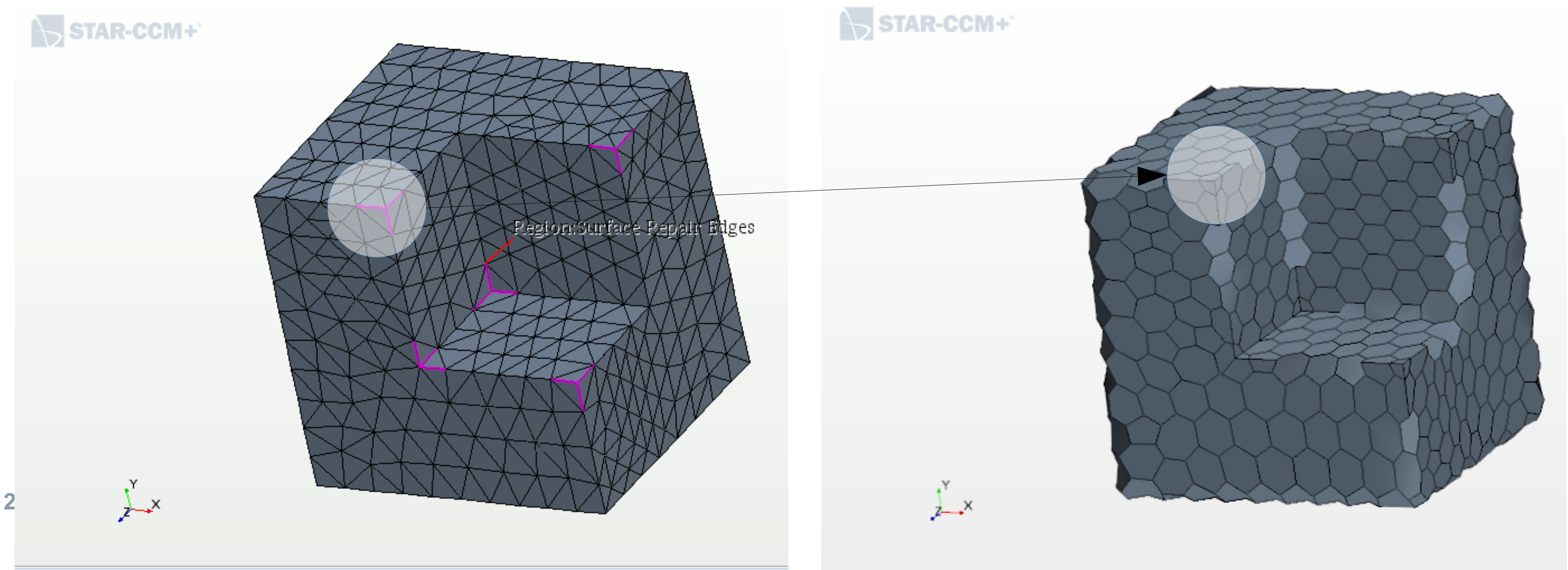
SIEMENS
Ingenuity for life

Feature corners recovery

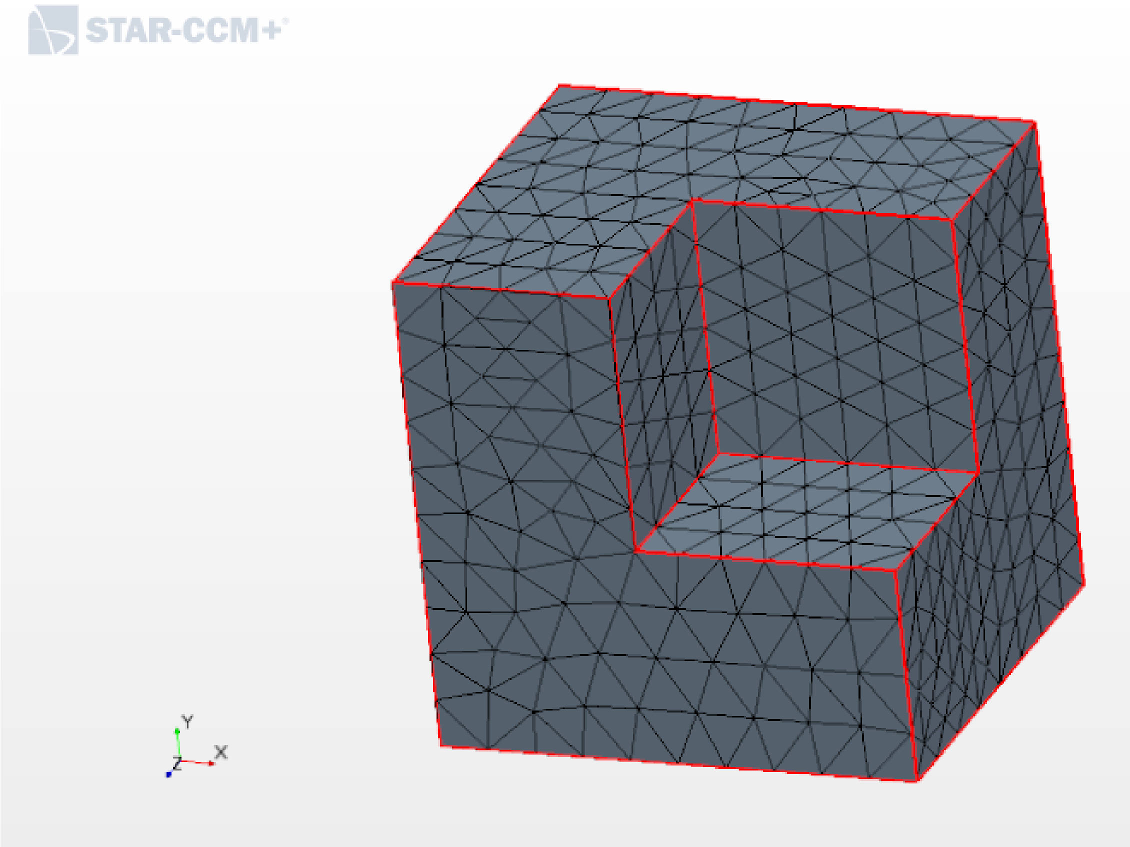
The primal vertex is shared by 3 or more feature edges.

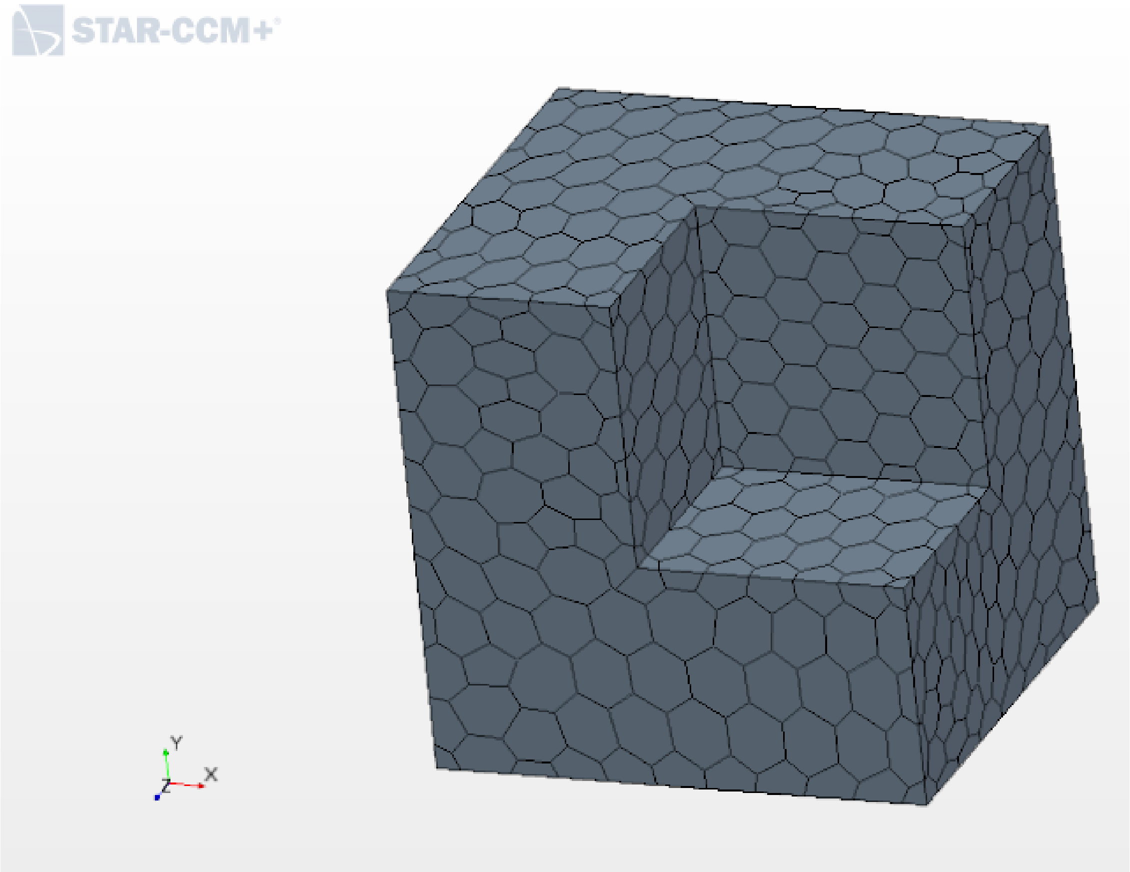
The dual face of the primal boundary boundary vertex needs to be modified by:

- Ordering the feature edges (either clockwise or counter-clockwise)
- Splitting the dual edges of all the primal feature edges with their midpoints
- Splitting the dual face in a number of dual faces equal to the number of primal feature edges using the primal feature vertex as a hub



Recovering all the feature edges and corners





Splitting concave cells

Every concave edge vertex and every corner in the primal mesh produce a concave dual cell. The quality of concave dual cells is quite low. There are cases when the center of the cell is outside the cell itself.

It is mandatory to remove the concave cells.

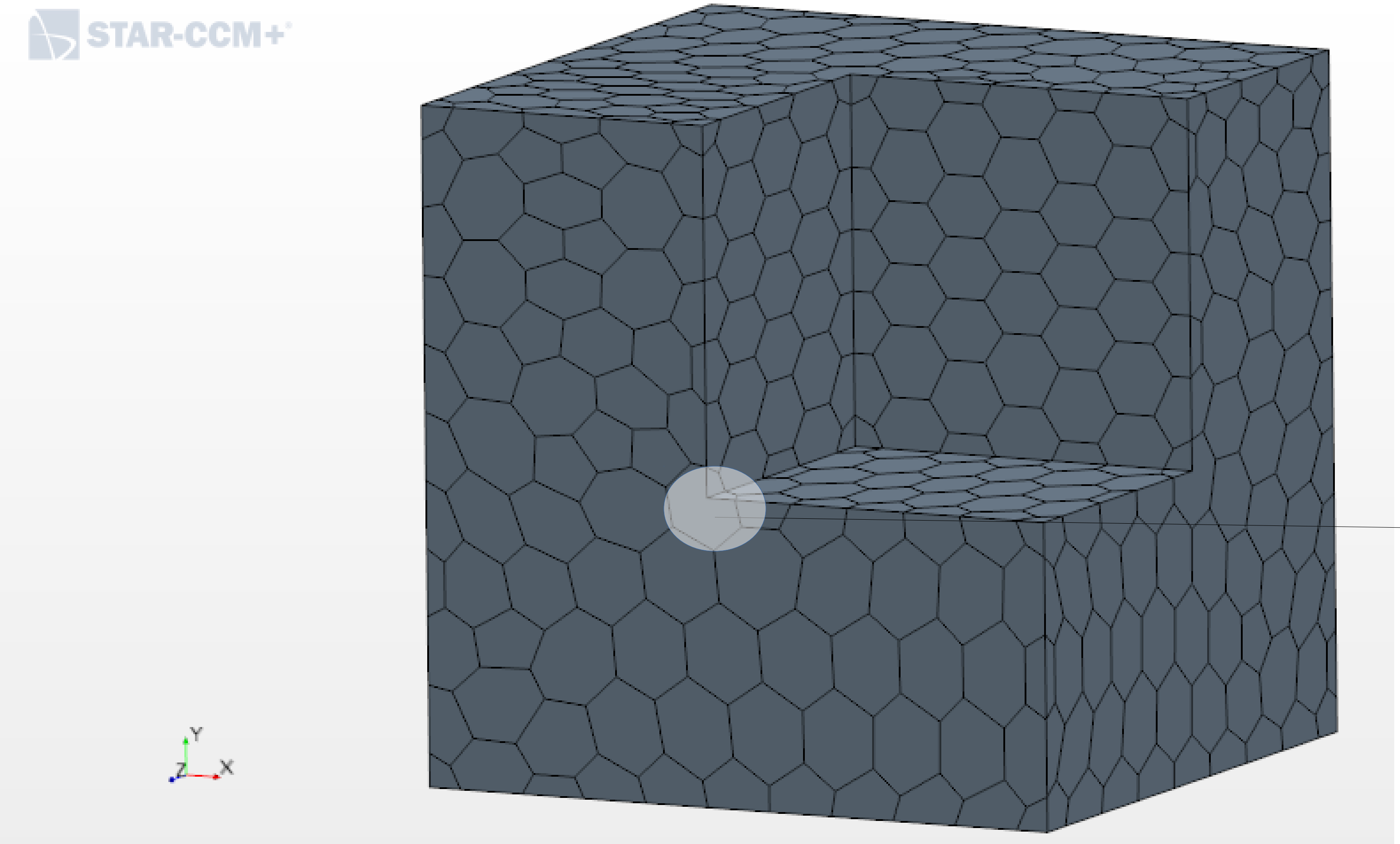
A very simple way would be to subdivide the concave cell into tetrahedra.

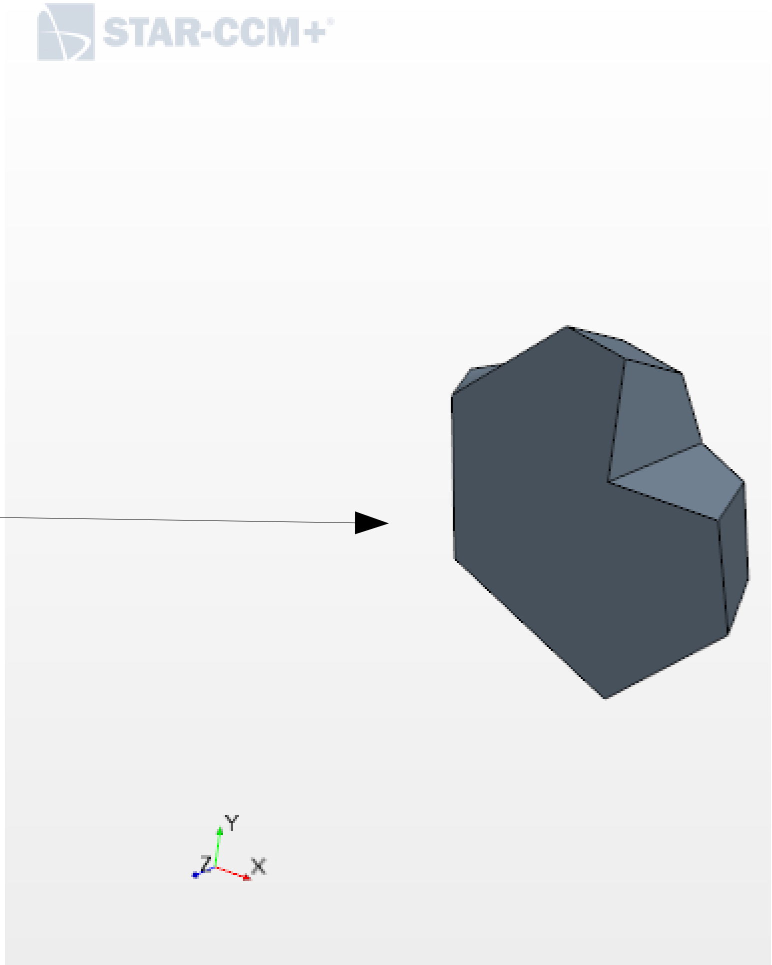
A better and relatively straigthforward way is to split the cells at their concave edges.
(see Bernard Chazelle: Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm)

Intuitively one can think of the splitting process as the action of a knife that cuts the cell at the concave edge(s) along the bisector of the dihedral angle formed by the two faces sharing the edge.

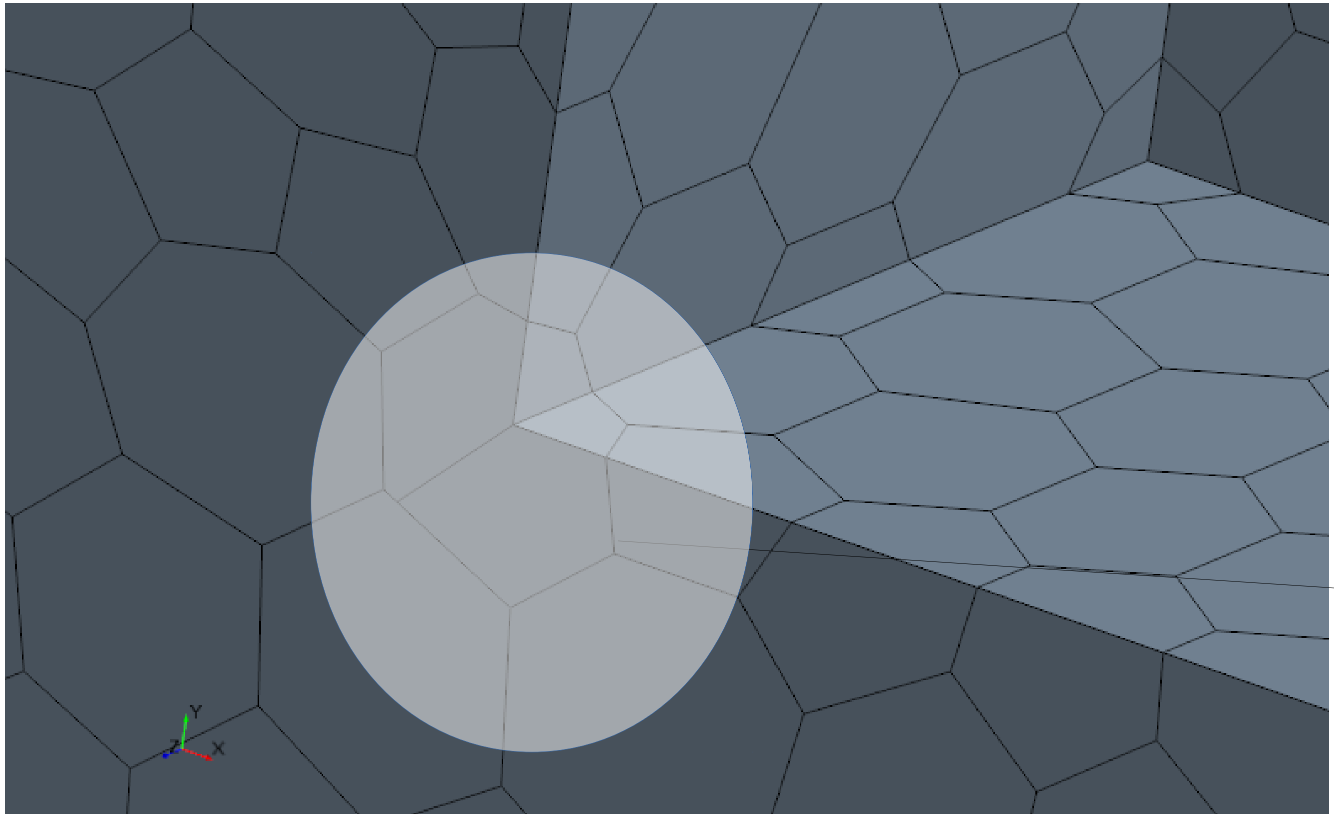
Splitting concave cells

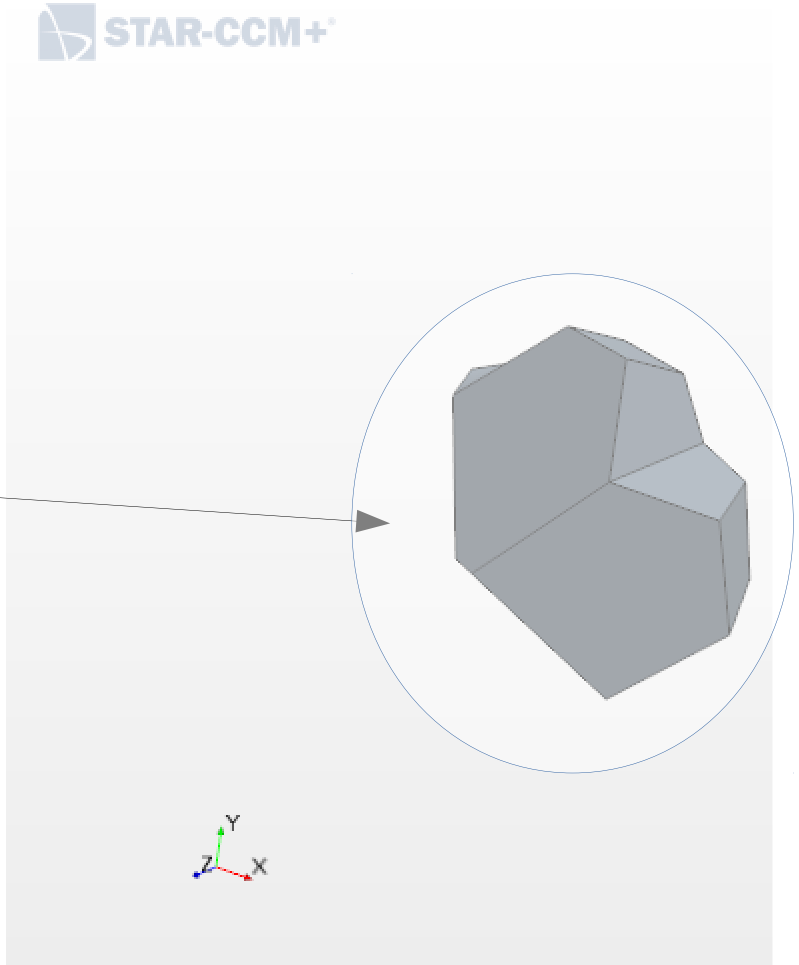
SIEMENS
Ingenuity for life





Splitting concave cells

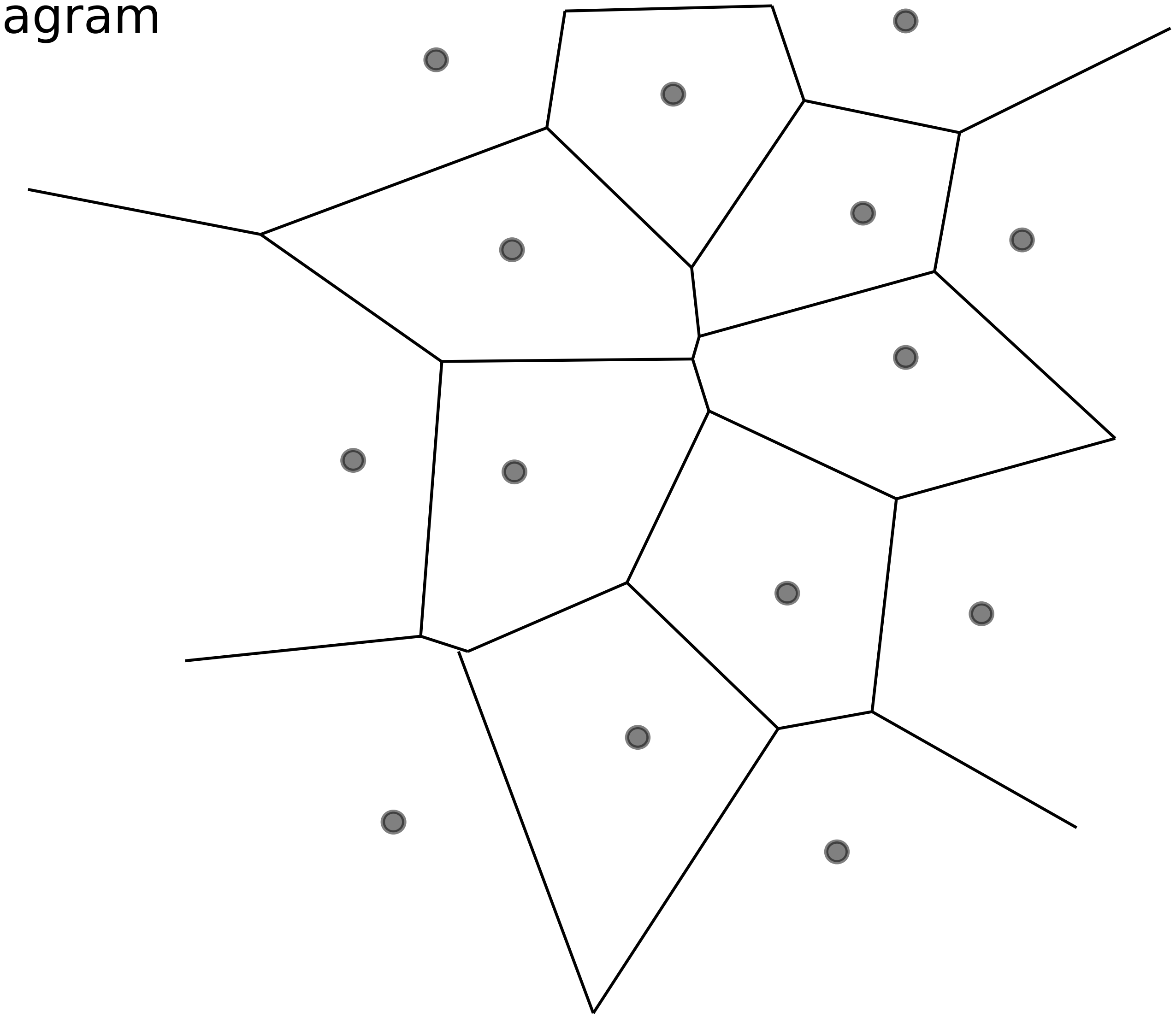




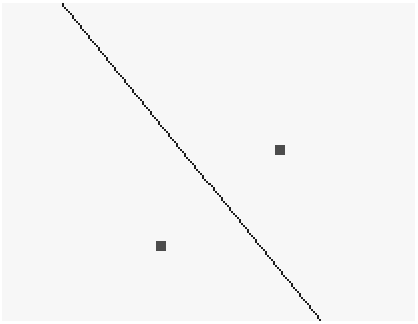
Voronoi Mesh generation direct method : incremental method

Starting with the Voronoi diagram

of a subset of the seeds

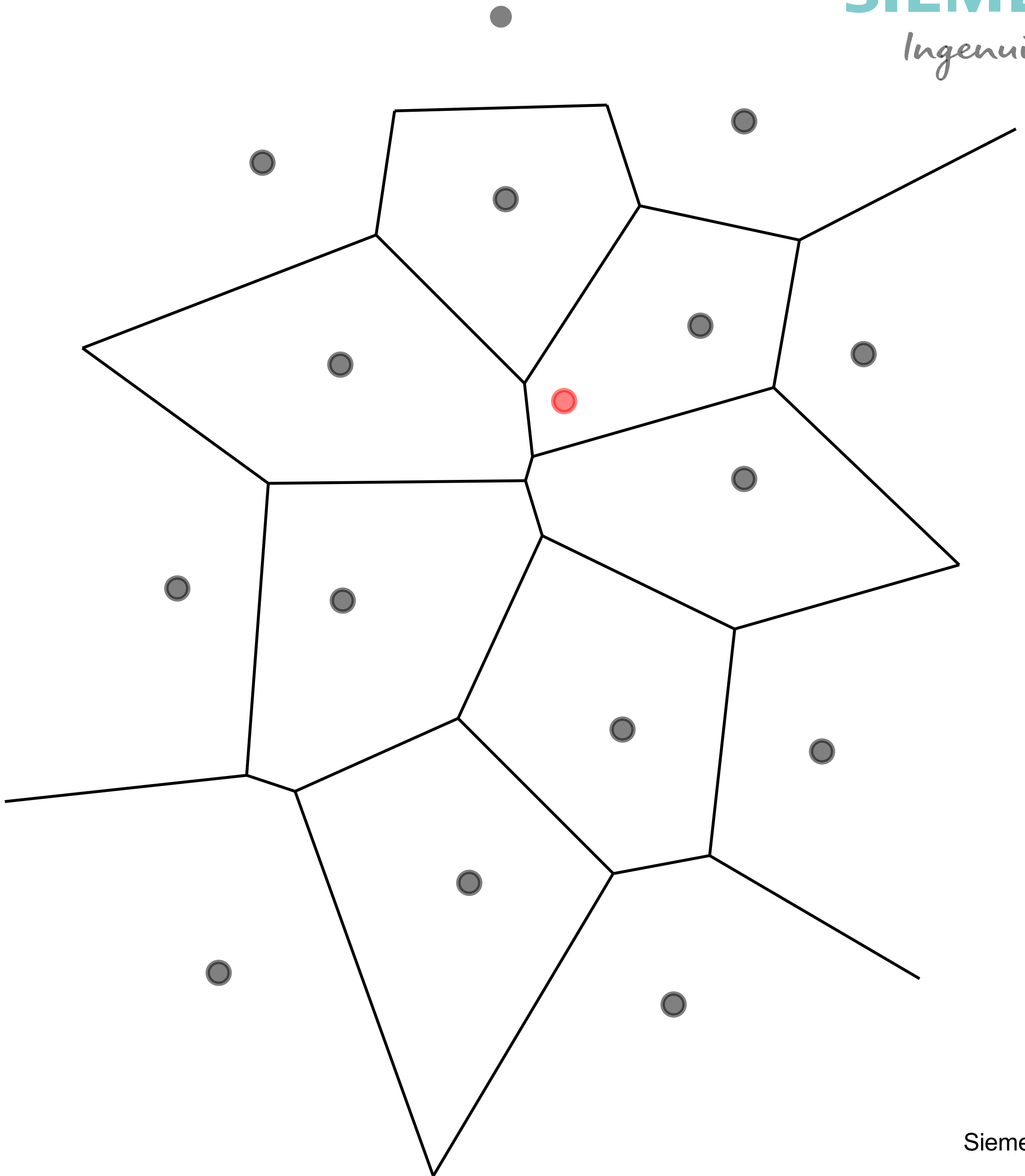


At the very beginning
it is started with 2 seeds



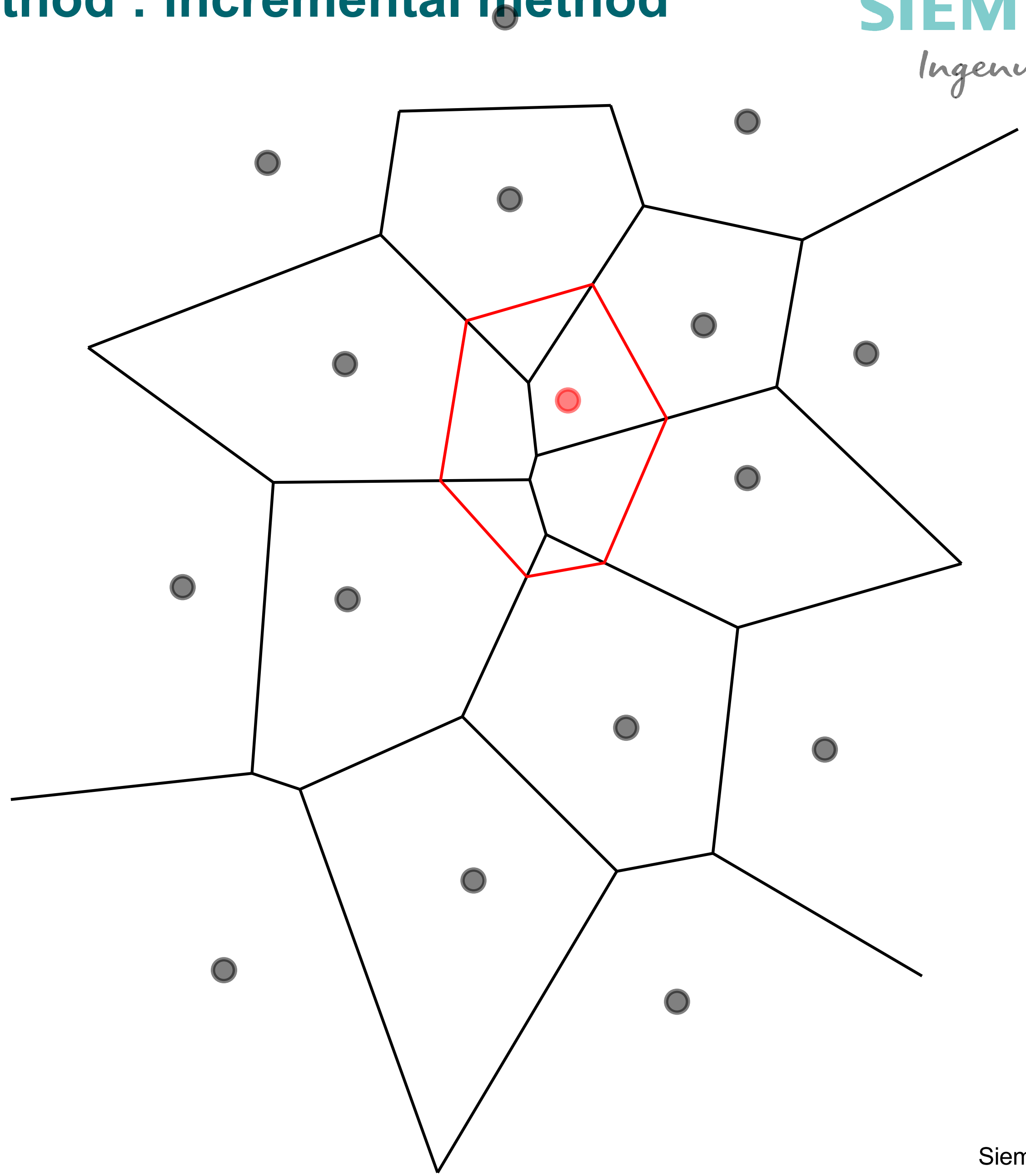
Voronoi Mesh generation direct method : incremental method

Inserting a new seed



Voronoi Mesh generation direct method : incremental method

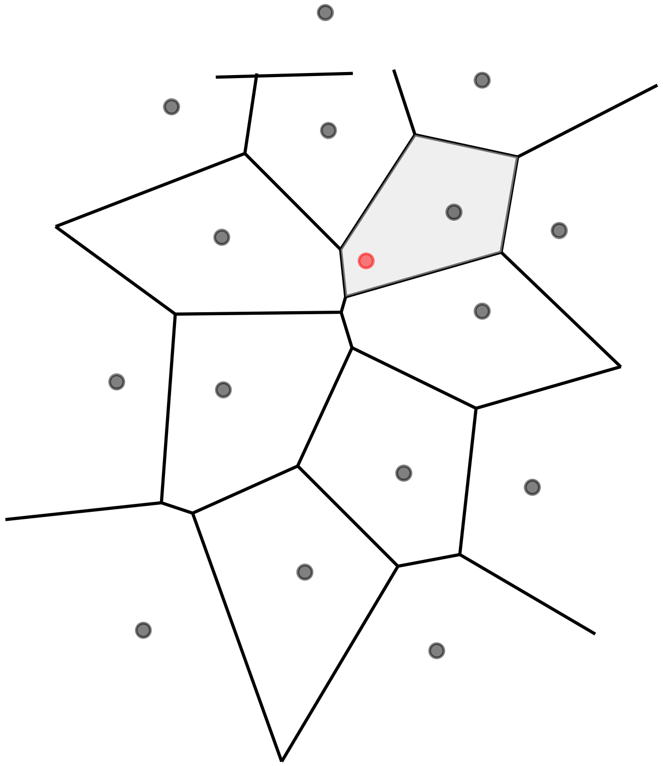
Computing the new voronoi cell



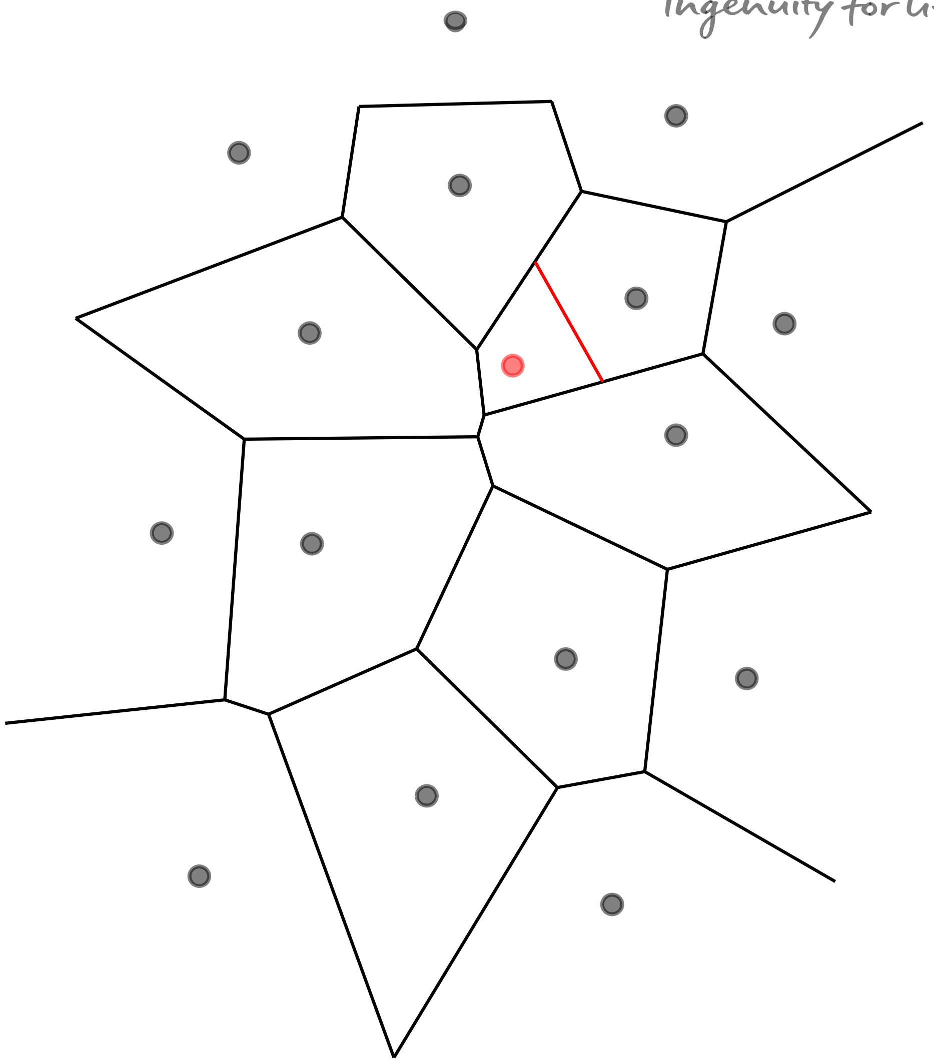
Voronoi Mesh generation direct method : incremental method

SIEMENS
Ingenuity for life

How to do it ?



Find the voronoi cell that
contains the new seed

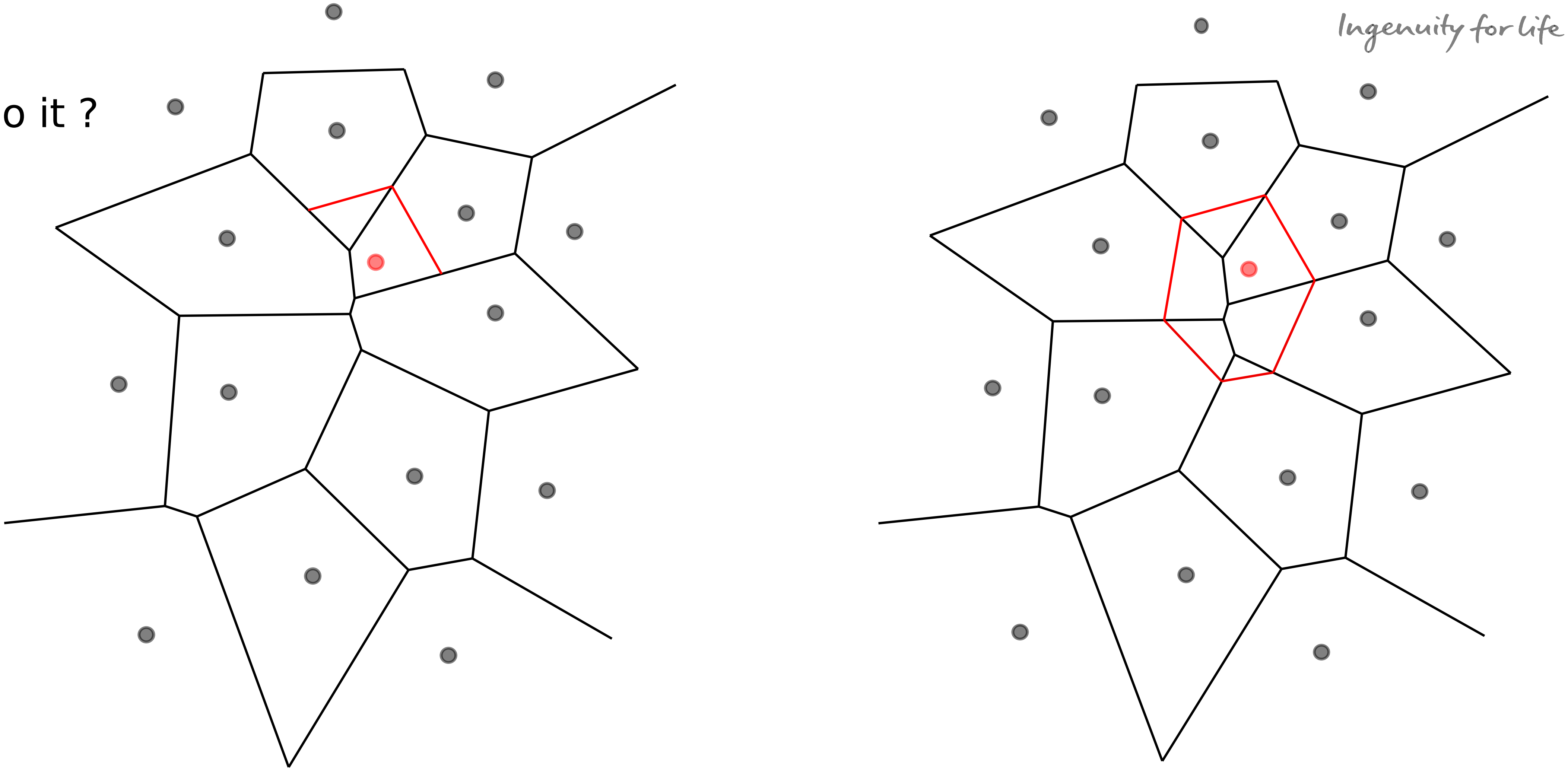


Split with the bisector between the new seed
and the seed of the containing cell.

Voronoi Mesh generation drect method : incremental method

SIEMENS
Ingenuity for life

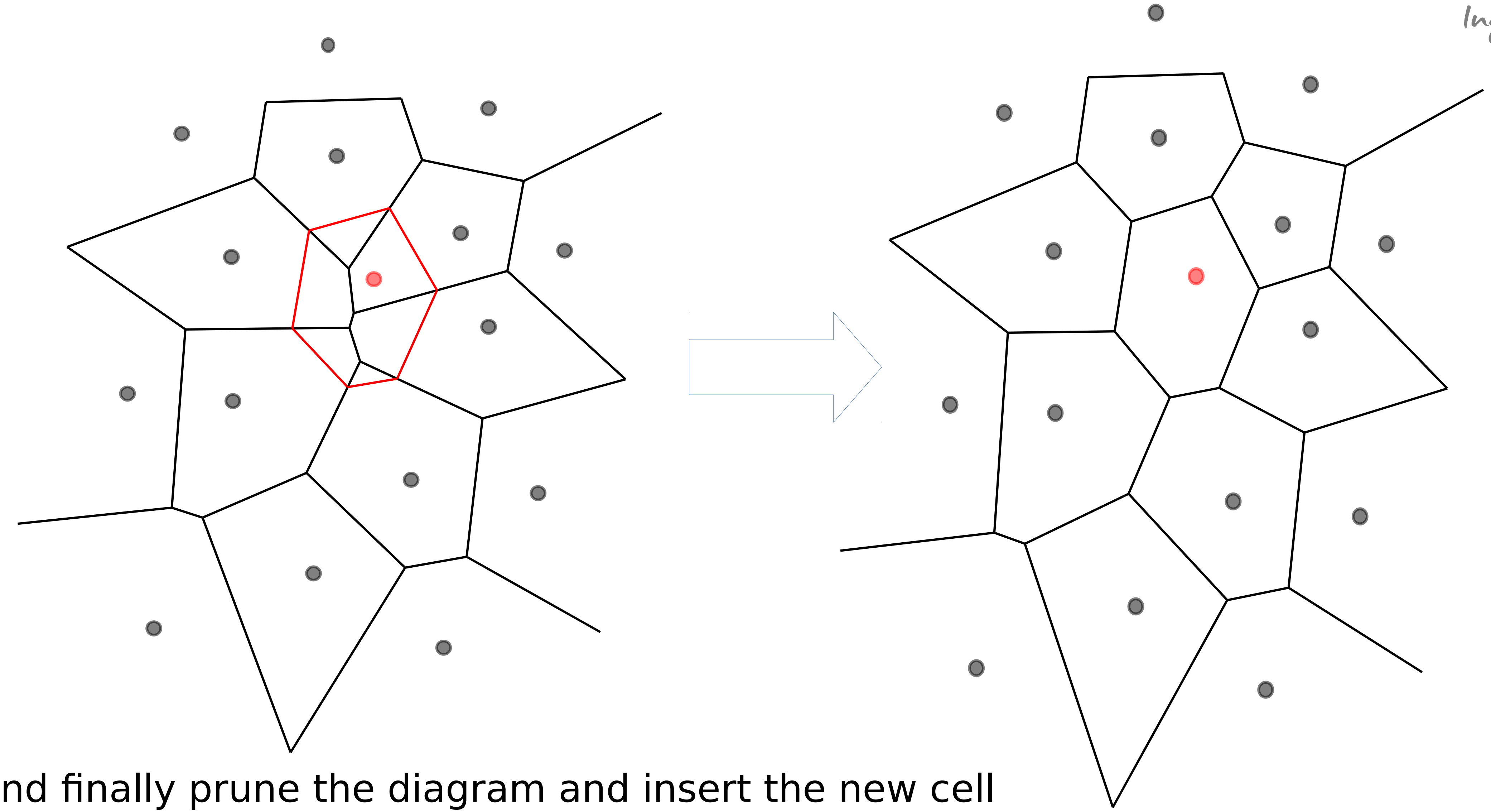
How to do it ?



Keep splitting using the adjacent cells ....until no more splitting is possible

Voronoi Mesh generation direct method : incremental method

SIEMENS
Ingenuity for life



And finally prune the diagram and insert the new cell

Voronoi Mesh generation direct method : the cuboid method

SIEMENS
Ingenuity for life

Pros:

- ***Based on simple successive cutting of a cuboid with planes orthogonal to the midpoint of the edge connecting a generator with its neighbors***
- ***Can be fast***
- ***Can be parallelized***

Cons:

- ***Requires the a-priori definition of the position of the generators (the seeds), although allows a-posteriori mesh modifications by local mesh regeneration***
- ***Requires a fast search-tree to find the neighbors***
- ***Does not recover the boundary of the input surface, therefore requires ad-hoc procedure like clipping or placement of auxiliary seeds.***

See the voro++ library: http://math.lbl.gov/voro++ for a free library for generating Voronoi polyhedral meshes

Voronoi Mesh generation by cuboid method

SIEMENS
Ingenuity for life

- ***The method uses the idea of a "cuboid" around the seed that is successively cut and updated by planes orthogonal to the segment connecting the seed to the neighbors seeds and passing through the mid point of that segment (the bisector).***
- ***A fast binary tree is needed to search for the neighbors seeds***
- ***If the neighbor seeds are ordered by distance there is a simple criterion to stop searching and cutting***
  - ***stop when the distance of the next neighbor from the seed is greater than twice the distance of the farthest vertex in the current cuboid.***

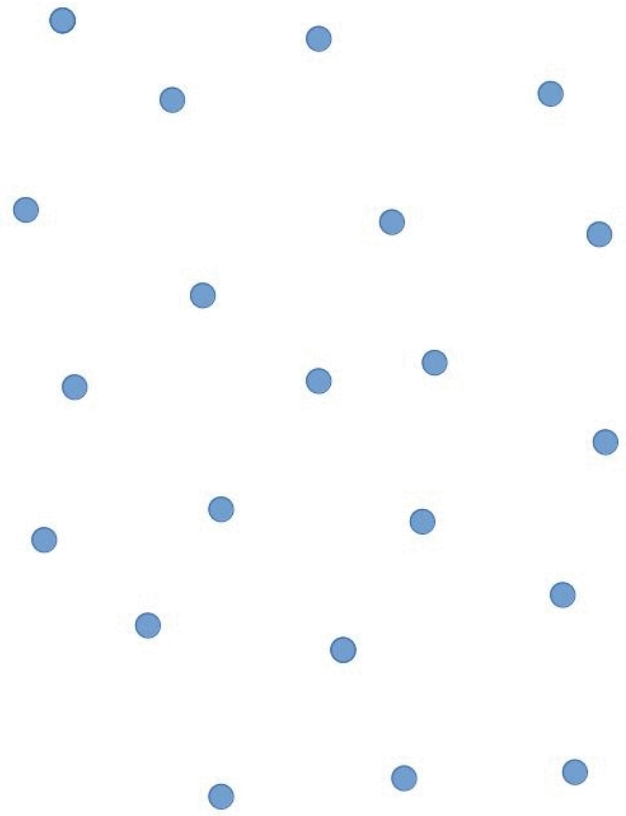
- ***The initial cuboid must completely include all the input surface that describes the object that we are about to mesh.***

Voronoi Mesh generation by the cuboid method

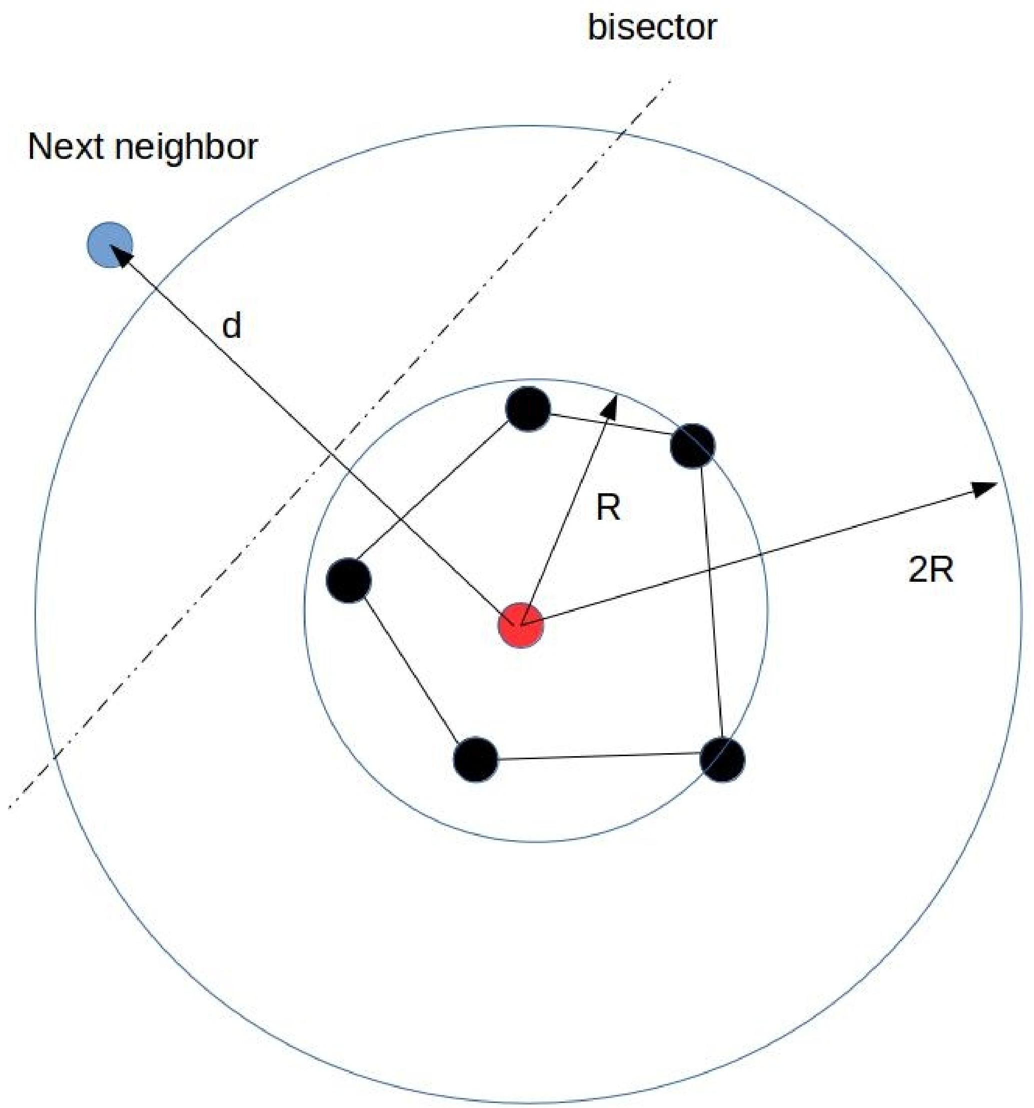
Around every seed (aka generator) the generation is performed in the way outlined in the animation.

The final “raw” mesh is then the set of all the updated cuboids built around each seed.

Since every initial cuboid is identical the final “raw” mesh is bounded by the ‘cuboid’



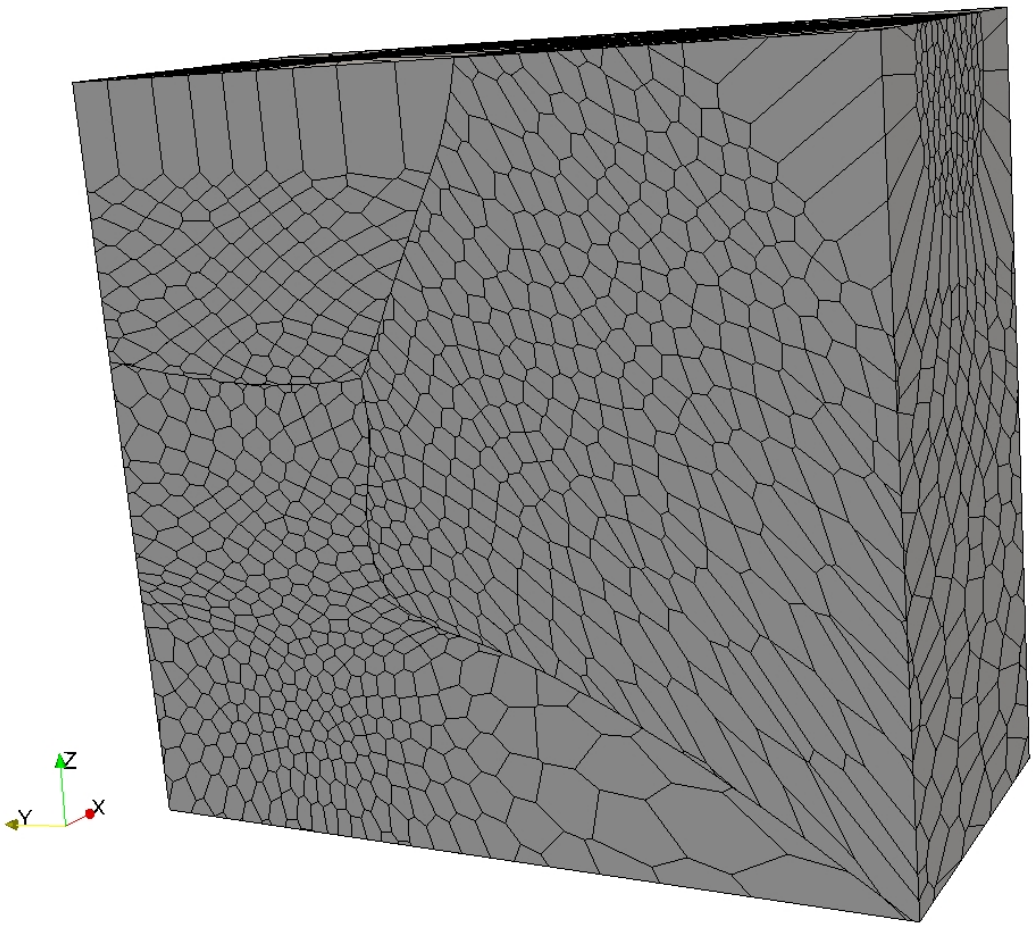
Voronoi Mesh generation by cuboid method: stop criterion

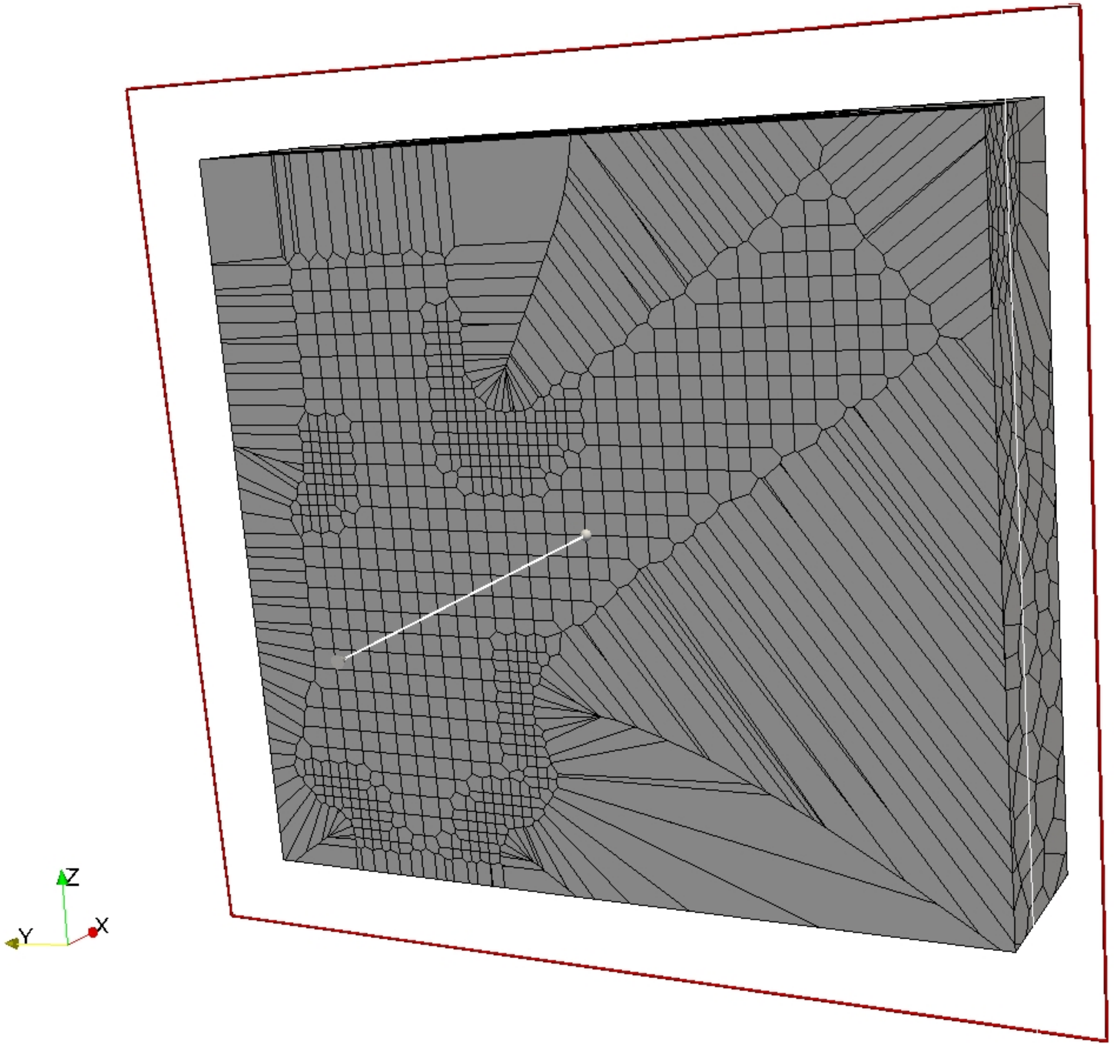


Stop when d>2R

Voronoi Mesh generation by cuboid method

SIEMENS
Ingenuity for life





Voronoi Mesh generation by cuboid method

SIEMENS
Ingenuity for life

Clipping the diagram to recover the input domain

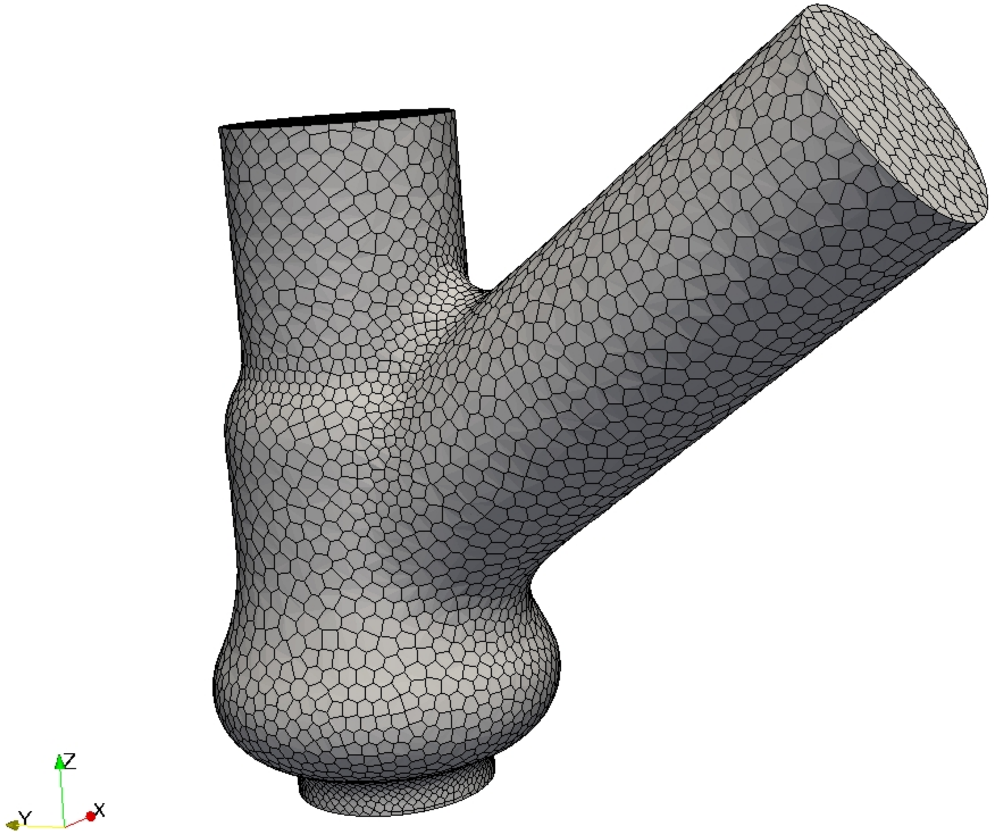
- ***Intersect the unclipped voronoi mesh with the input triangulation***
  - ***The process involves the clipping of all the cells whose seeds are coincident with the vertices of the input surface or that have vertices outside it***

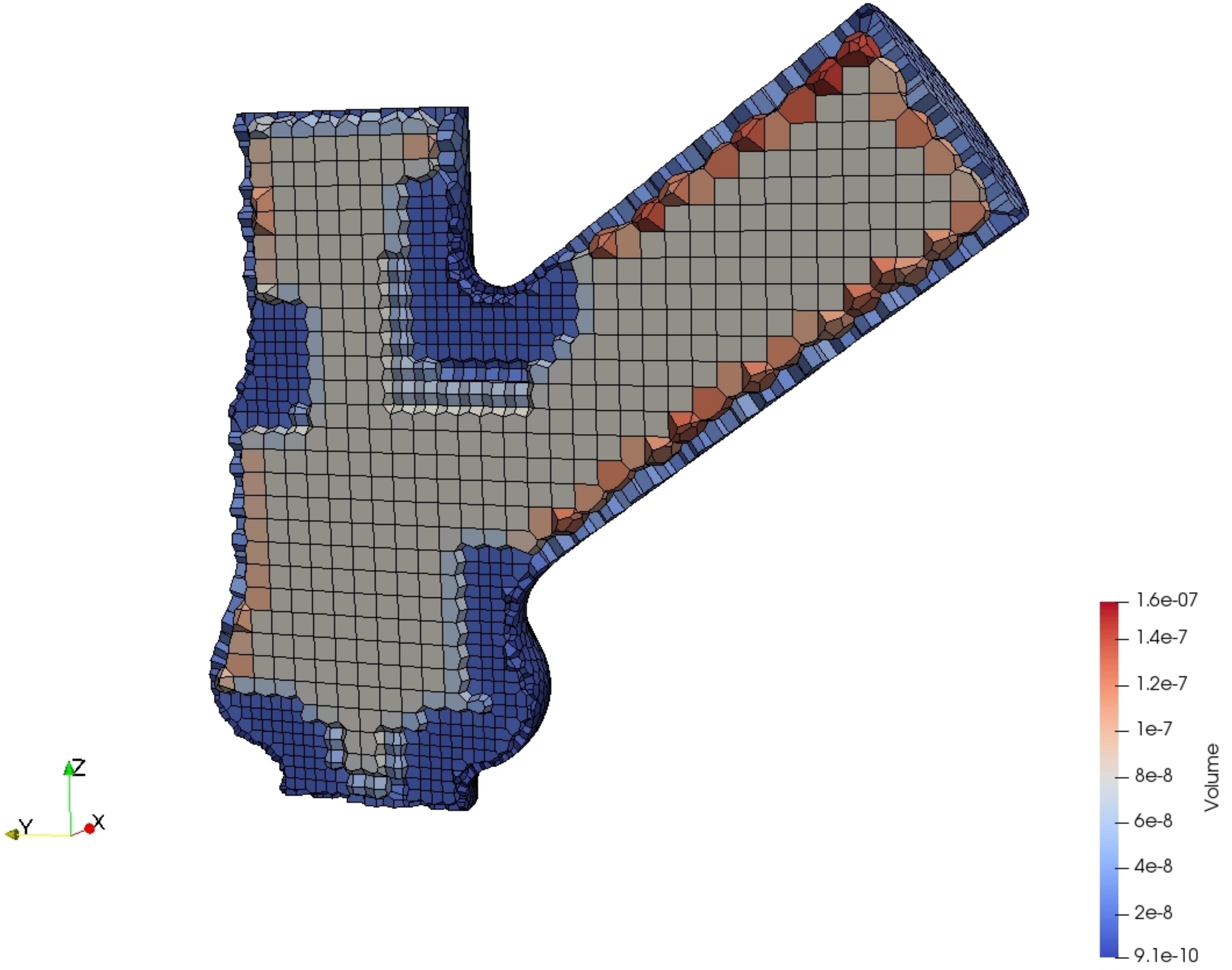
$$C_i = V_i \cap S$$

C_i *is the clipped Voronoi cell*
V_i *is the unclipped Voronoi cell*
S *is the input surface*

Voronoi Mesh generation by cuboid method after clipping

SIEMENS
Ingenuity for life

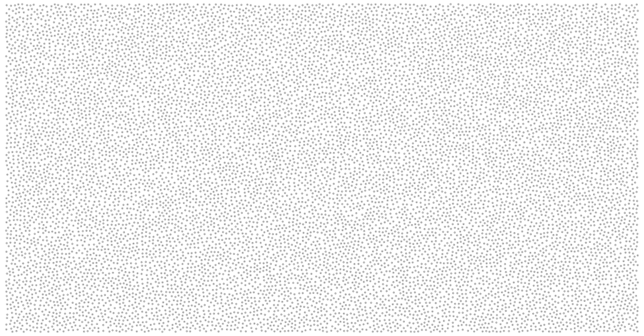




Seed points generation

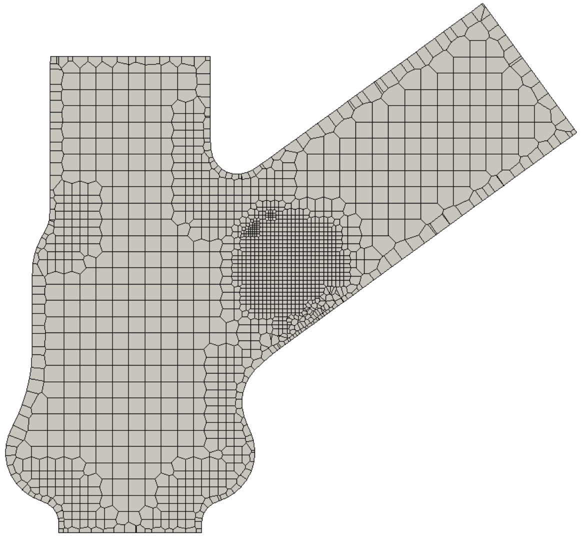
Defining the position of the generators (the seeds):

Poisson Disk Sampling to obtains a non-biased isotropic distribution

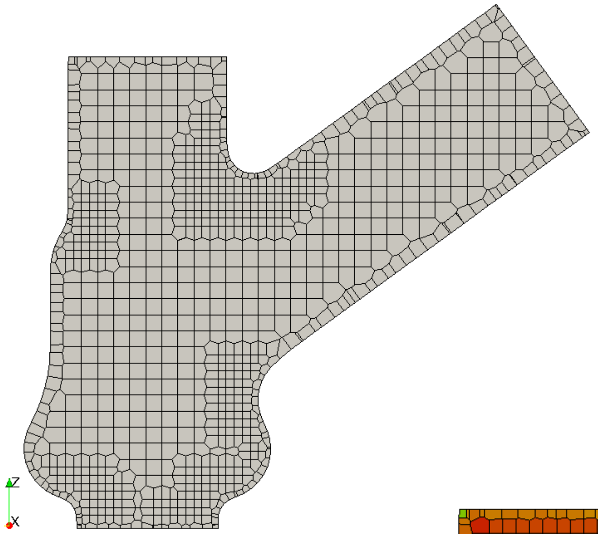


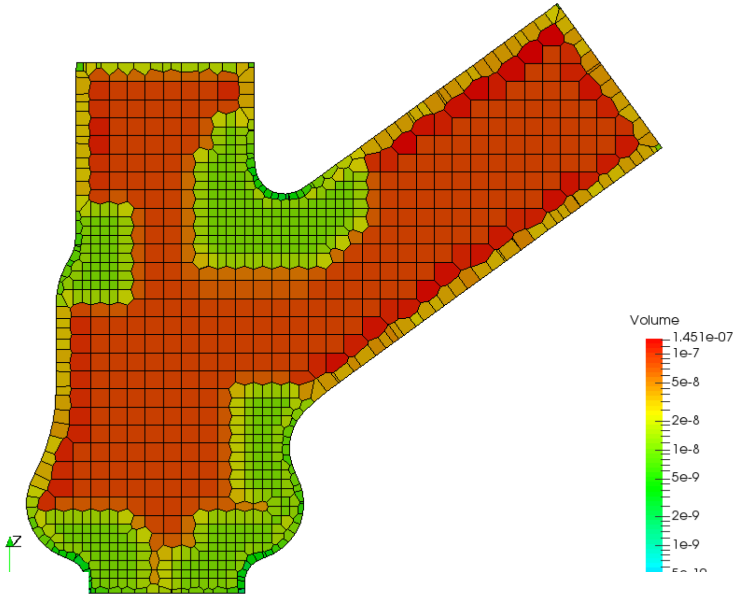
Octree-based generation using the centers of the octants (biased distribution)

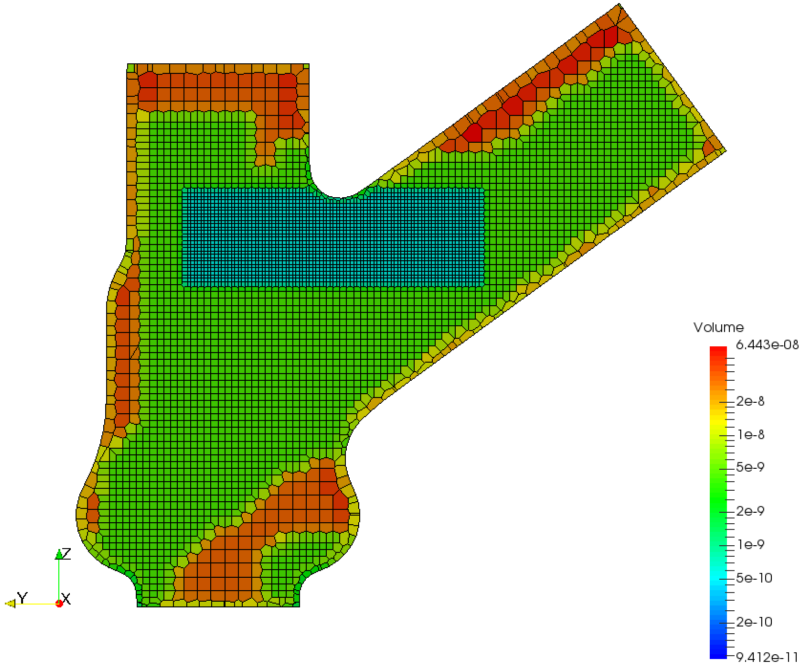
Specialized point distribution methods are also possible (more later)

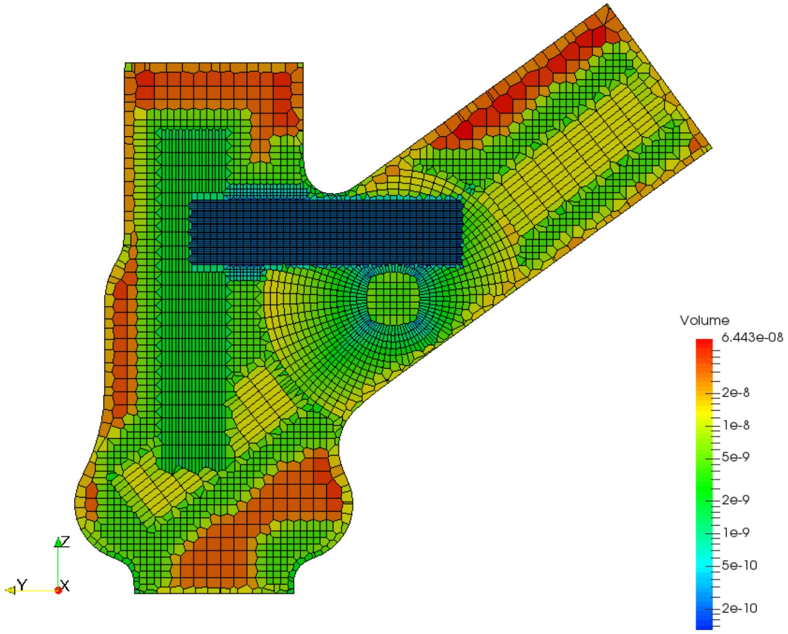


Seed points generation: special distributions with local coordinate systems









Lloyd’s smoothing

SIEMENS

Ingenuity for life

Optimize the mesh using Lloyd’s smothing to approximate a Centroidal Voronoi Tassellation (CVT)s

$$F(\mathbf{X}) = \sum_{i=1}^{n} F_i(\mathbf{X}) = \sum_{i=1}^{n} \int_{\Omega_i} \rho(\mathbf{x}) \|\mathbf{x} - \mathbf{x}_i\|^2 \, d\sigma.$$

← Energy function .
Notice the similarity with the moment of inertia

$$I_P = \iiint_Q \rho(x, y, z) \|\mathbf{r}\|^2 \mathrm{d}V$$

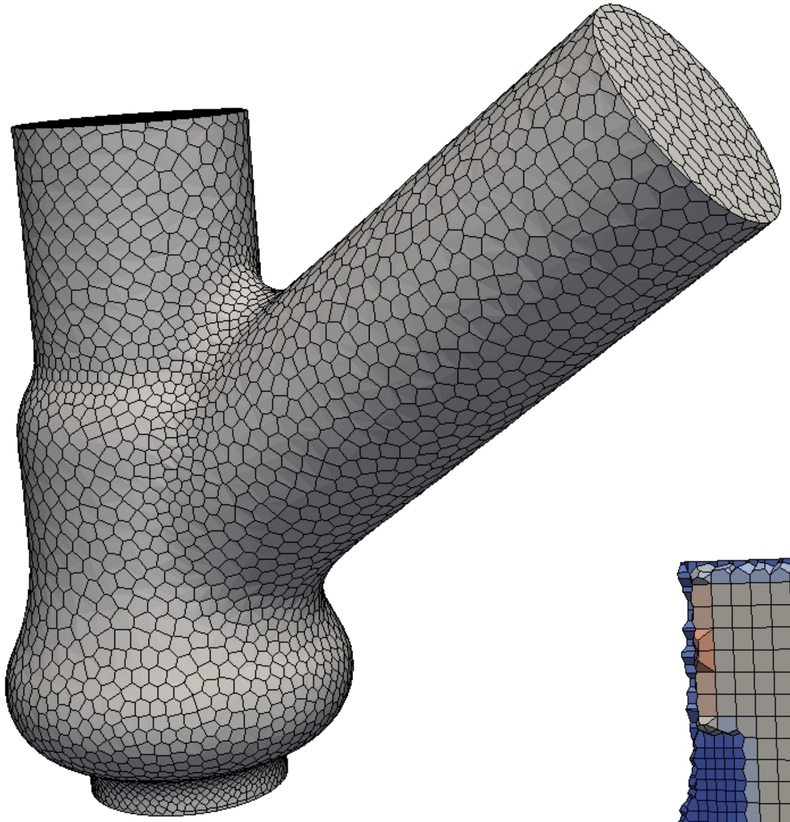
$$\frac{\partial F}{\partial \mathbf{x}_i} = 2m_i(\mathbf{x}_i - \mathbf{c}_i),$$

← this is the origin of Lloyd’s smoothing
$\mathbf{x}_i$ is the seed , $\mathbf{c}_i$ is the centroid.
Equating the derivative to 0 yelds to Lloyd’s smoothing iteration: ($\mathbf{x}_i$=$\mathbf{c}_i$)

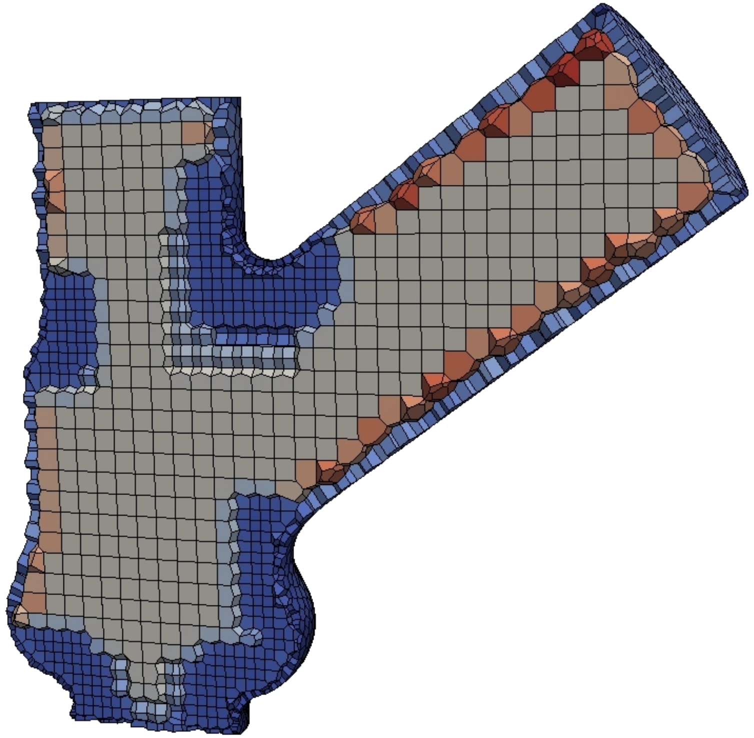
$$m_i = \int_{\mathbf{x} \in \Omega_i} \rho(\mathbf{x}) \, d\sigma$$

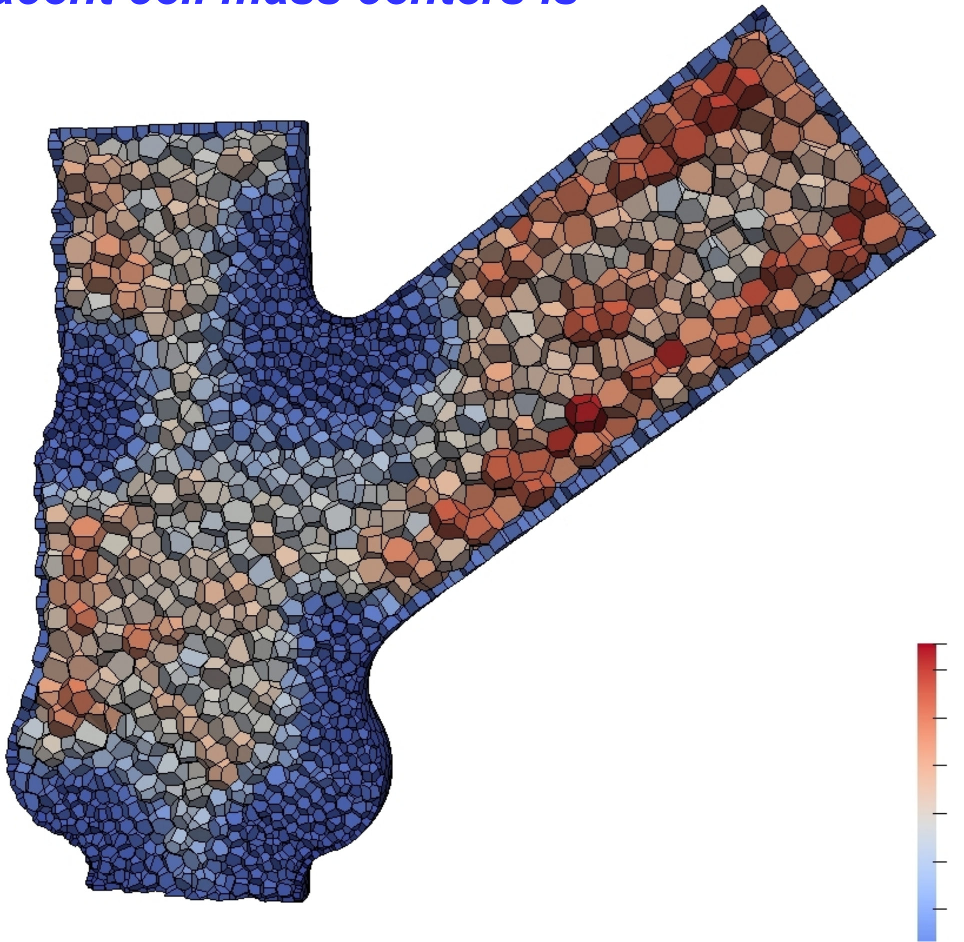
m_i is the mass of the cell i

Voronoi Mesh + Lloyd‘s smoothing



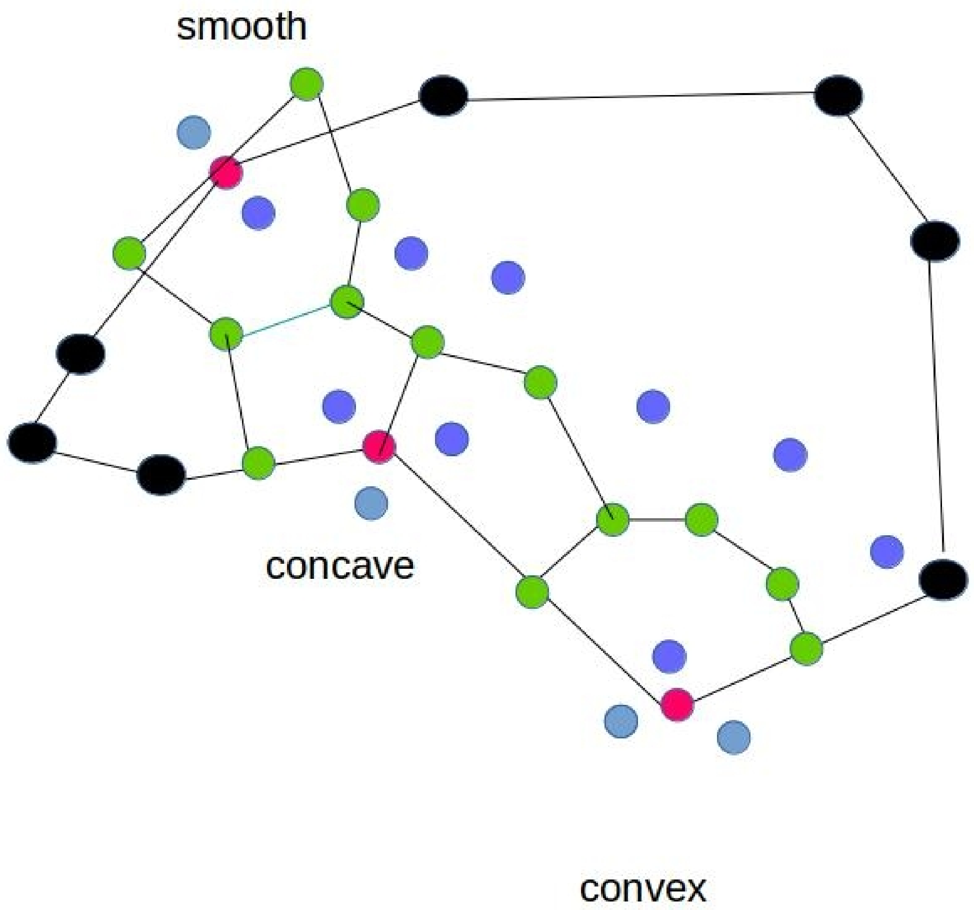
Appying the Lloyd’s smoothing at each iteration we get an update of the position of the seeds and therefore a new Voronoi mesh must be constructed. A Voronoi cell generated with a converged Lloyd’s smoothing has zero skewness, i.e. the segment connecting two adjacent cell mass centers is orthogonal to the common face.





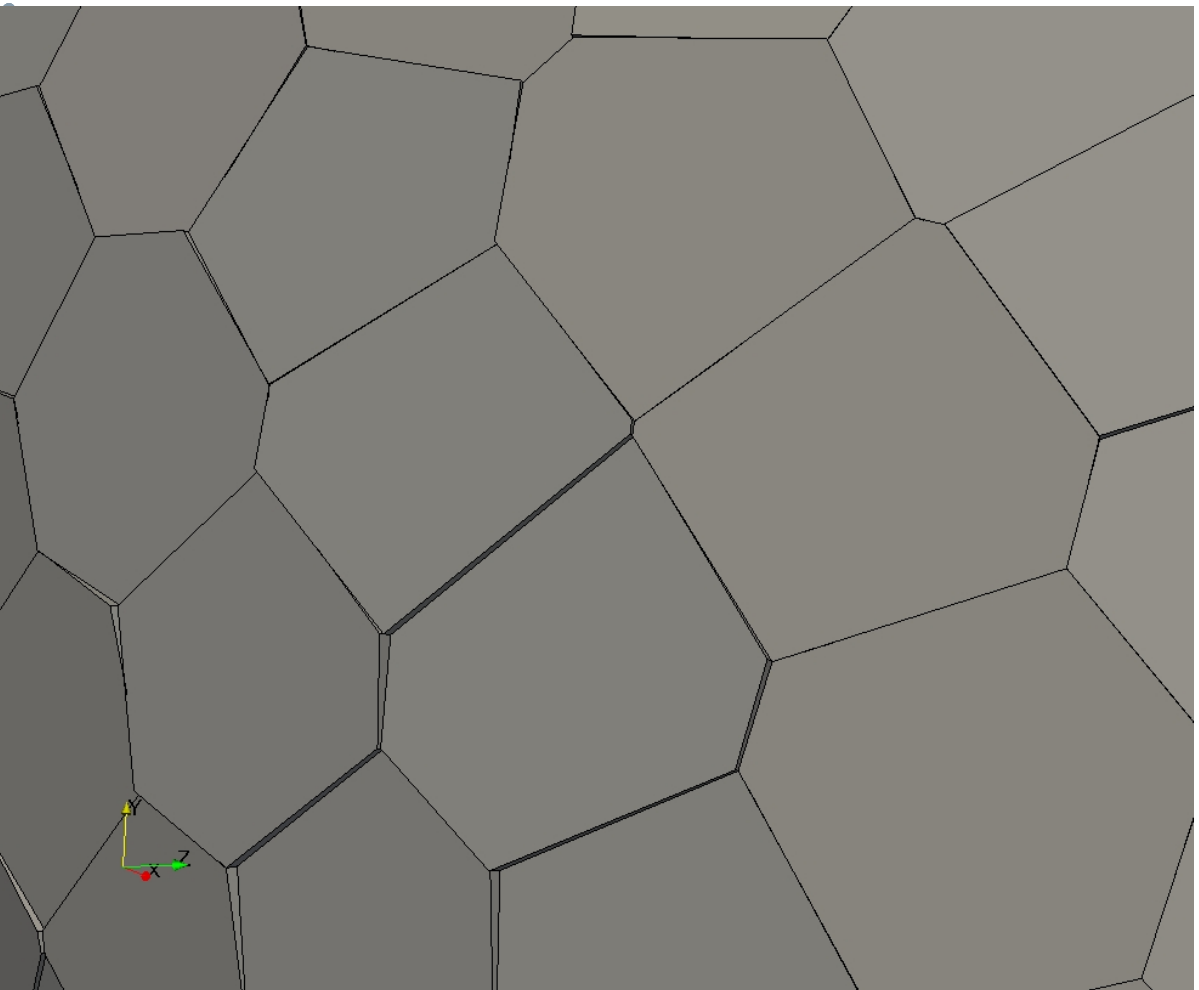
Boundary recovery without explicit clipping

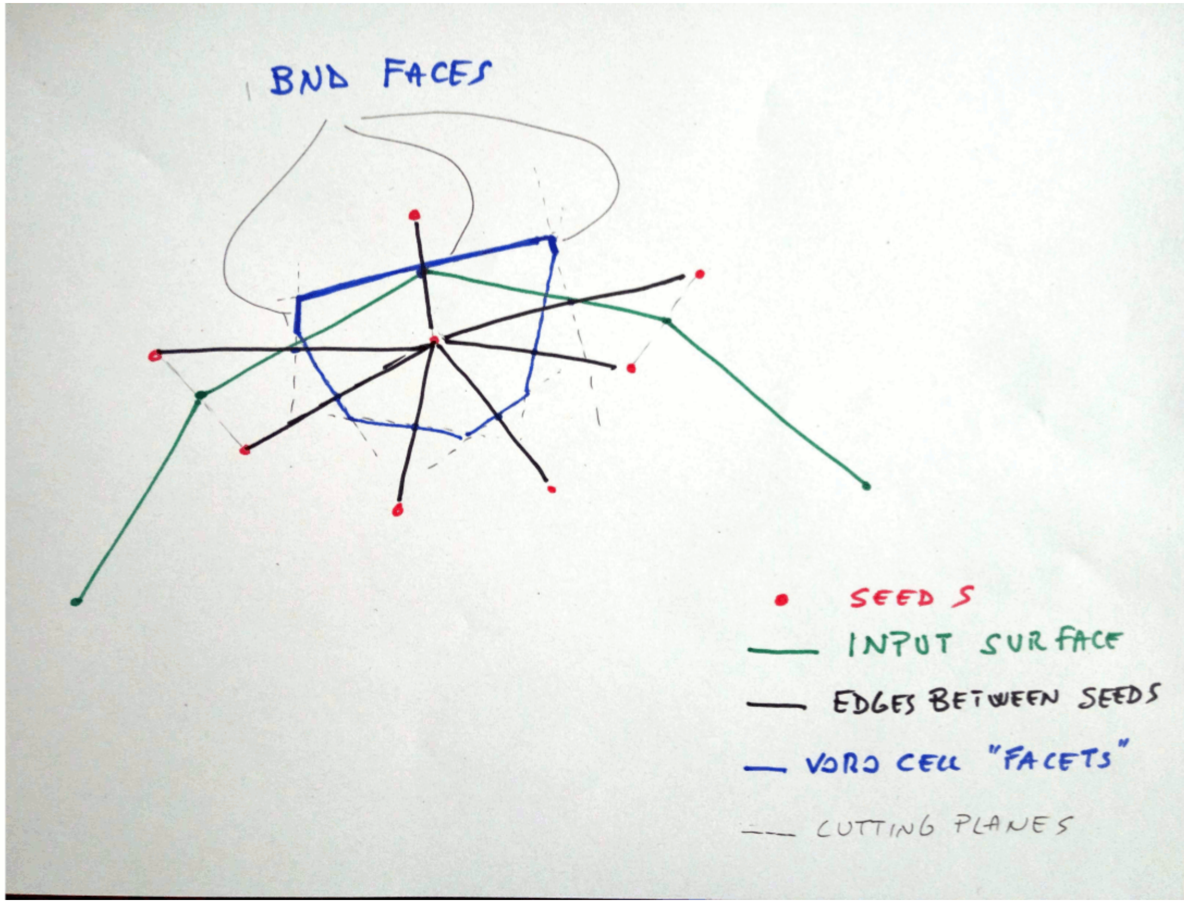
- **By placing appropriately auxiliary seeds across the boundary, Instead of the original boundary seeds, one can recover the boundary automatically without an explicit a-posteriory step.**



Problem in BND faces recovery

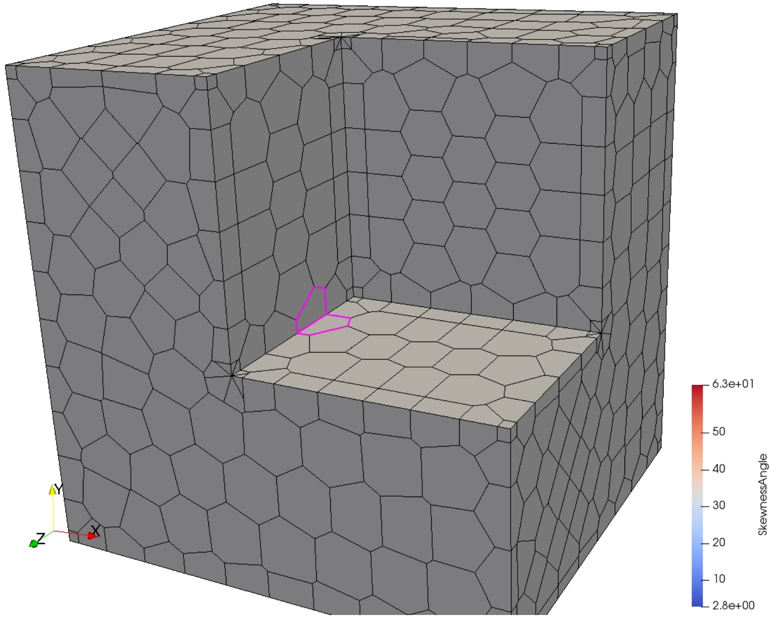
Polishing, i.e. removal of small faces and edges is required

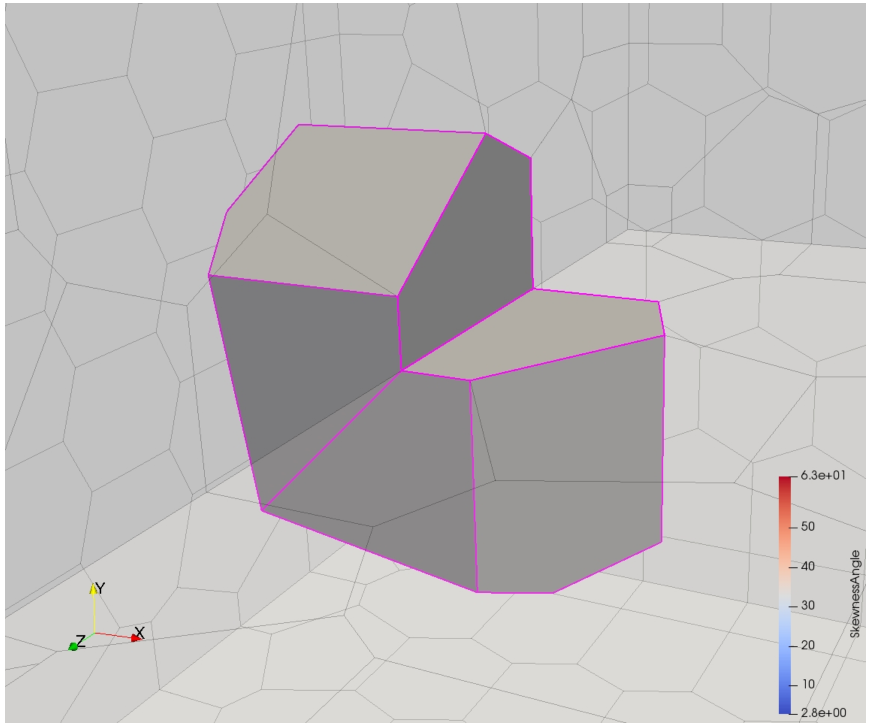


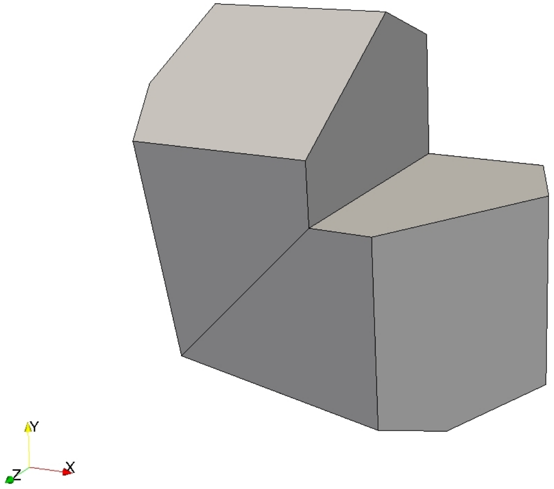


Automatic splitting of concave cells

- **The idea here is applied to a one-concave-edge cell .**



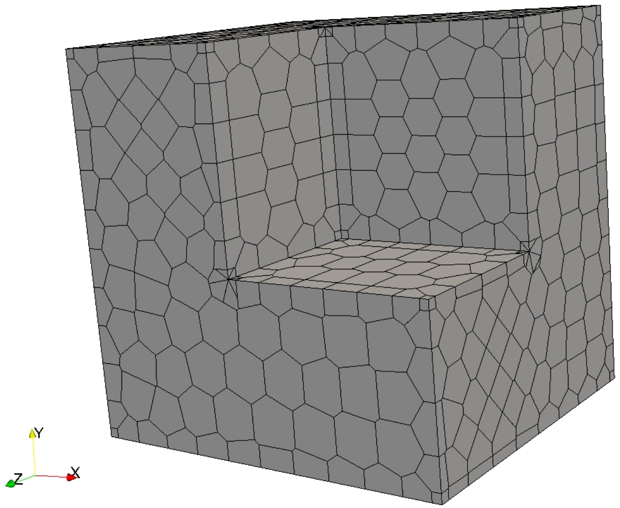


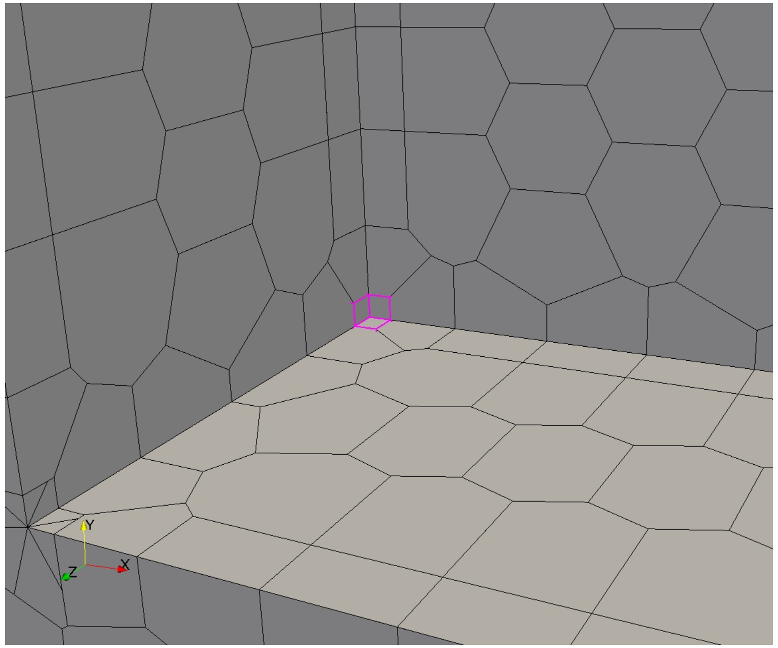


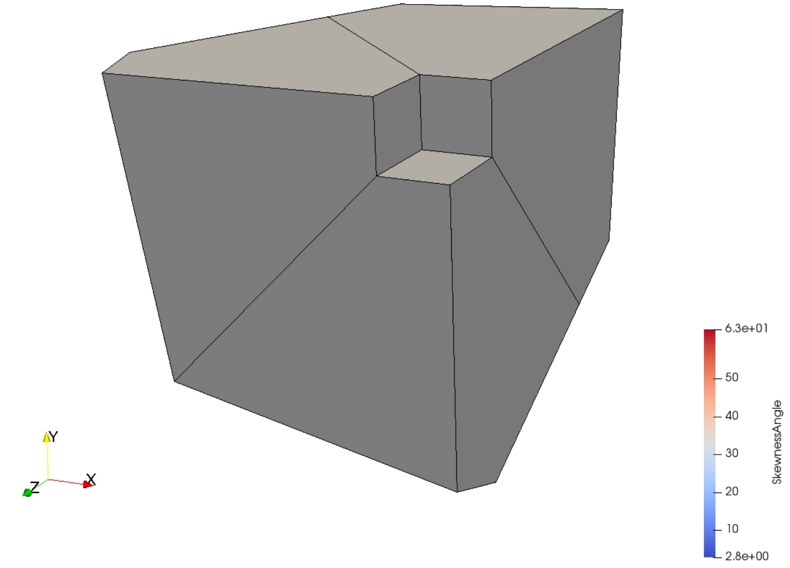
Automatic splitting of concave cells

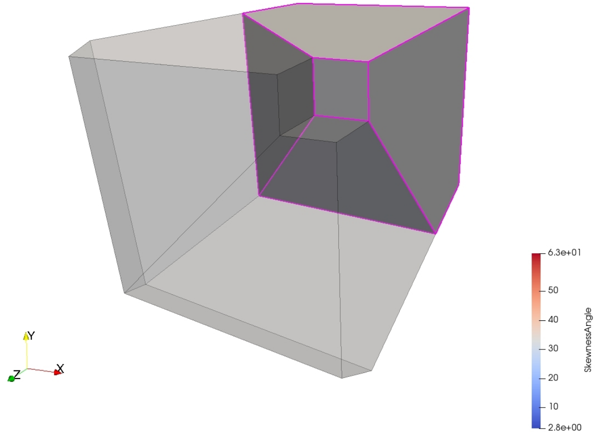
SIEMENS
Ingenuity for life

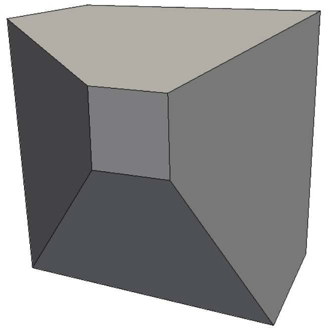
- **The idea can be extended to concave corners where multiple convex edges meet .**





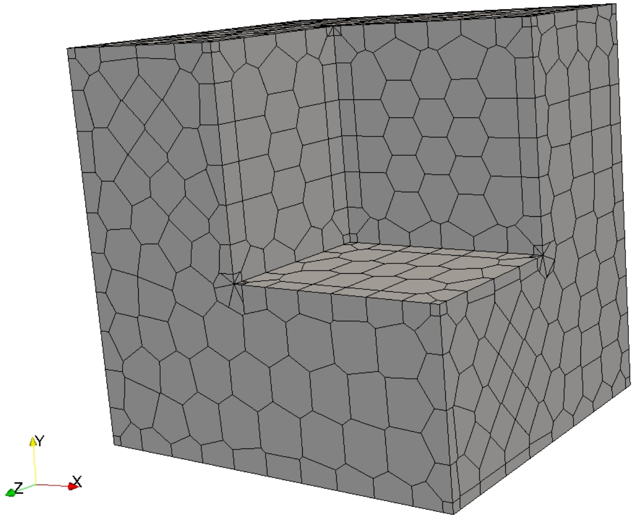


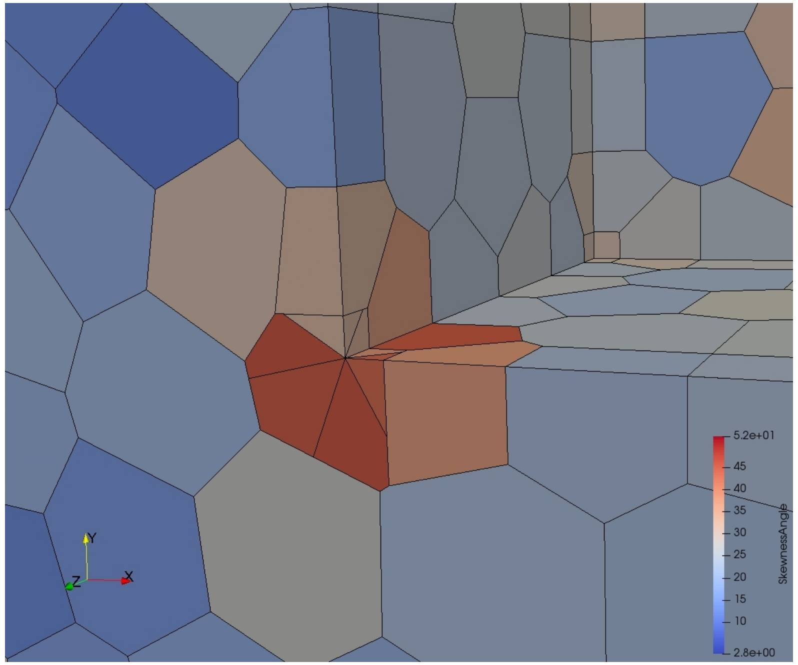


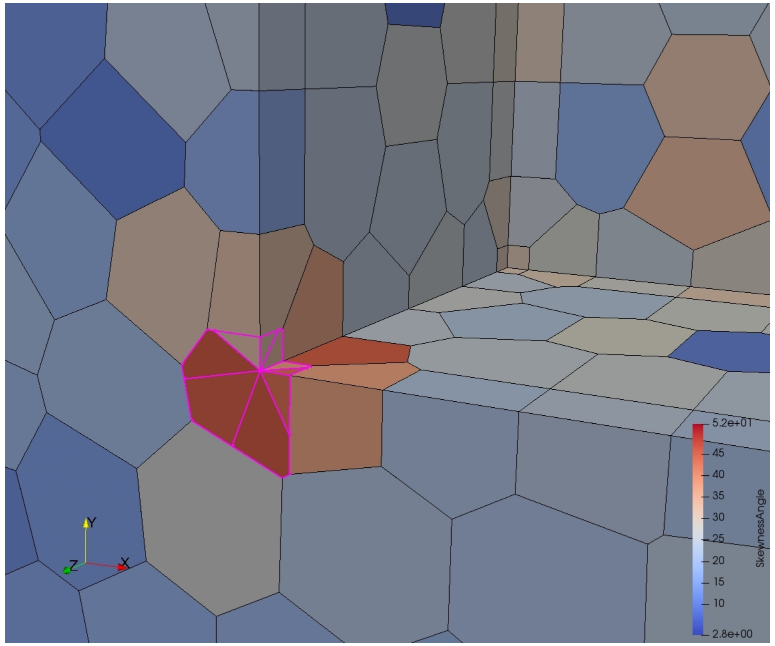


Automatic splitting of concave cells

- **The idea can also be extended to saddle-type corners where both convex and concave edges meet.**



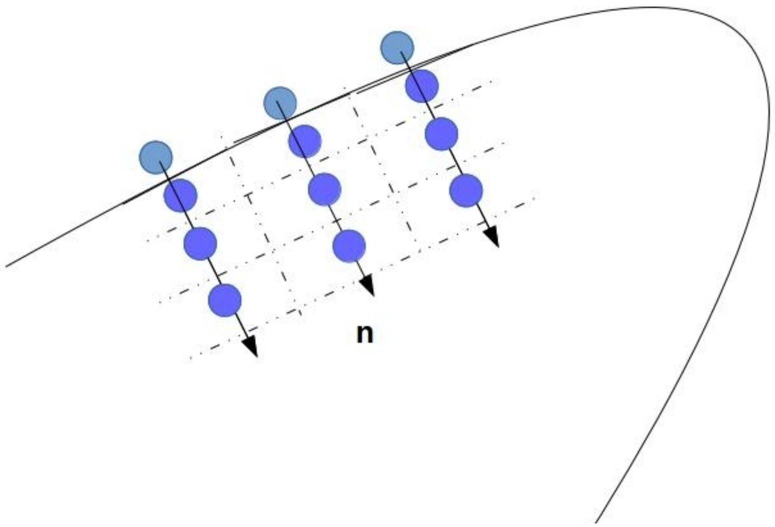


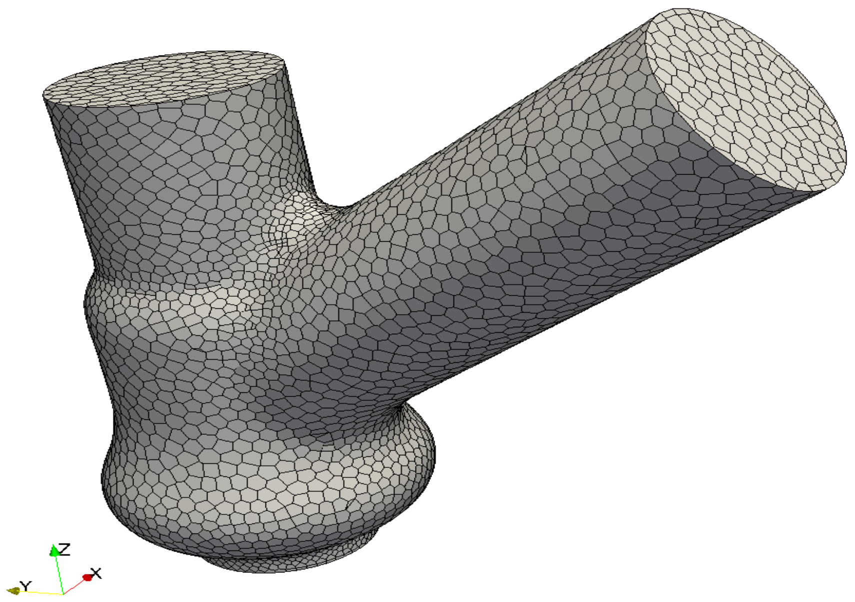


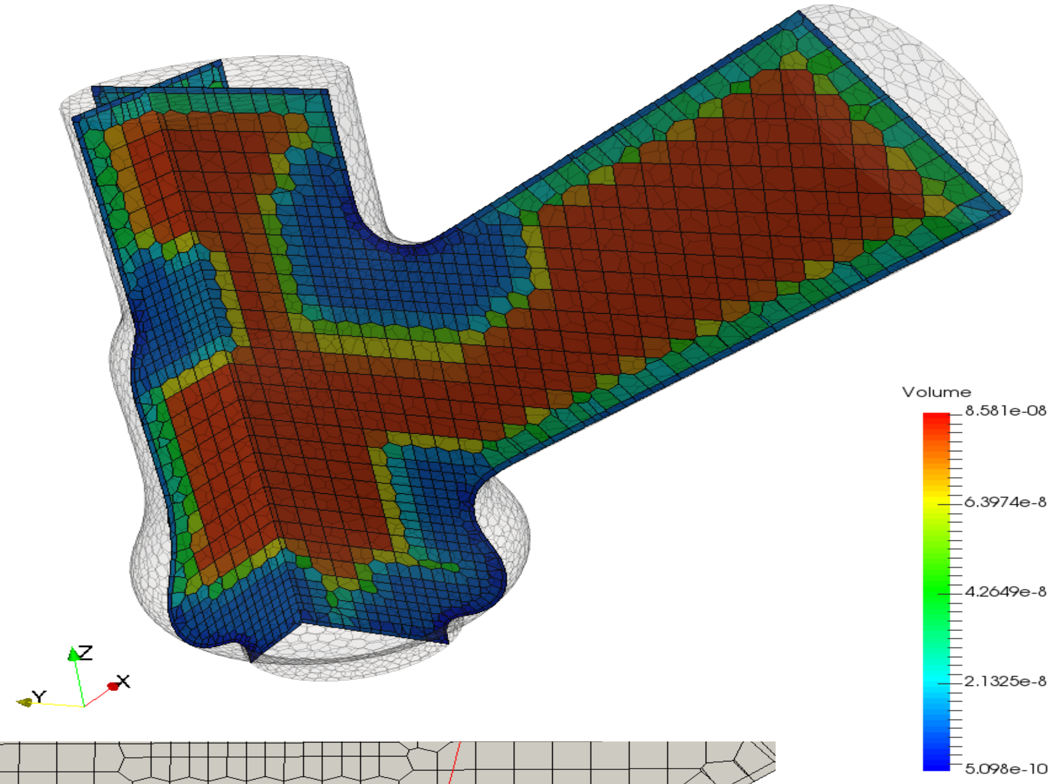
Prism layers in voronoi meshes

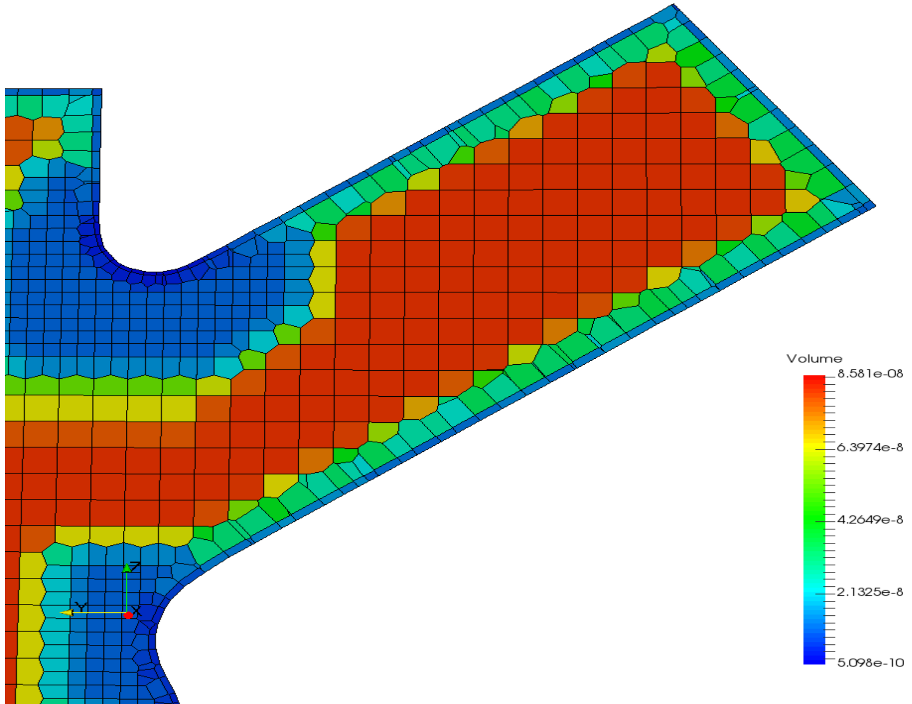
Layers of prismatic polyhedra can be created by appropriately stacking seeds along the inner normals at the vertices of the input surface.

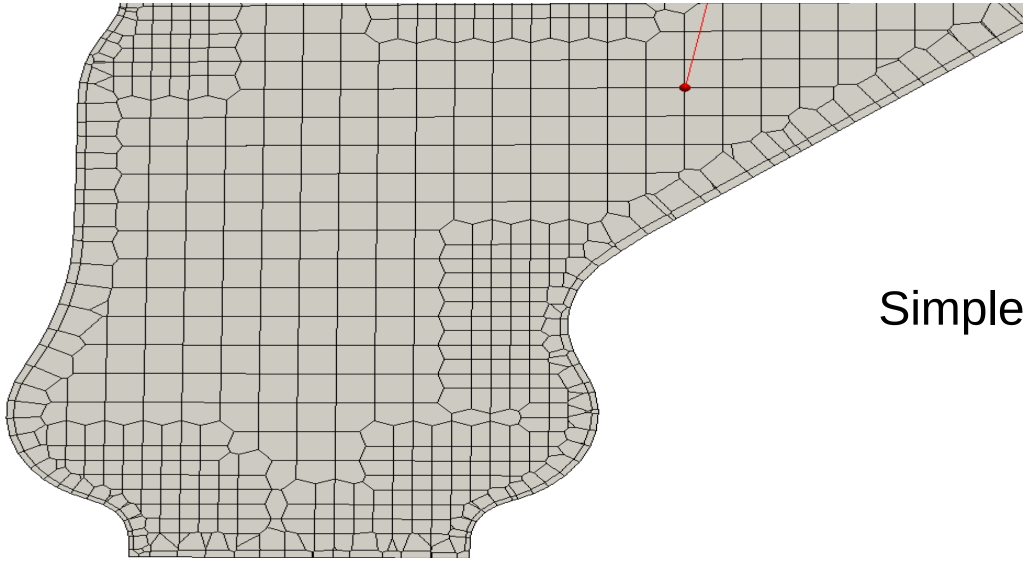
Stacks of seeds to generate prismatic layers





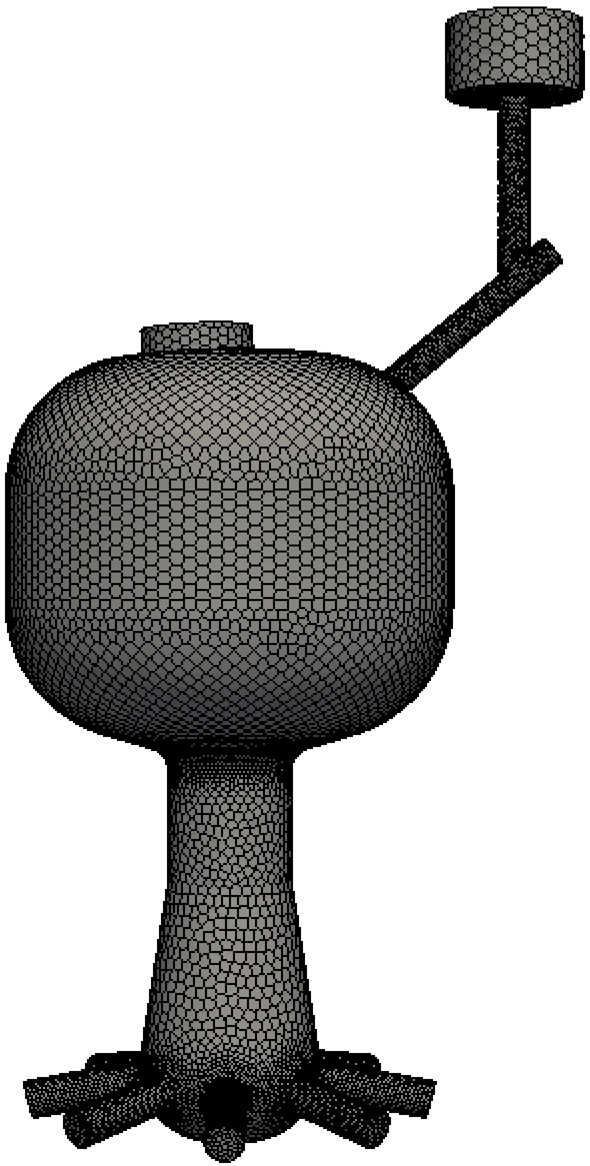


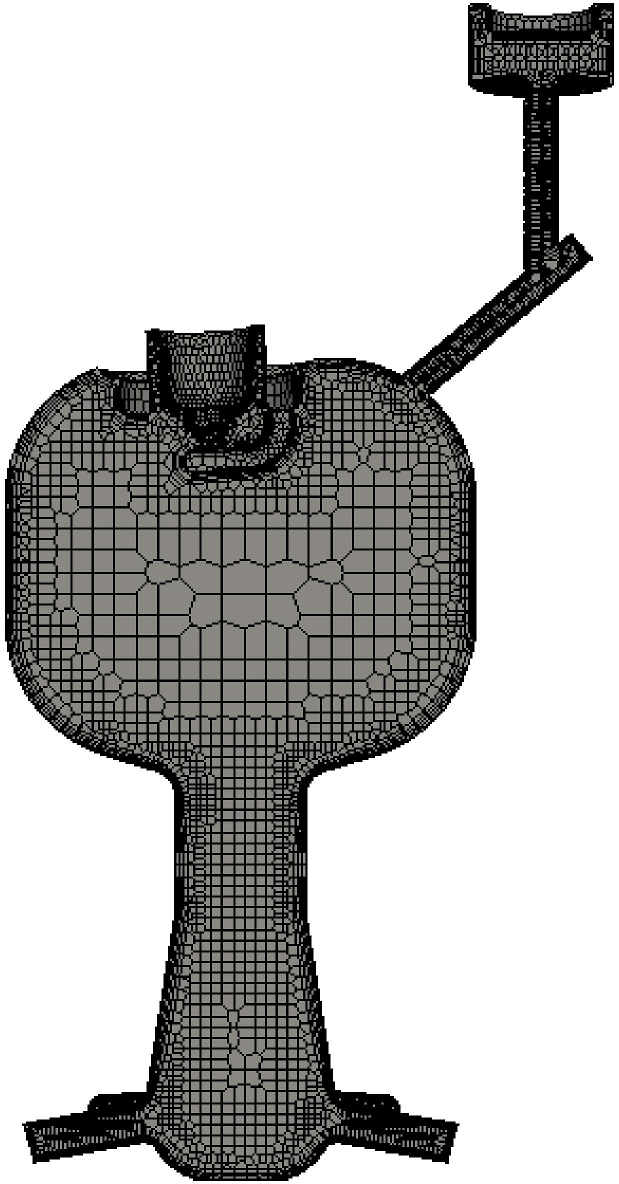




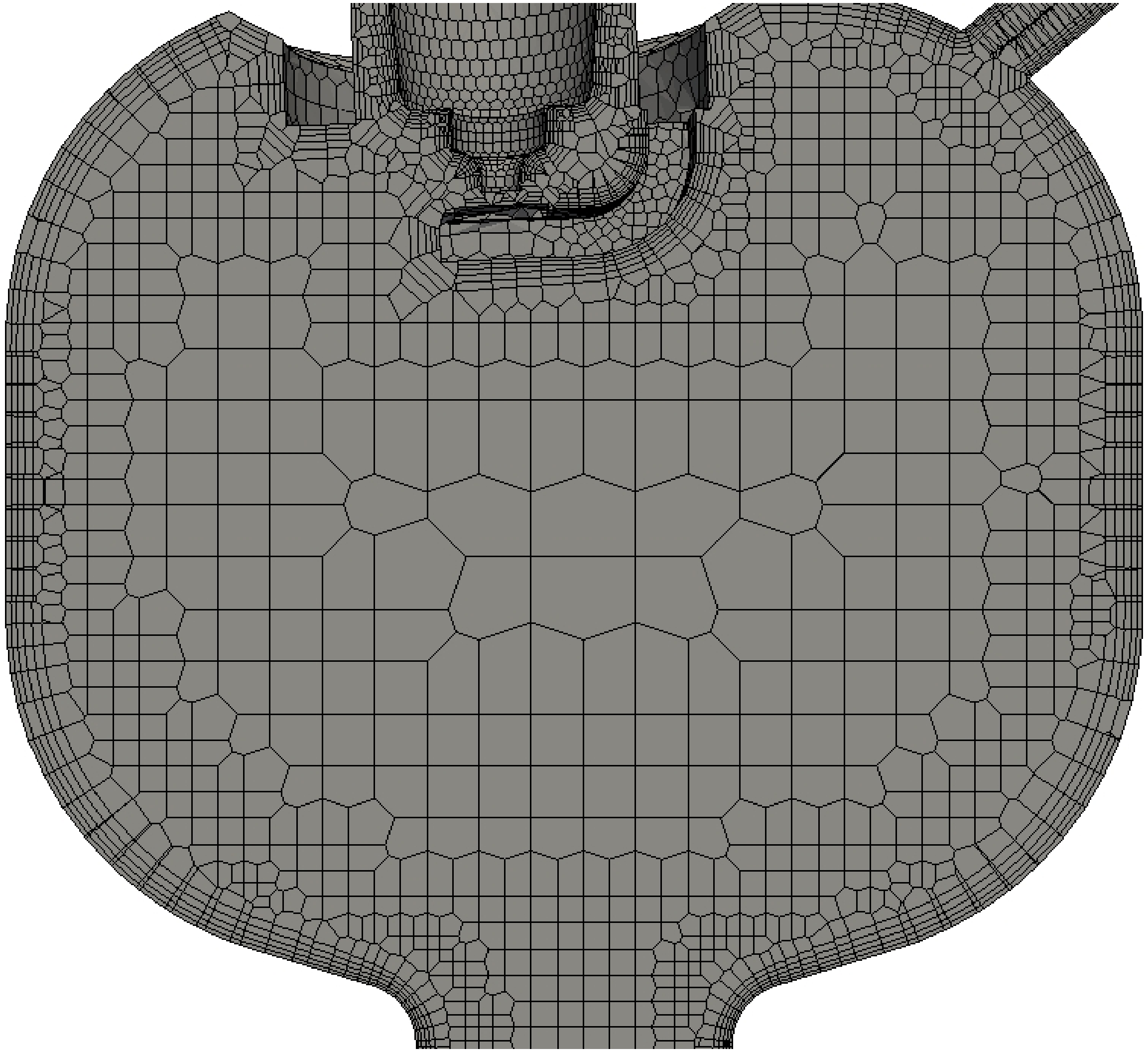
Simple prism layer

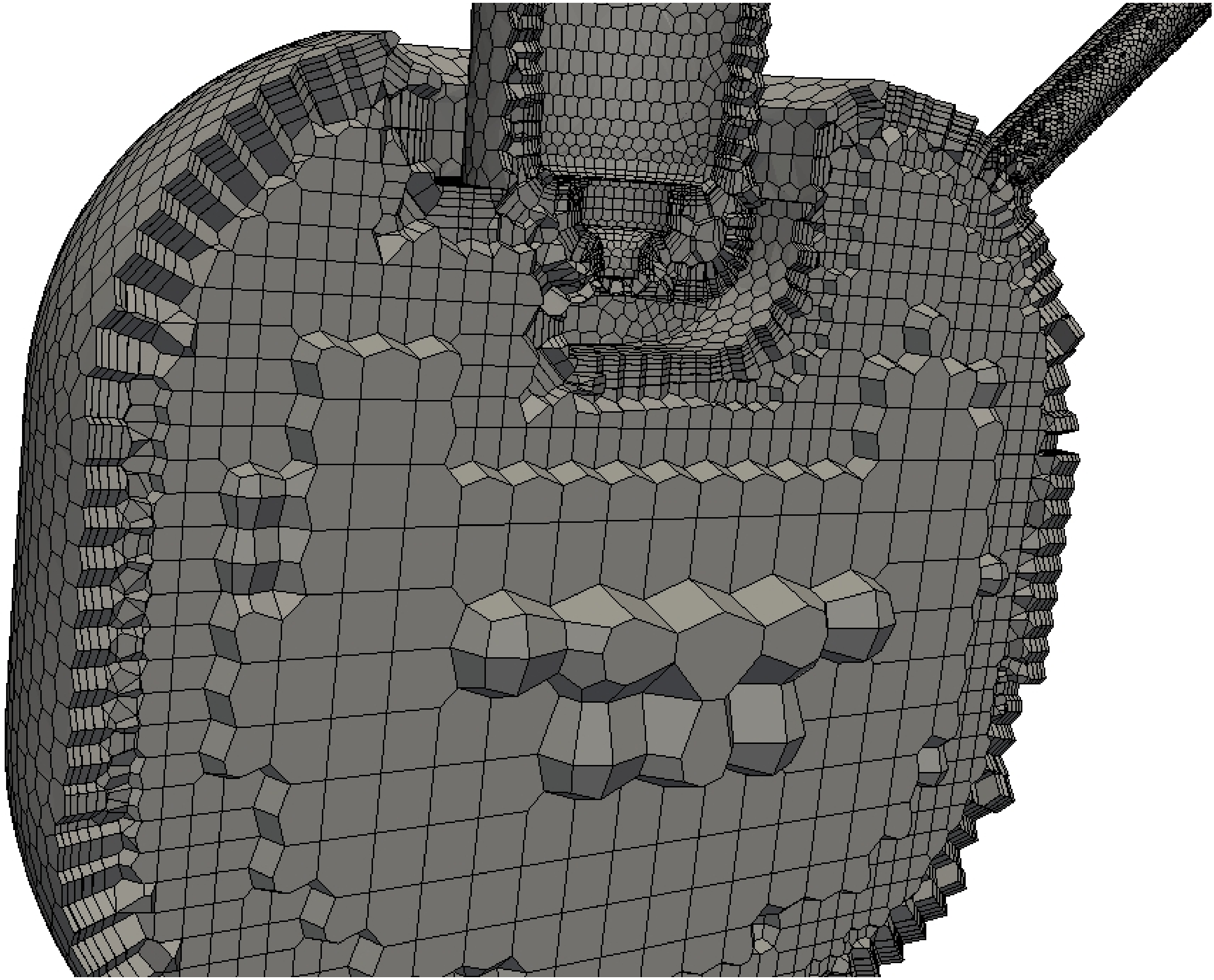
Examples of prism layers in voronoi meshes





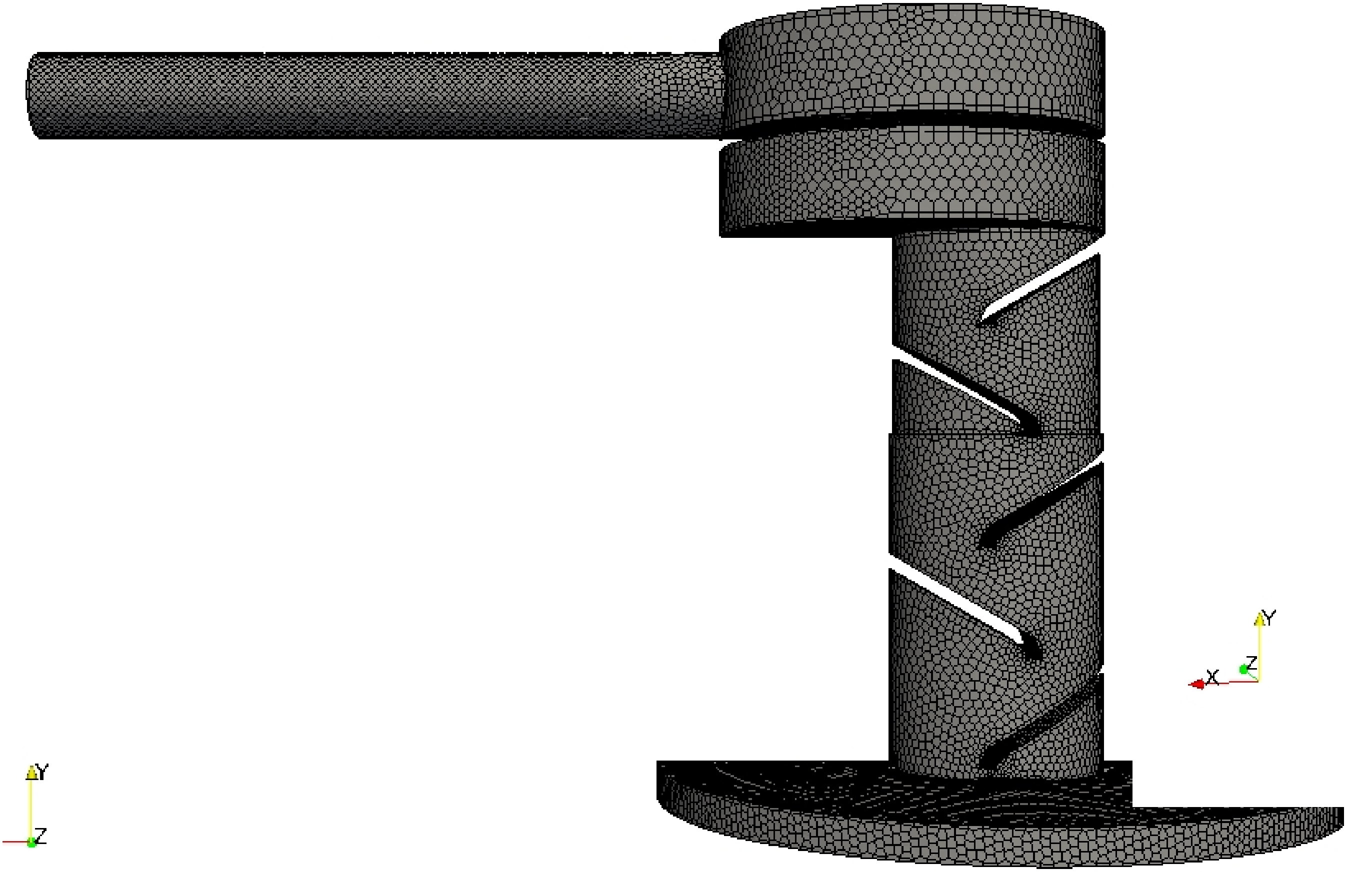
Examples of prism layers in voronoi meshes

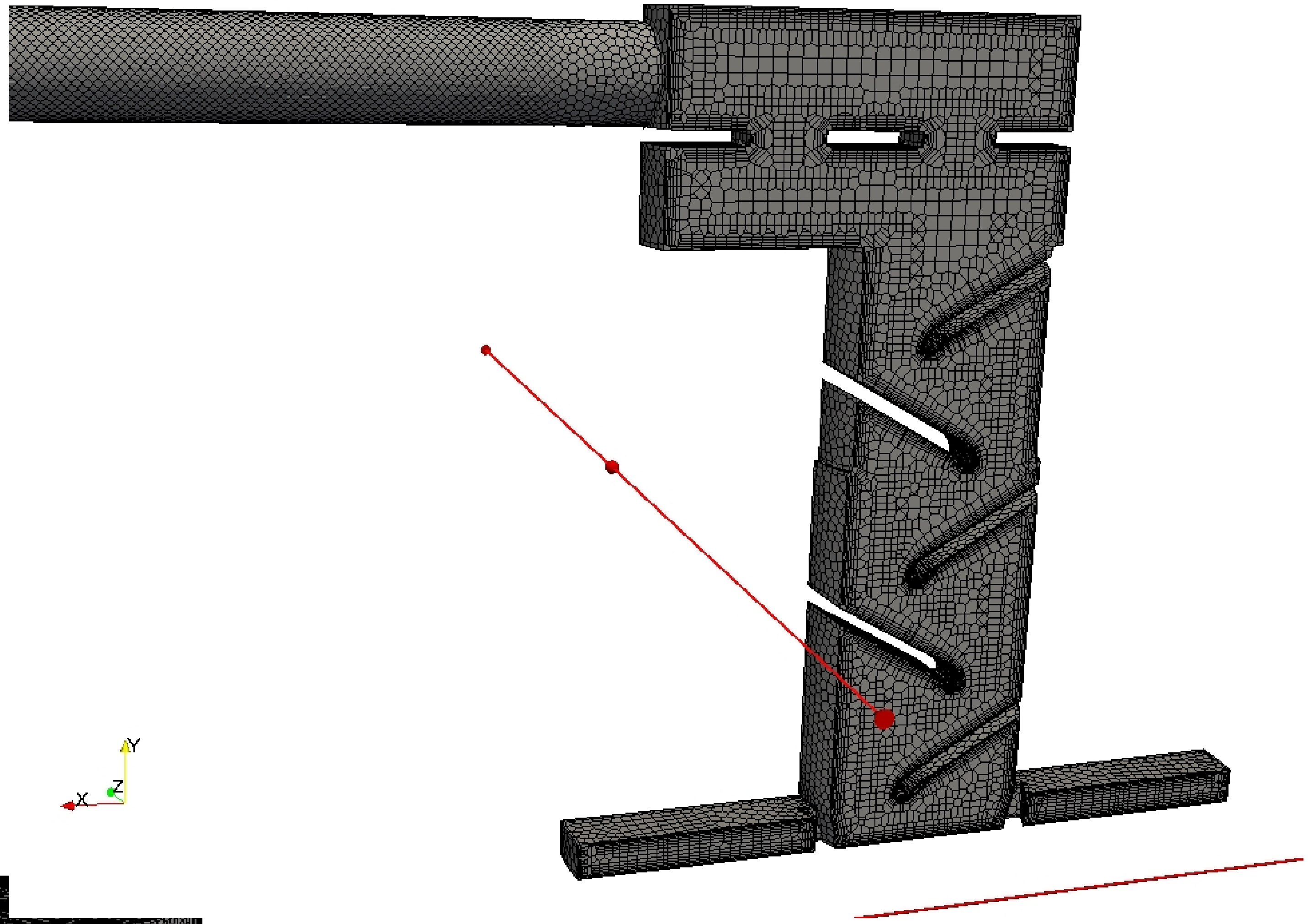




Examples of prism layers in voronoi meshes of industrial objects

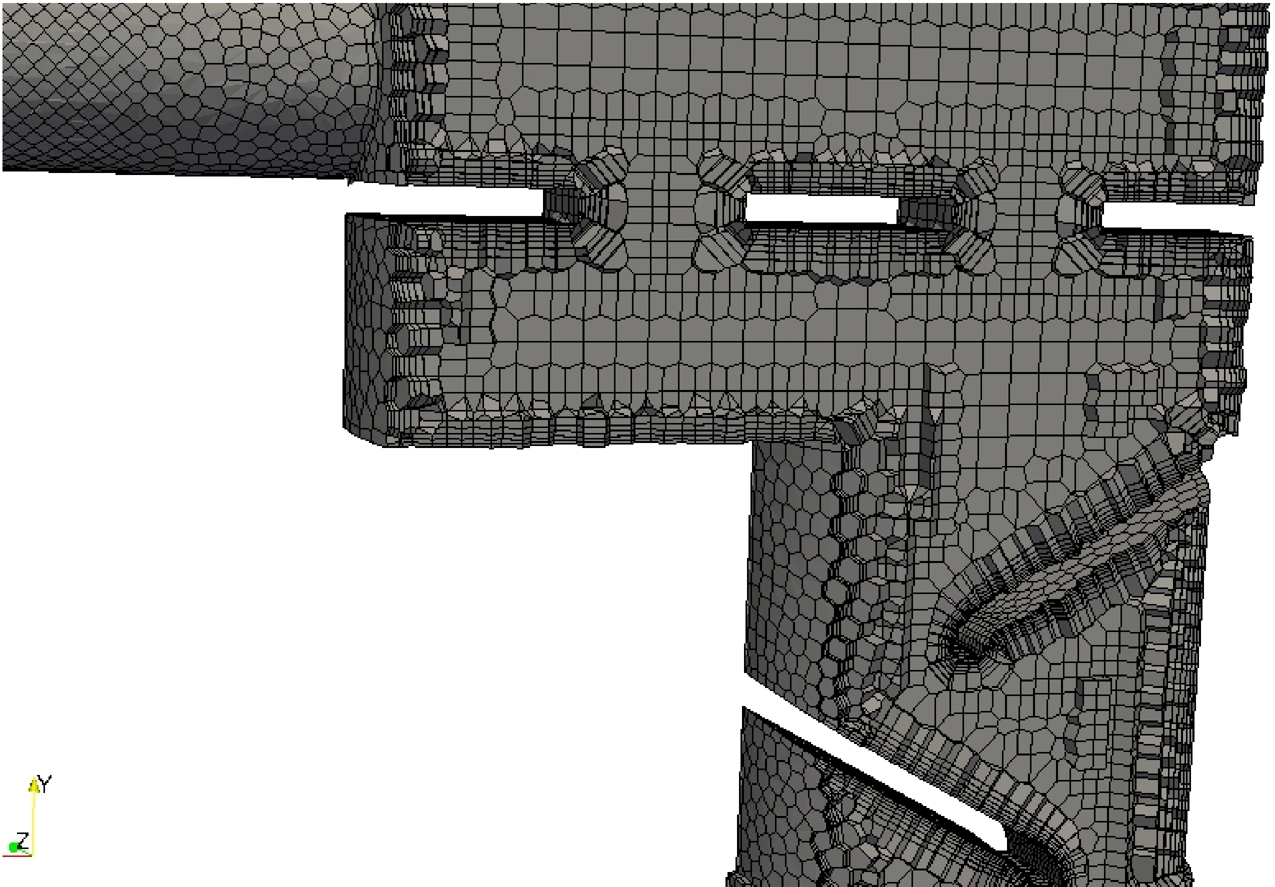
SIEMENS
Ingenuity for life

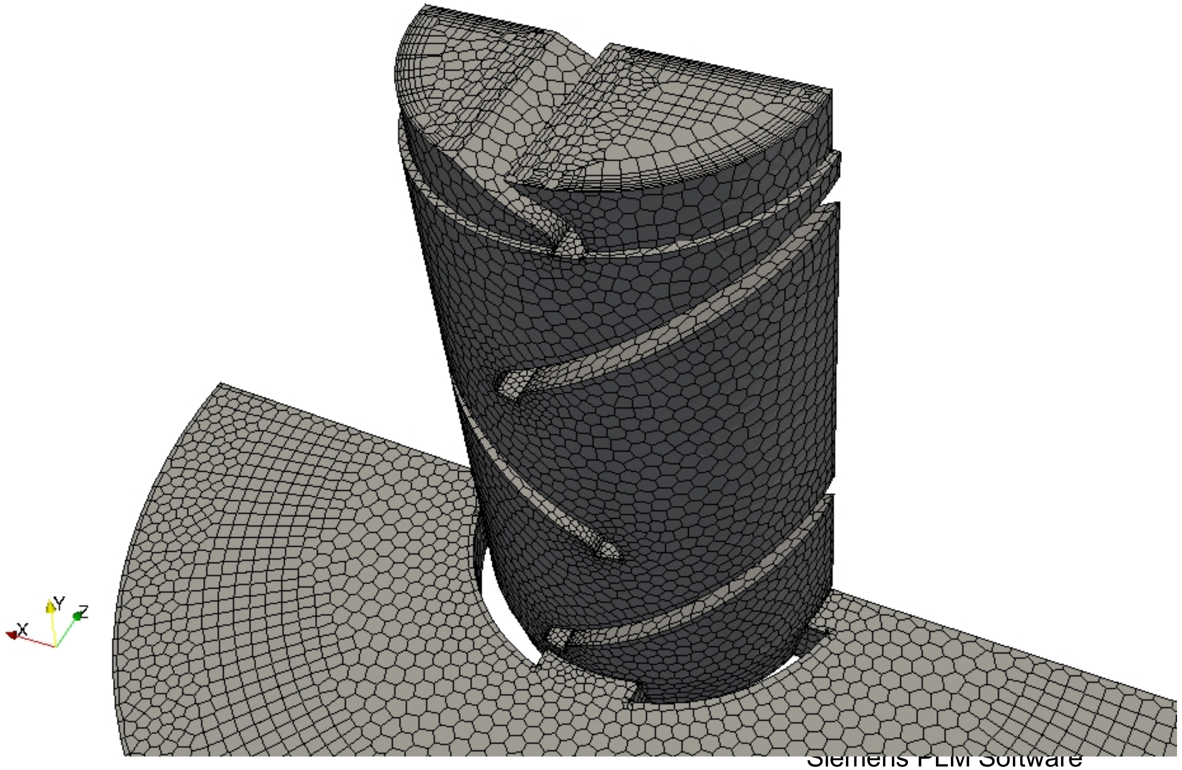




Examples of prism layers in voronoi meshes of industrial objects

SIEMENS
Ingenuity for life





Parallel generation with the cuboid method

SIEMENS
Ingenuity for life

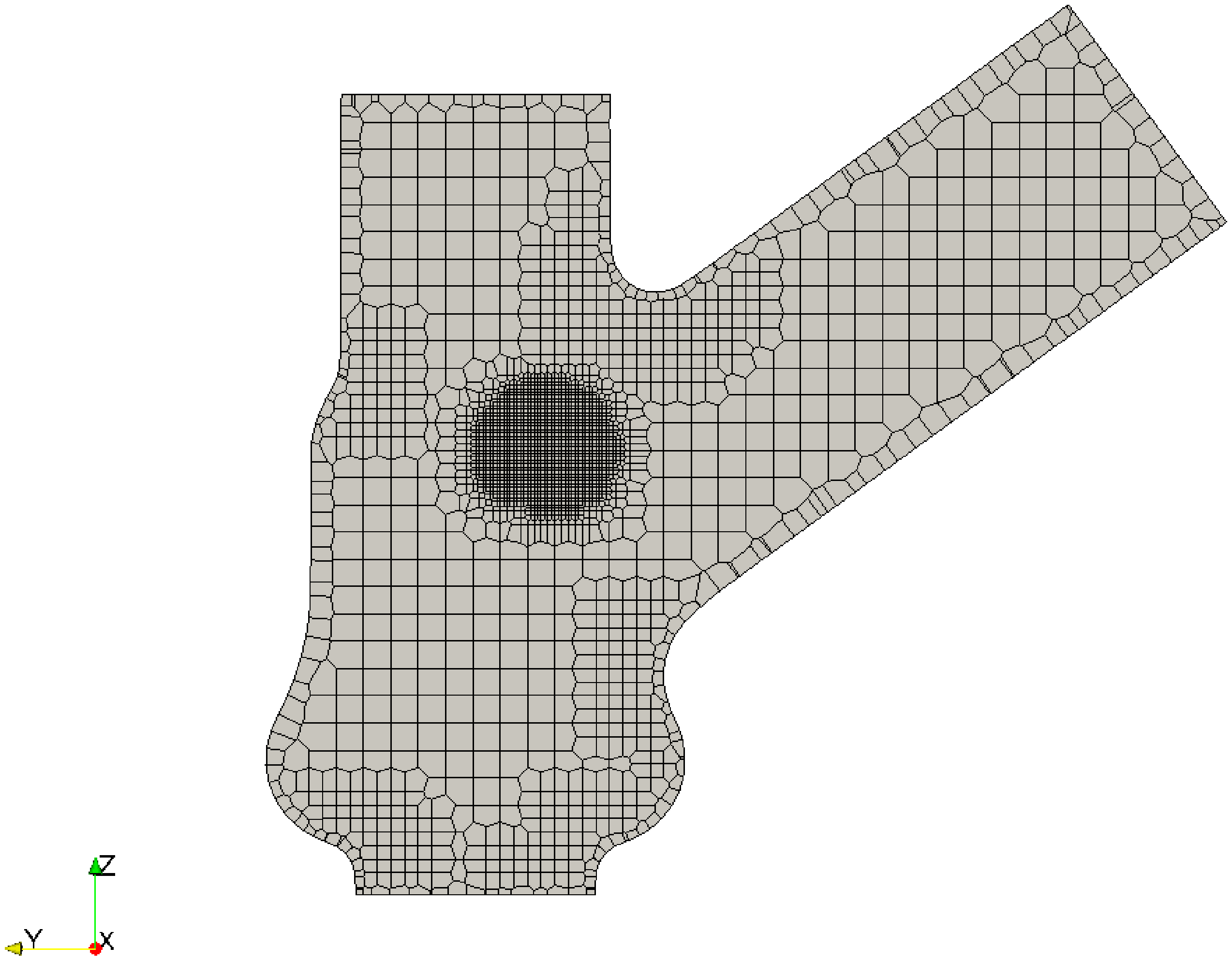
- The cutting and updating of the cuboids around each of the vertices is an "*embarrassingly parallel*" operation. Basically each cuboid update is indepent from the others. Therefore this part of the generation can be parallelized either by threads or by partitioning the seed set.
- However, the insertion of each Voronoi cell in the polyhedral mesh needs to be planned carefully and appropriate locks may be necessary to avoid threads contention when writing the cell description.

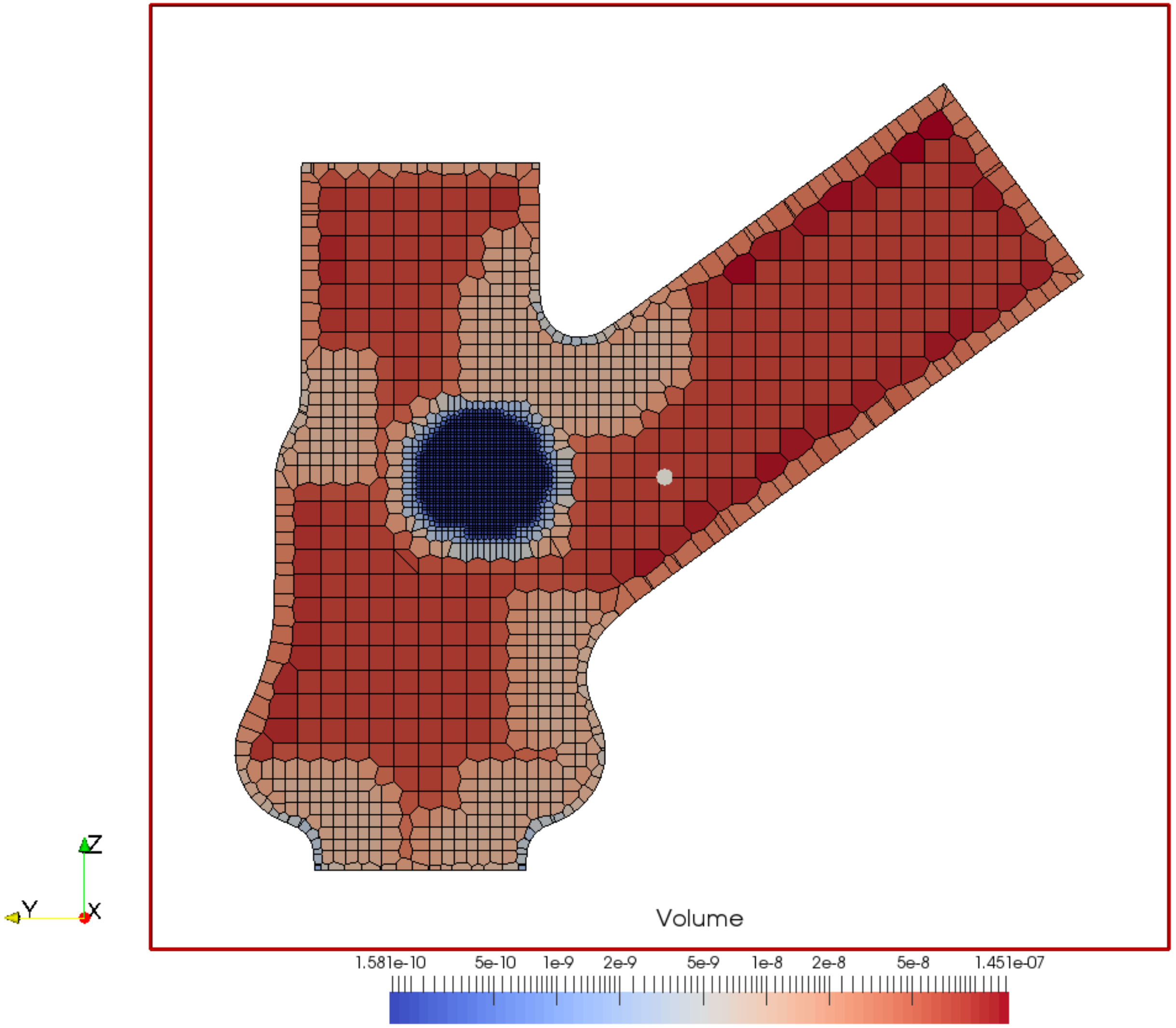
Refinement in voronoi meshes

Moving the focus from creating/removing cells to creating/removing seeds.

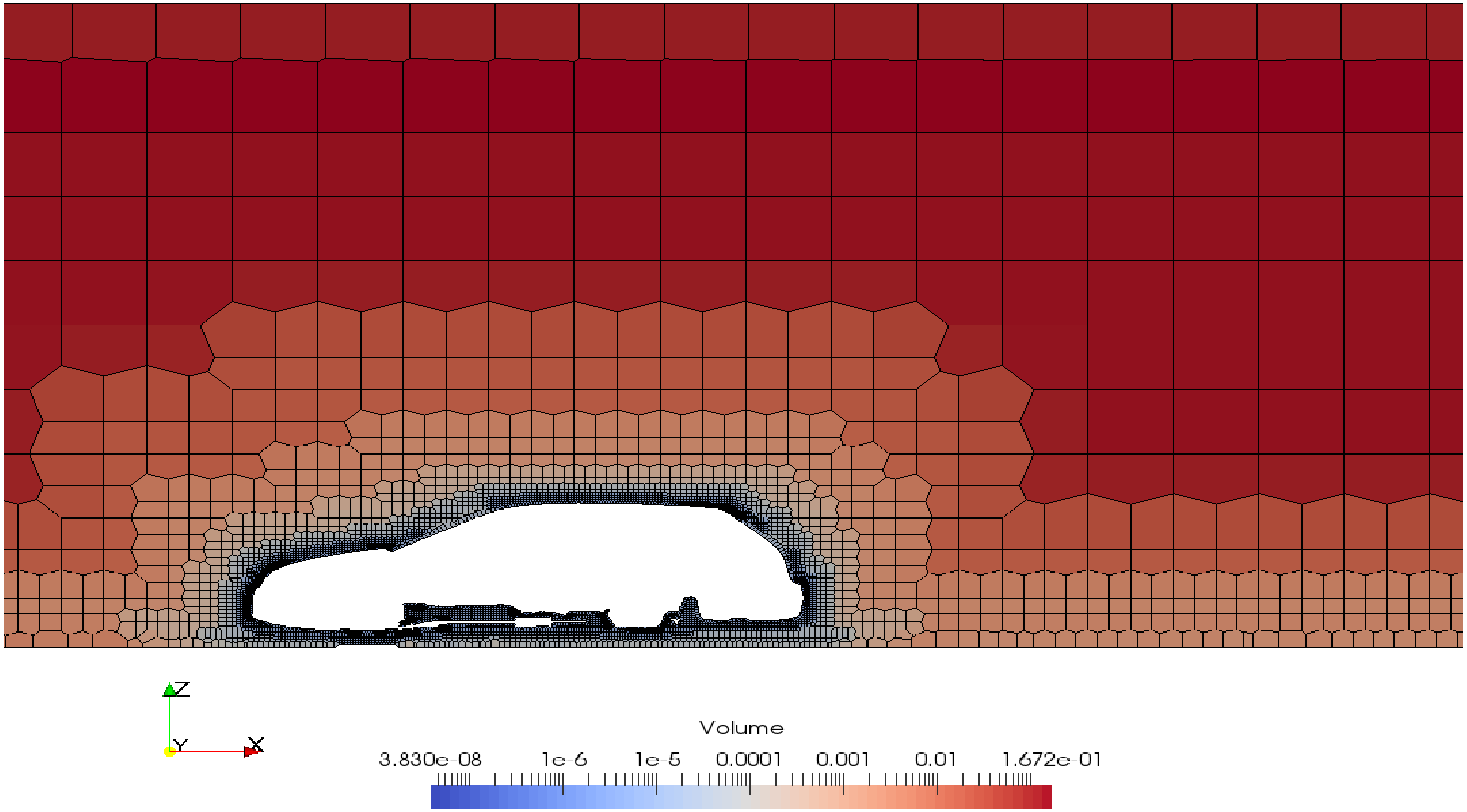
- *Simple Idea: remove a cell (actually its seed) and replace it with seeds half way through the initial seed location and the cell vertices. Finally reconstruct the voronoi cells for the newly inserted seeds and their neighbors.*
- *For octree initial mesh seed distribution this is equivalent to standard hex-style refinement.*

Refinement in voronoi meshes

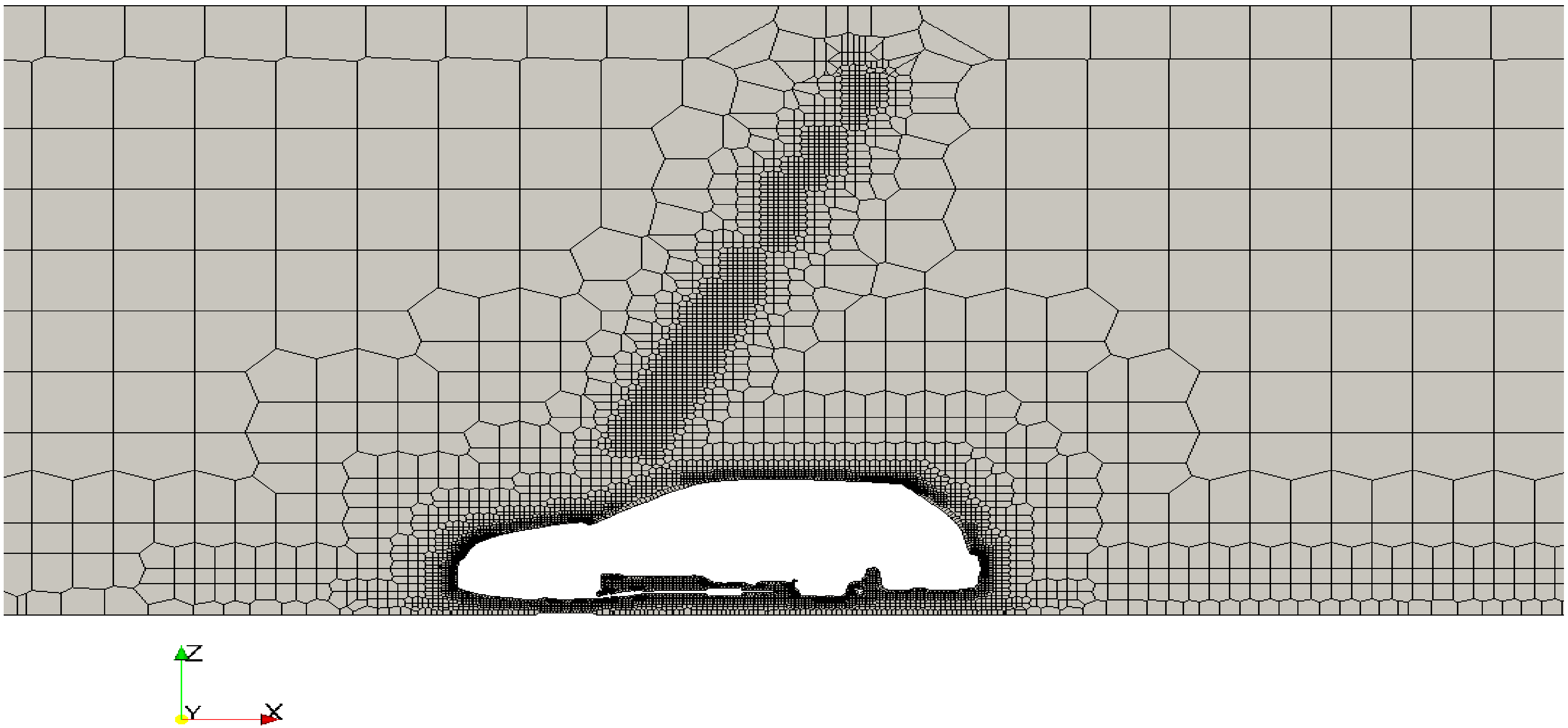




Refining a voronoi mesh

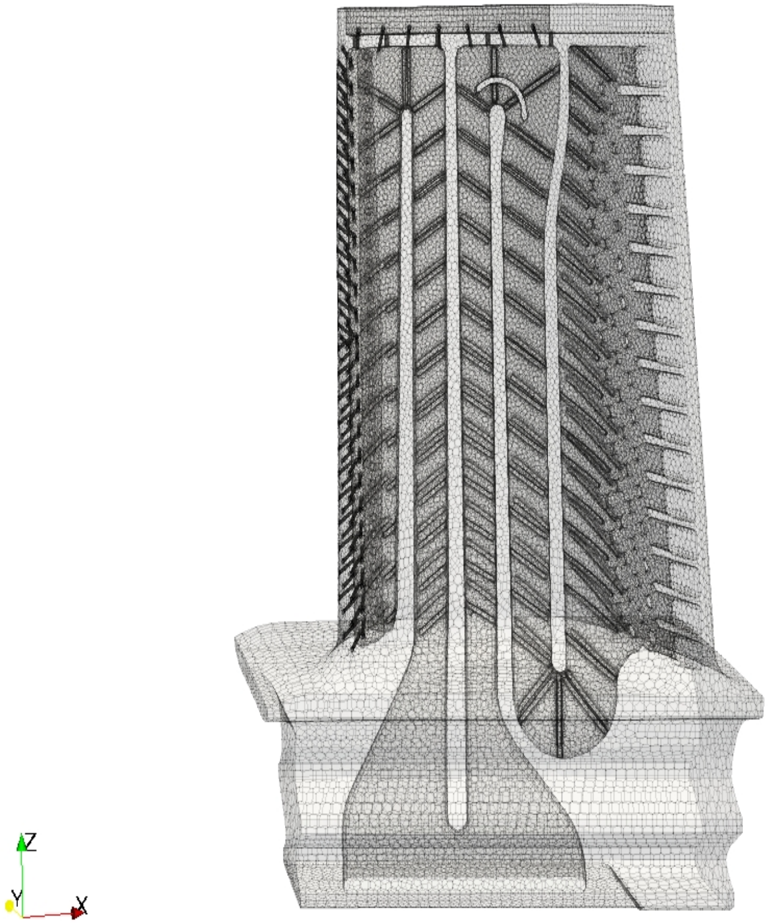


Refining a voronoi mesh

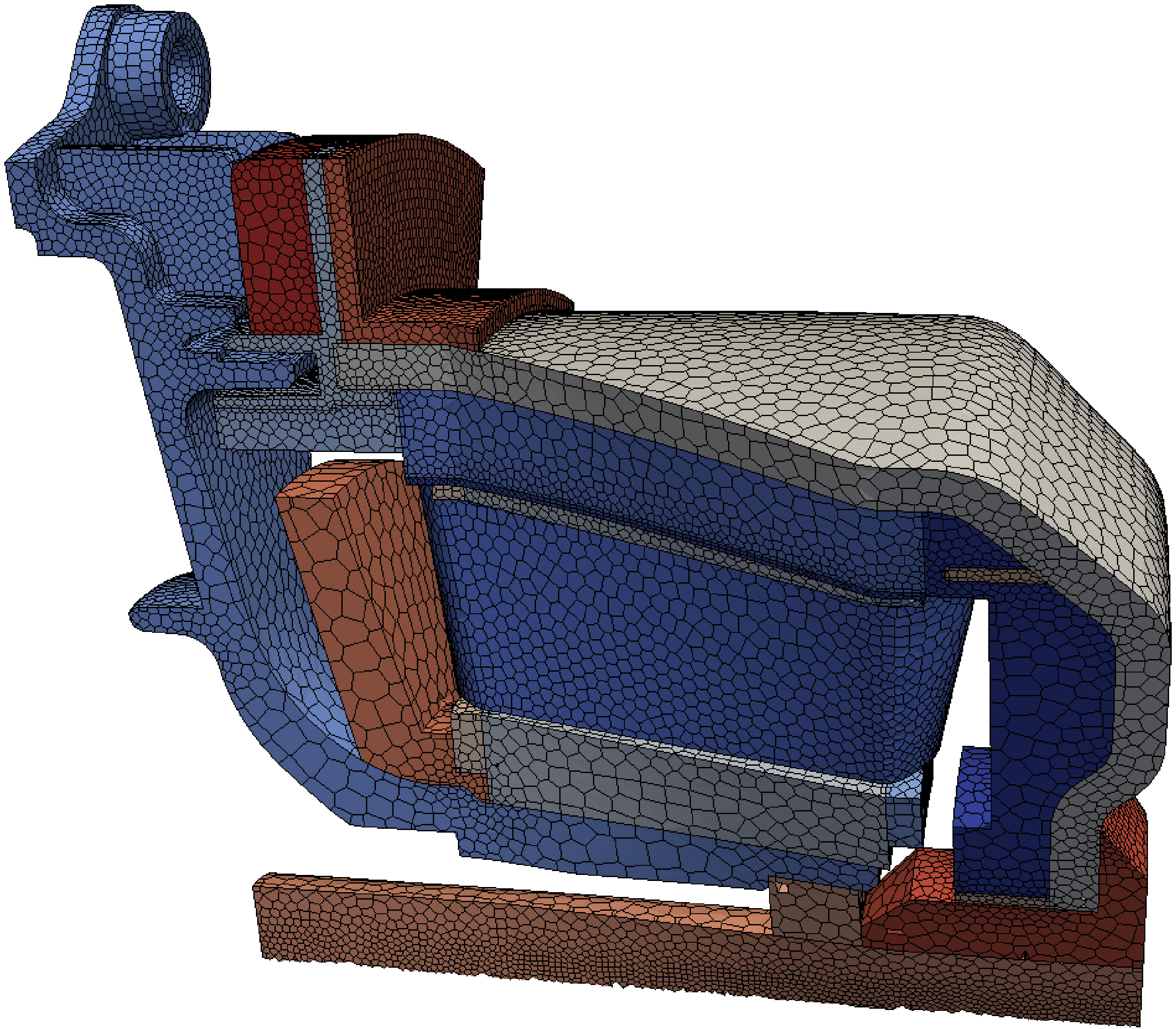


Examples of voronoi meshes of industrial objects

SIEMENS
Ingenuity for life

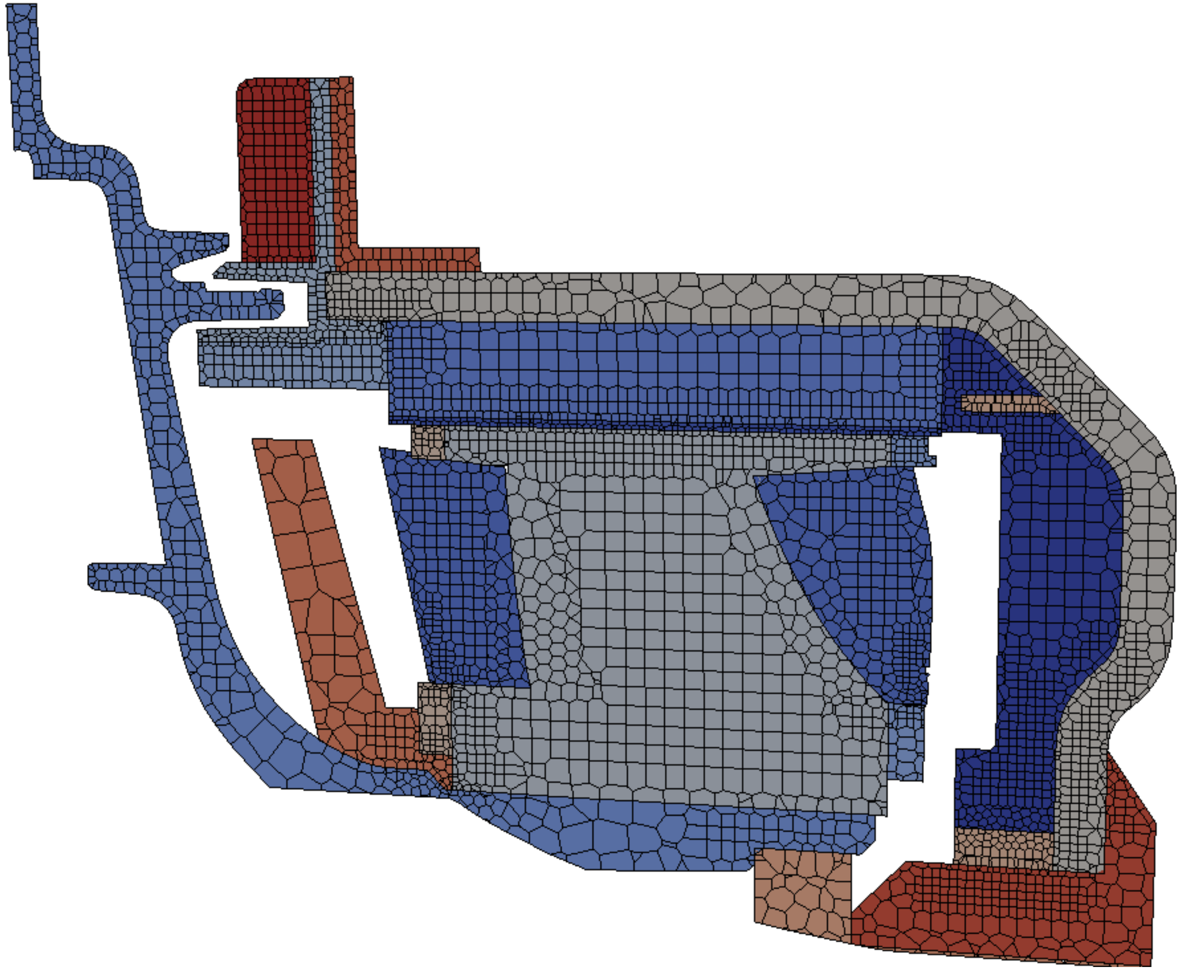


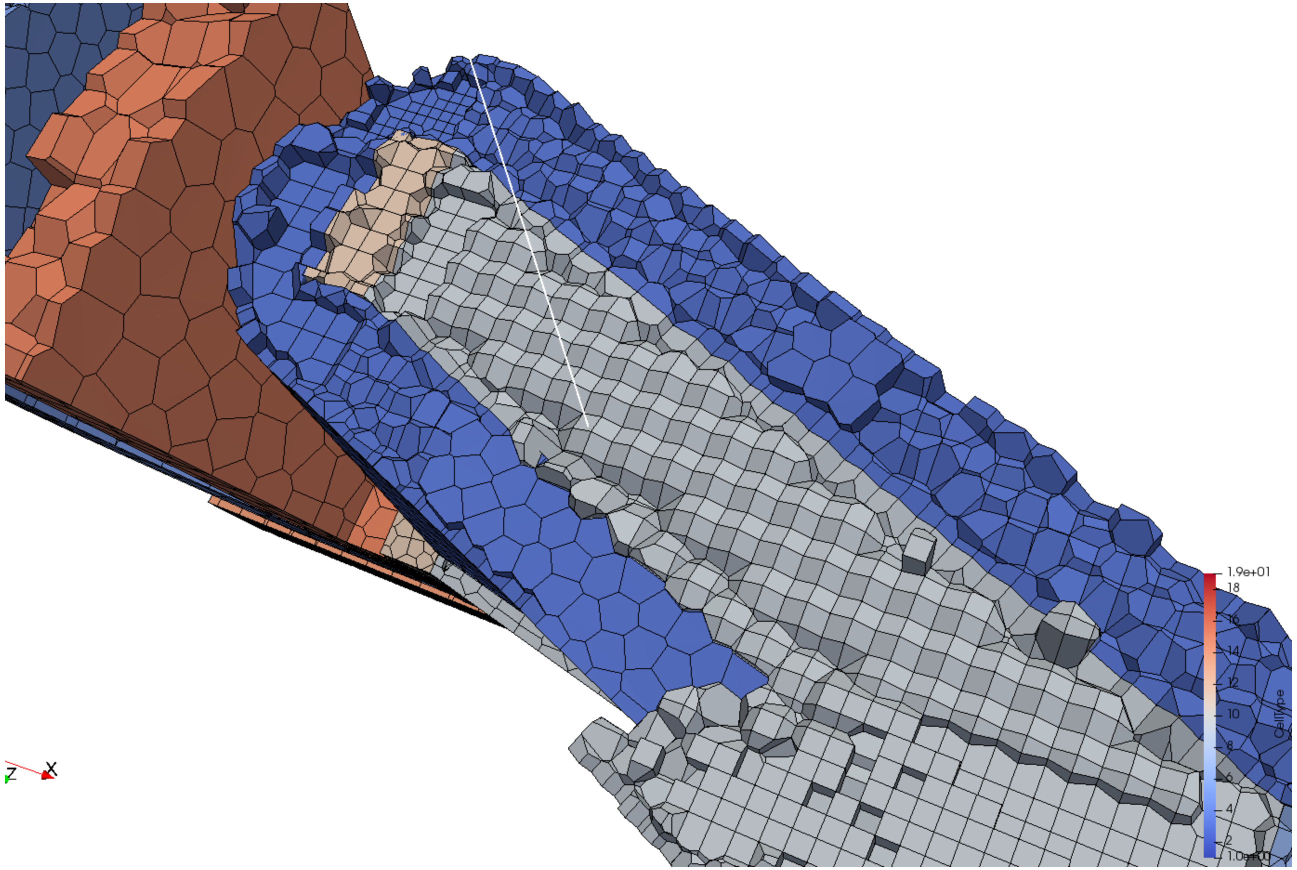
Examples of voronoi meshes of industrial objects



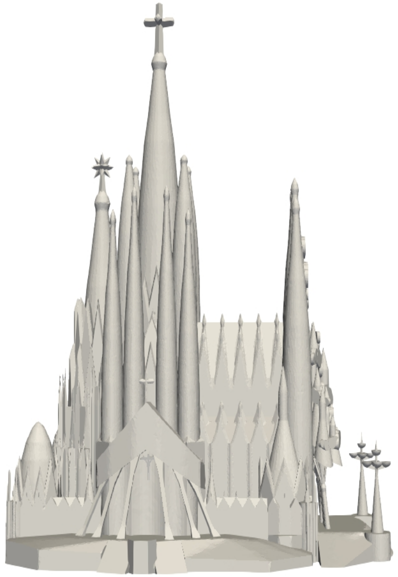
Examples of voronoi meshes of industrial objects

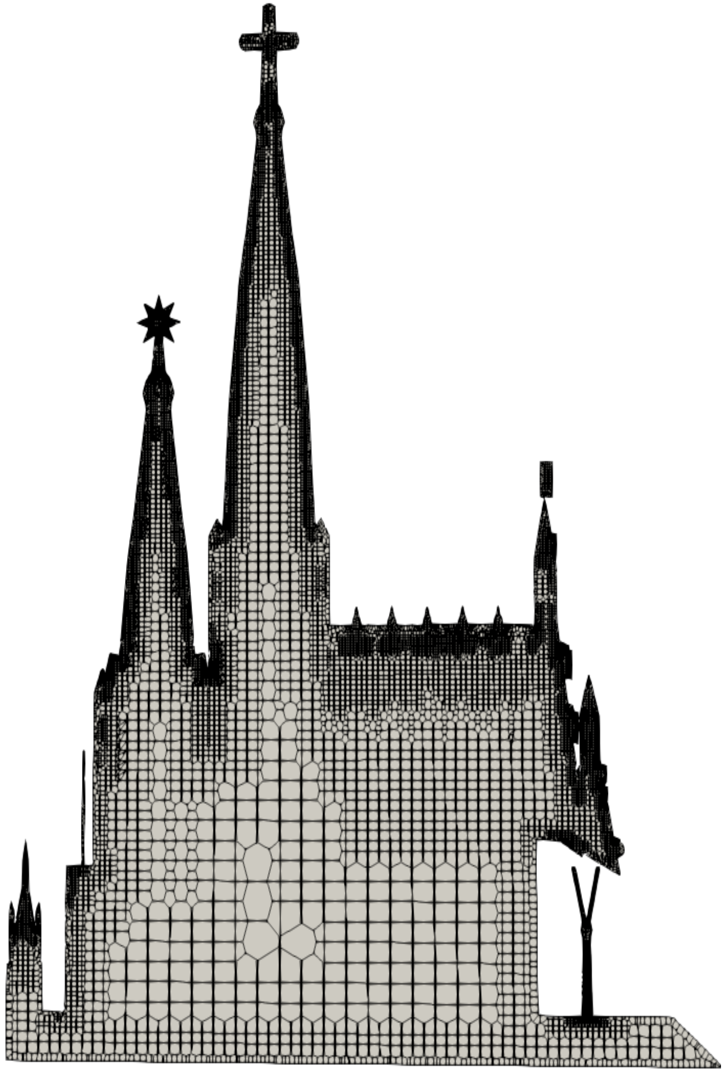
SIEMENS
Ingenuity for life

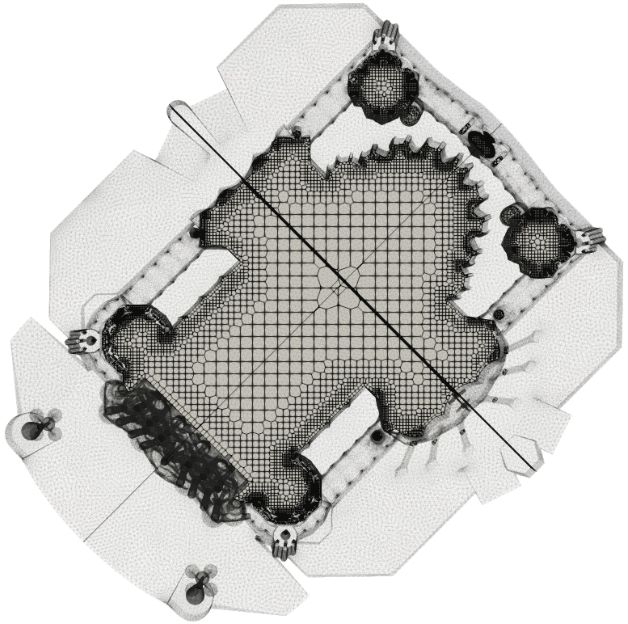


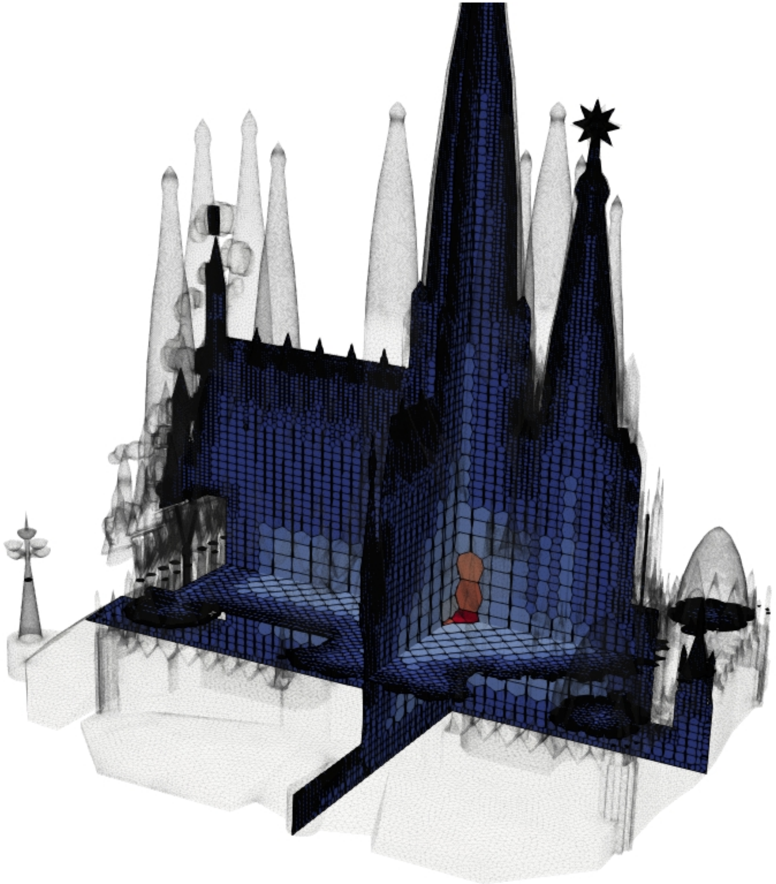


Voronoi mesh of the Sagrada Familia cathedral

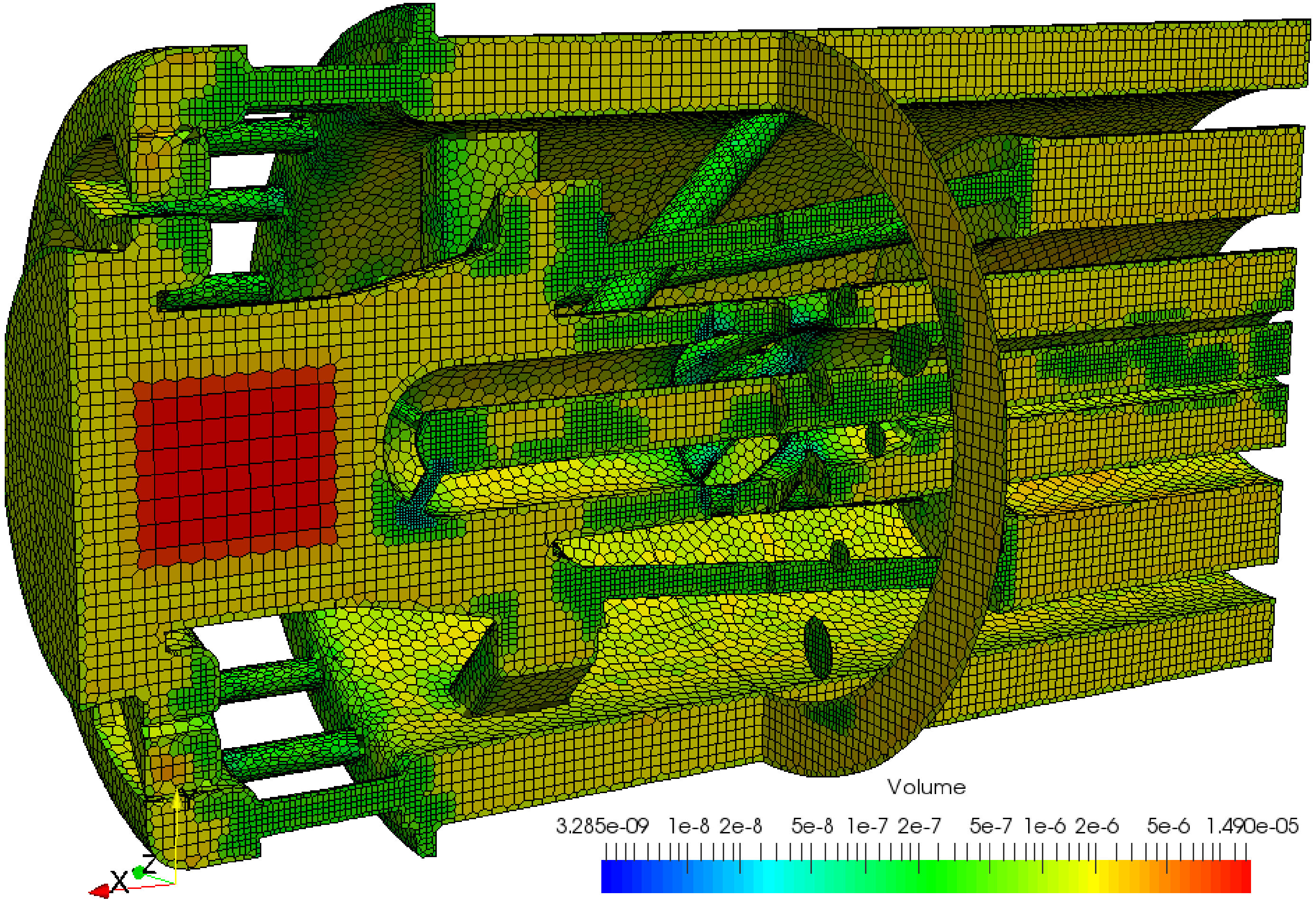








Thank for your attention!



Contact page

Stefano Paoletti
Director of Meshing Research
Siemens SISW

Via Durban 4
00144 Rome Italy

Phone: +39 06 97790200
Mobile: +39 3471250590

E-mail:
stefano.paoletti@siemens.com