Tools for Finite Element Mesh Generation

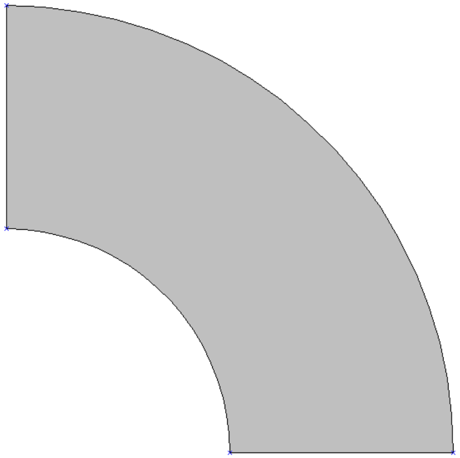
Luiz Fernando Martha

CIV 2802 – Sistemas Gráficos para Engenharia
Departamento de Engenharia Civil – PUC-Rio
2013.1

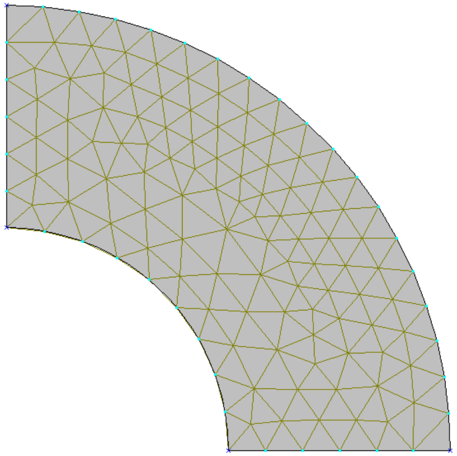
Tecgraf PUC-RIO

PUC RIO

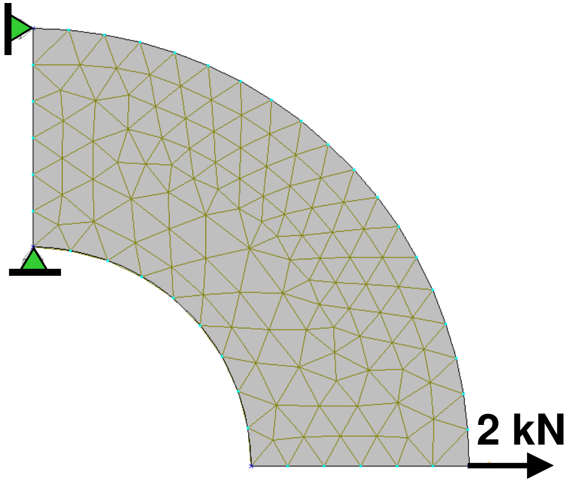
Traditional FE simulation process



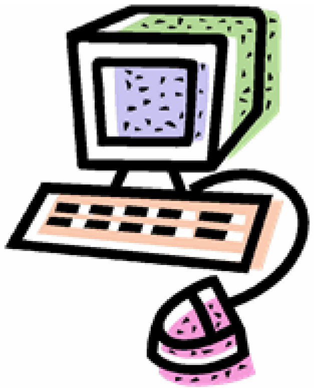
1. Build geometric model



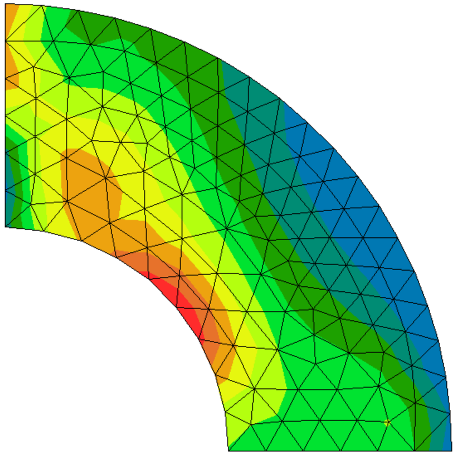
2. Mesh



3. Apply boundary conditions

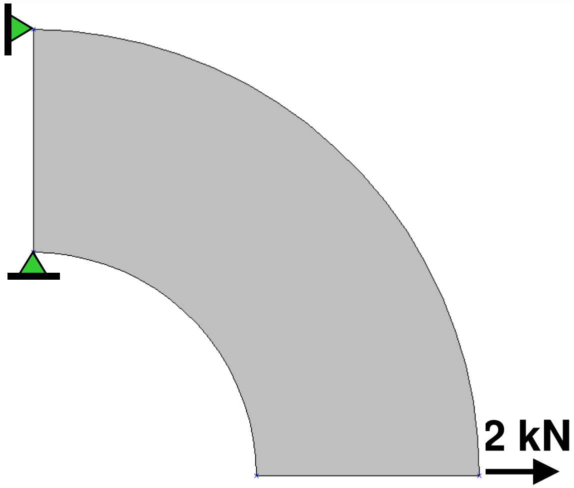


4. Computational analysis

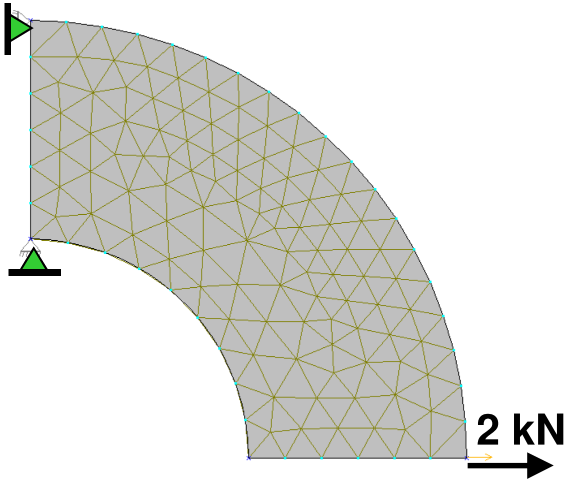


5. Result visualization

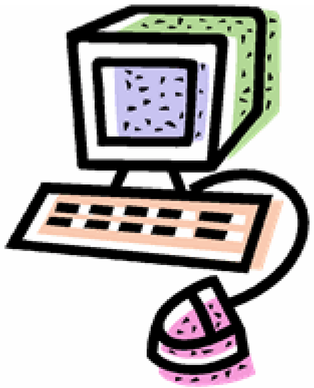
Geometry-based FE simulation process



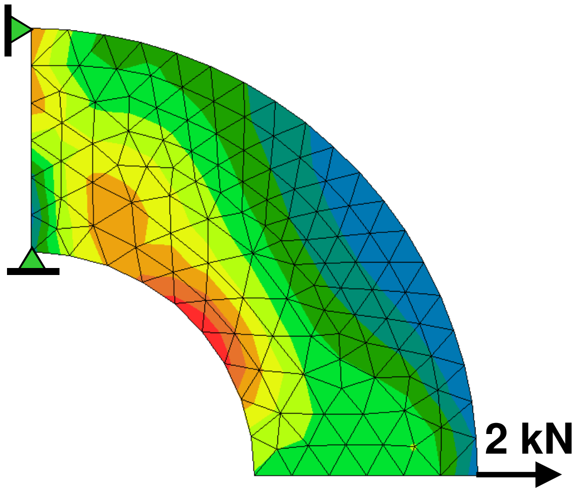
1. Geometric modelling, apply attributes and boundary conditions



2. FE mesh generation, apply boundary conditions



3. Computational analysis

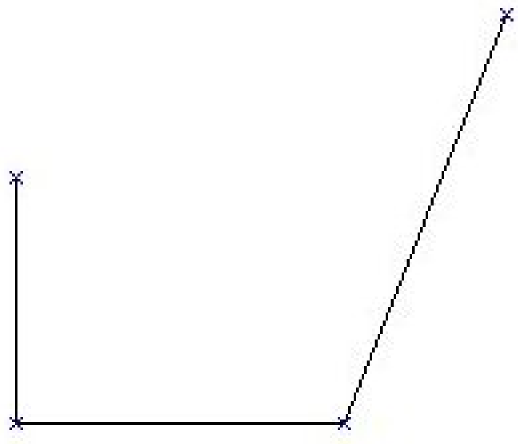


4. Result visualization

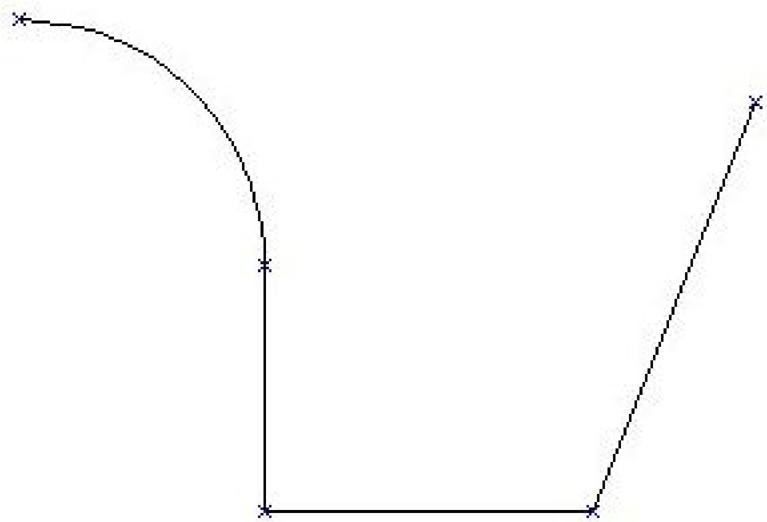
Construction of a simple FE model

Construction of a simple FE model

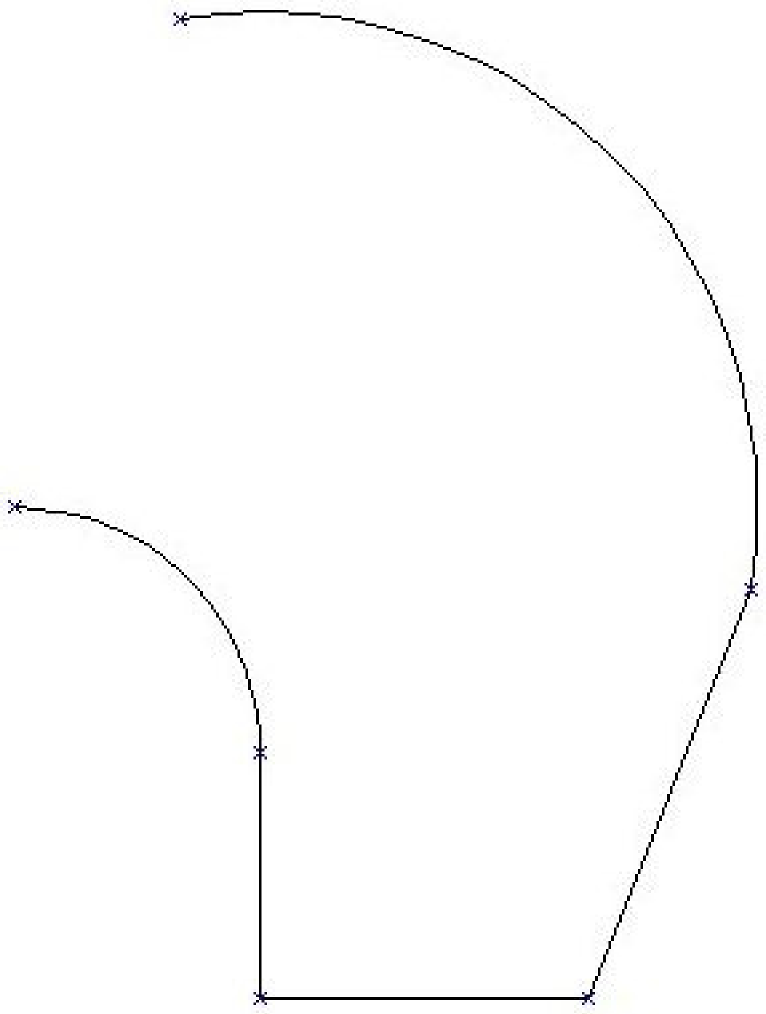
Construction of a simple FE model



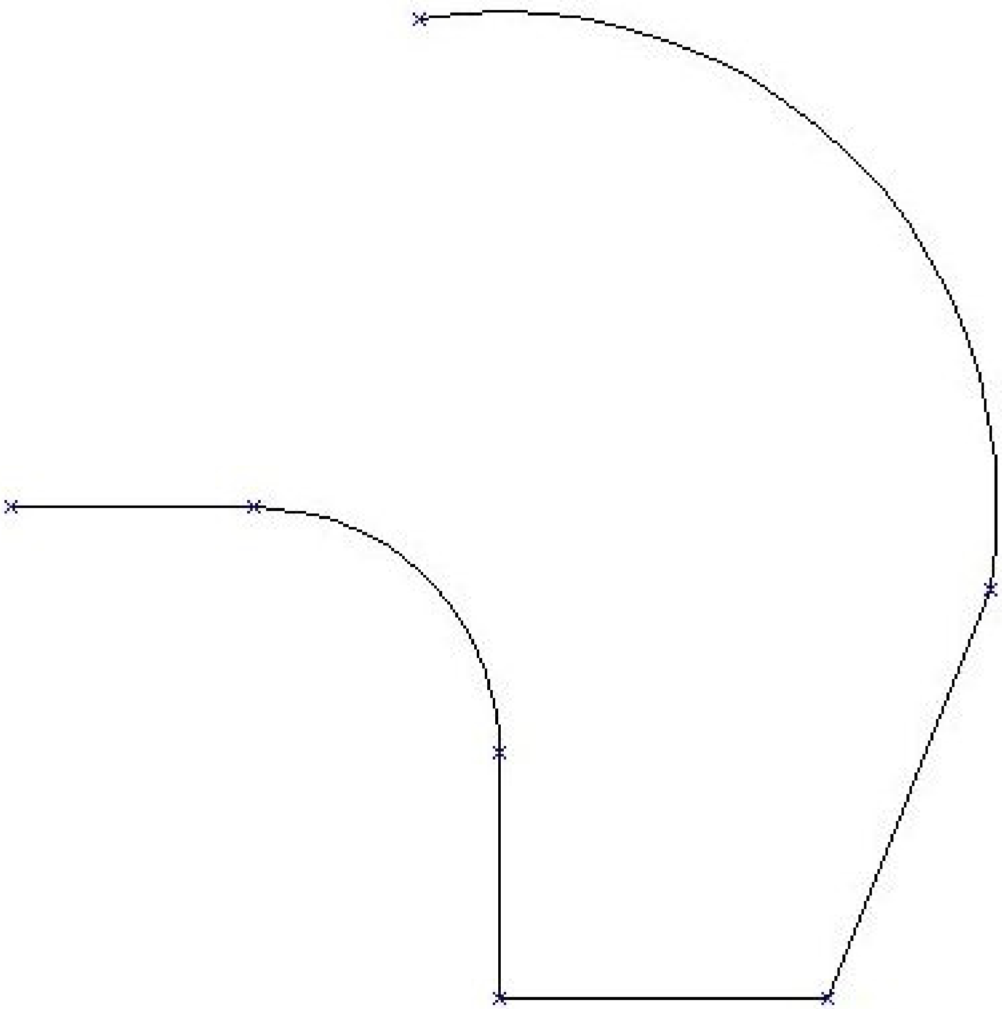
Construction of a simple FE model



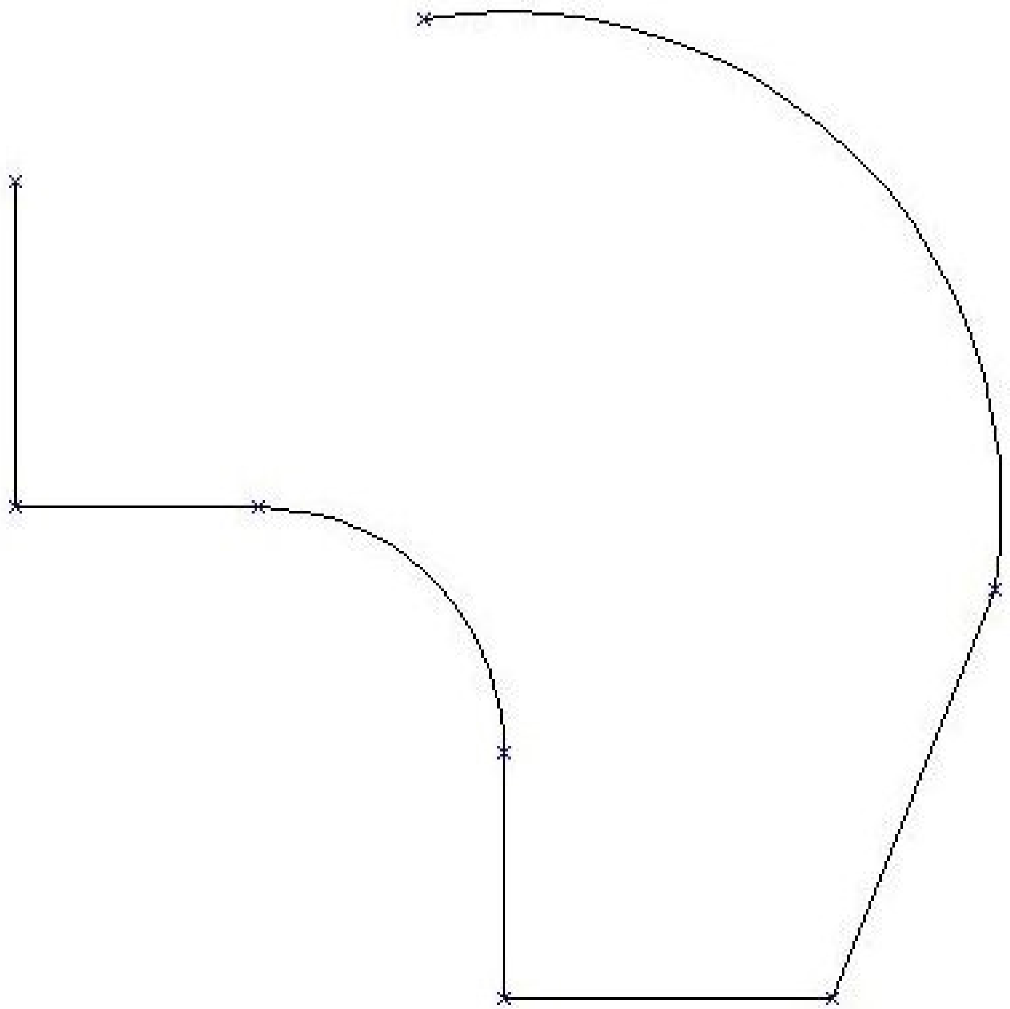
Construction of a simple FE model



Construction of a simple FE model

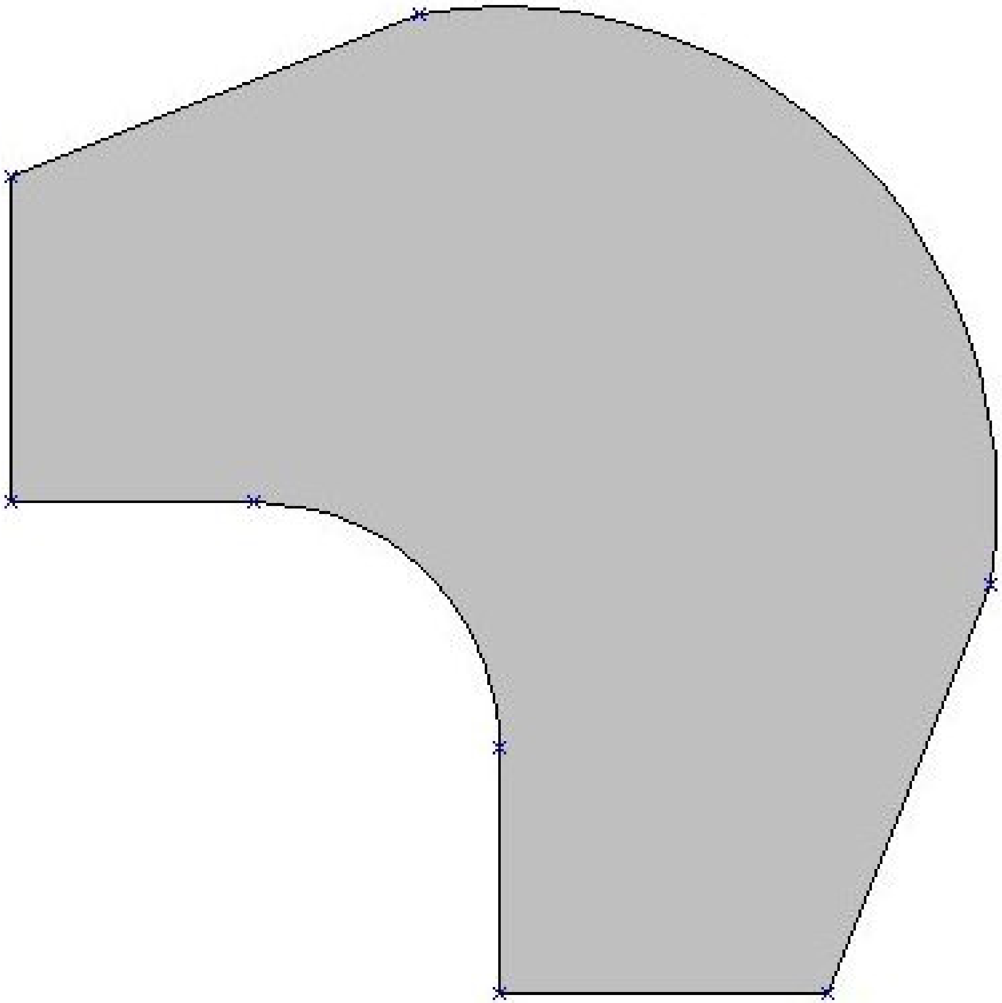


Construction of a simple FE model



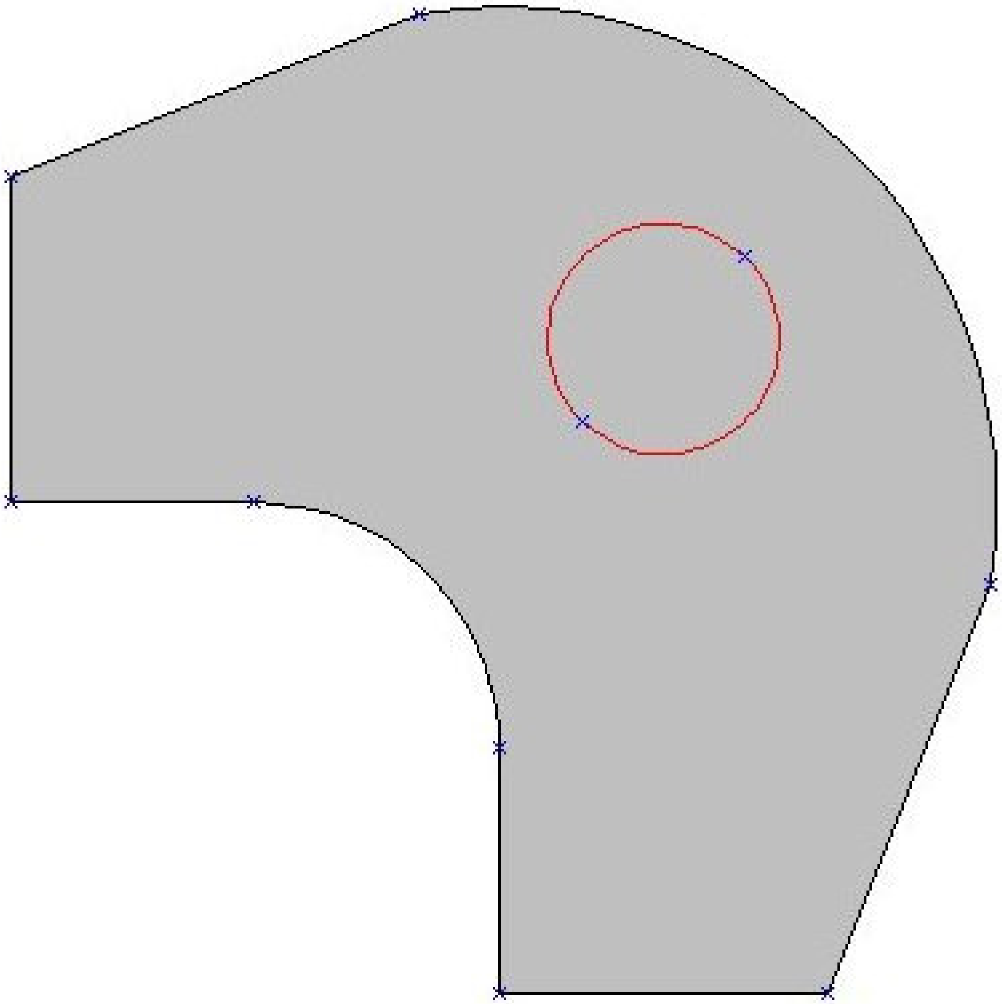
Construction of a simple FE model

Automatic region recognition

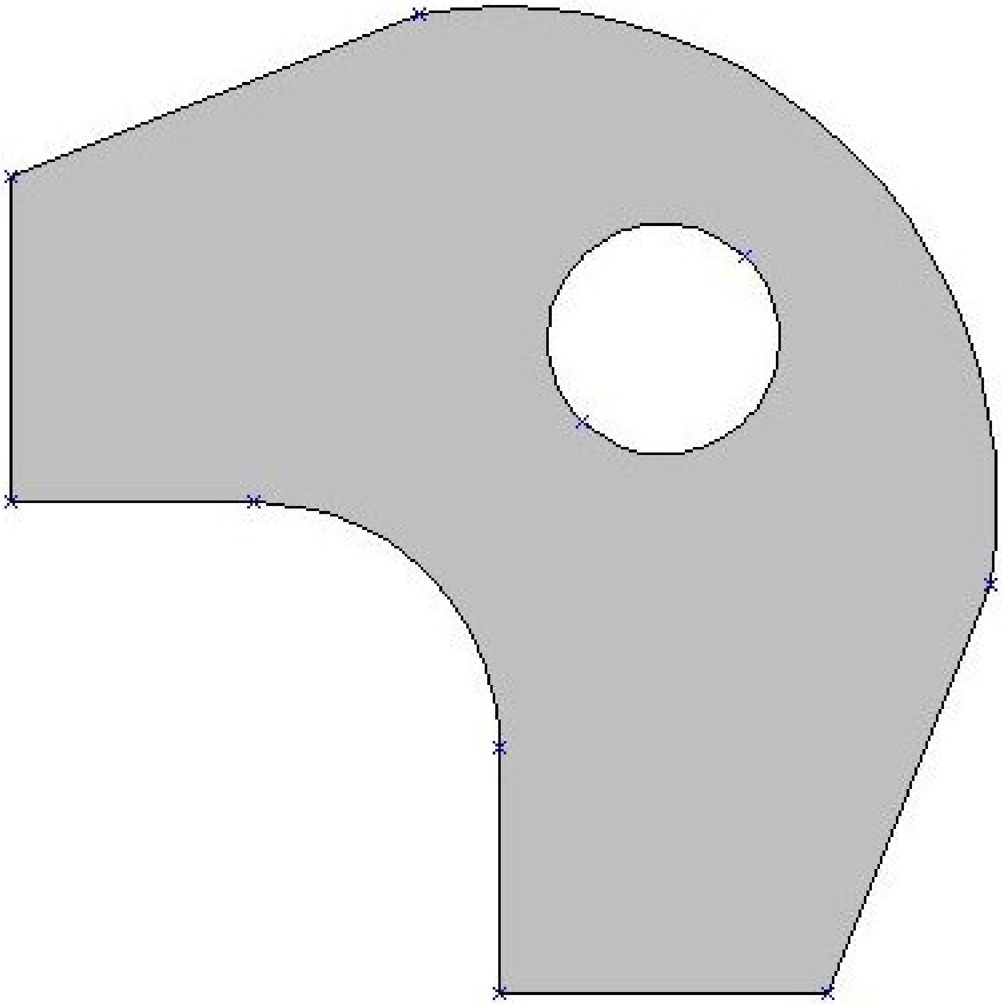


Construction of a simple FE model

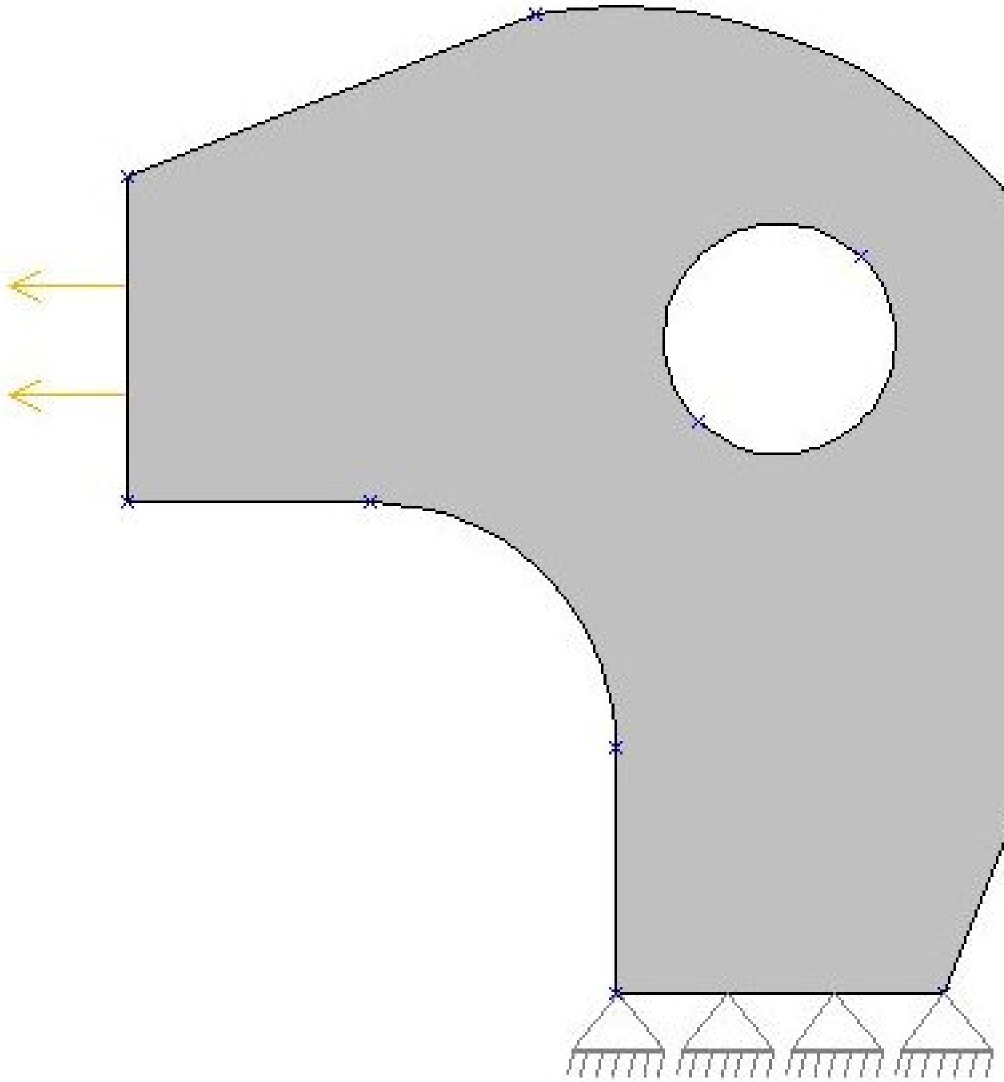
Creating a hole



Assigning hole attribute

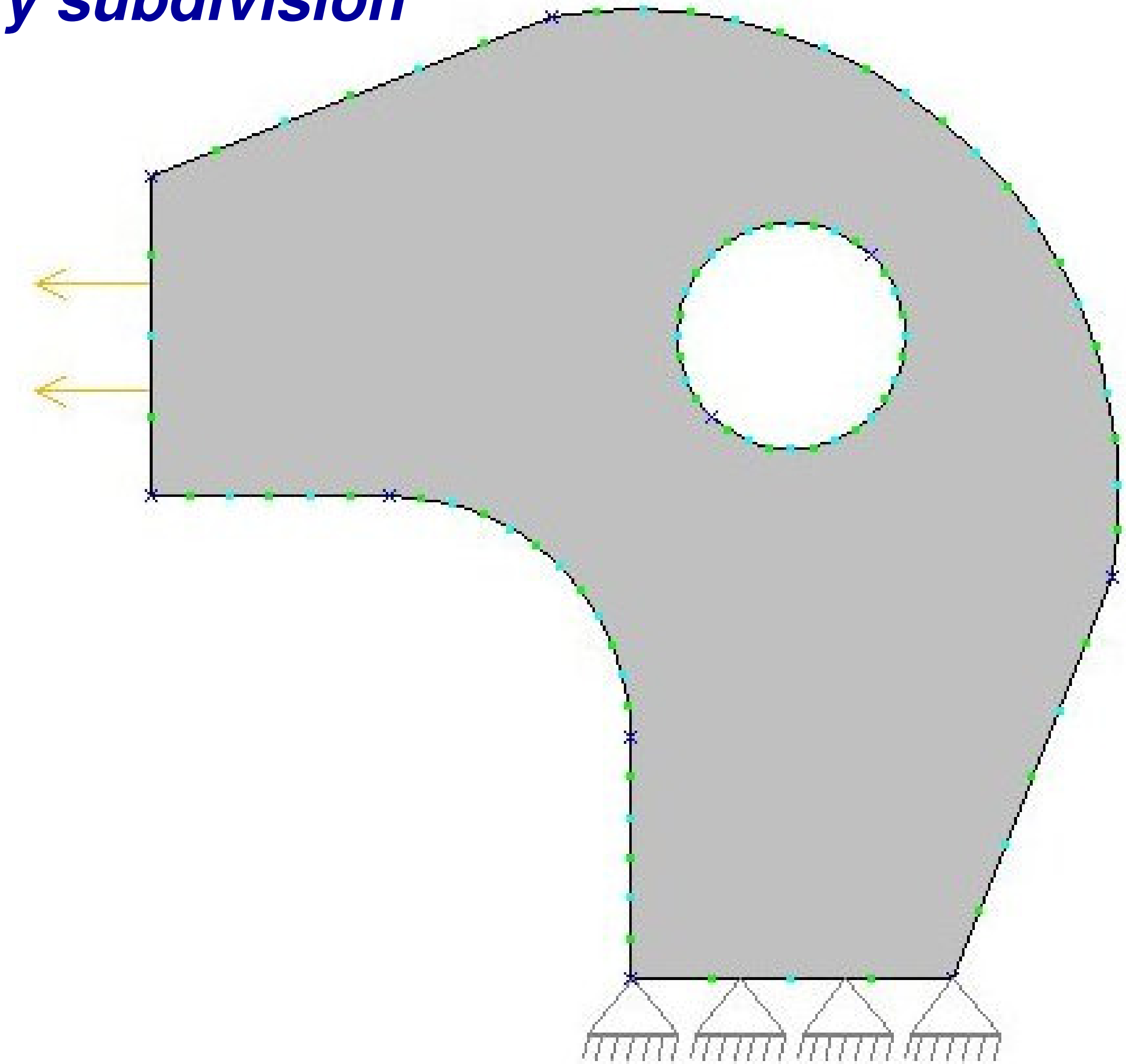


Applying attributes to geometry



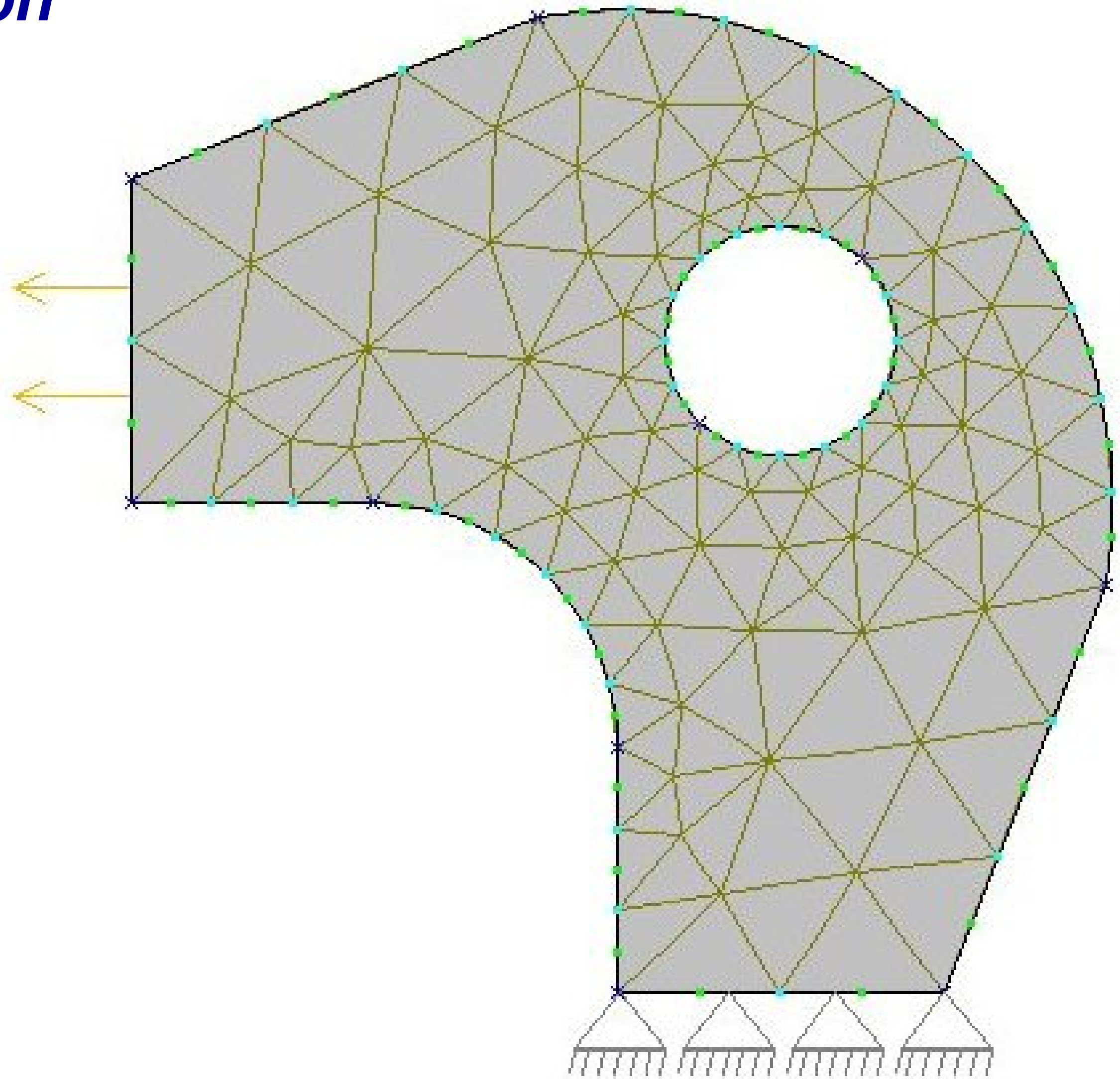
Construction of a simple FE model

Defining meshing refinement parameters: boundary subdivision



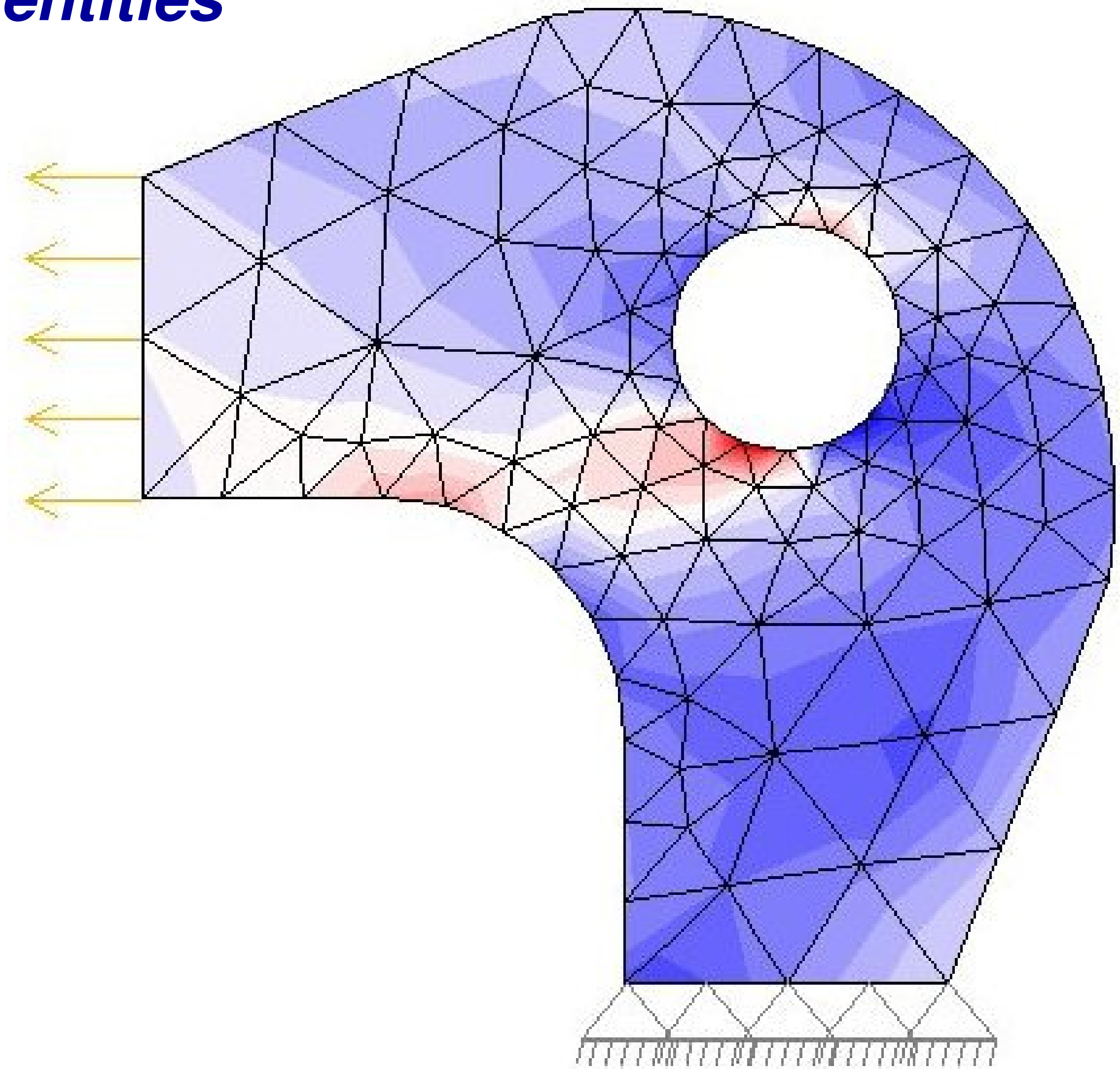
Construction of a simple FE model

Automatic unstructured mesh generation



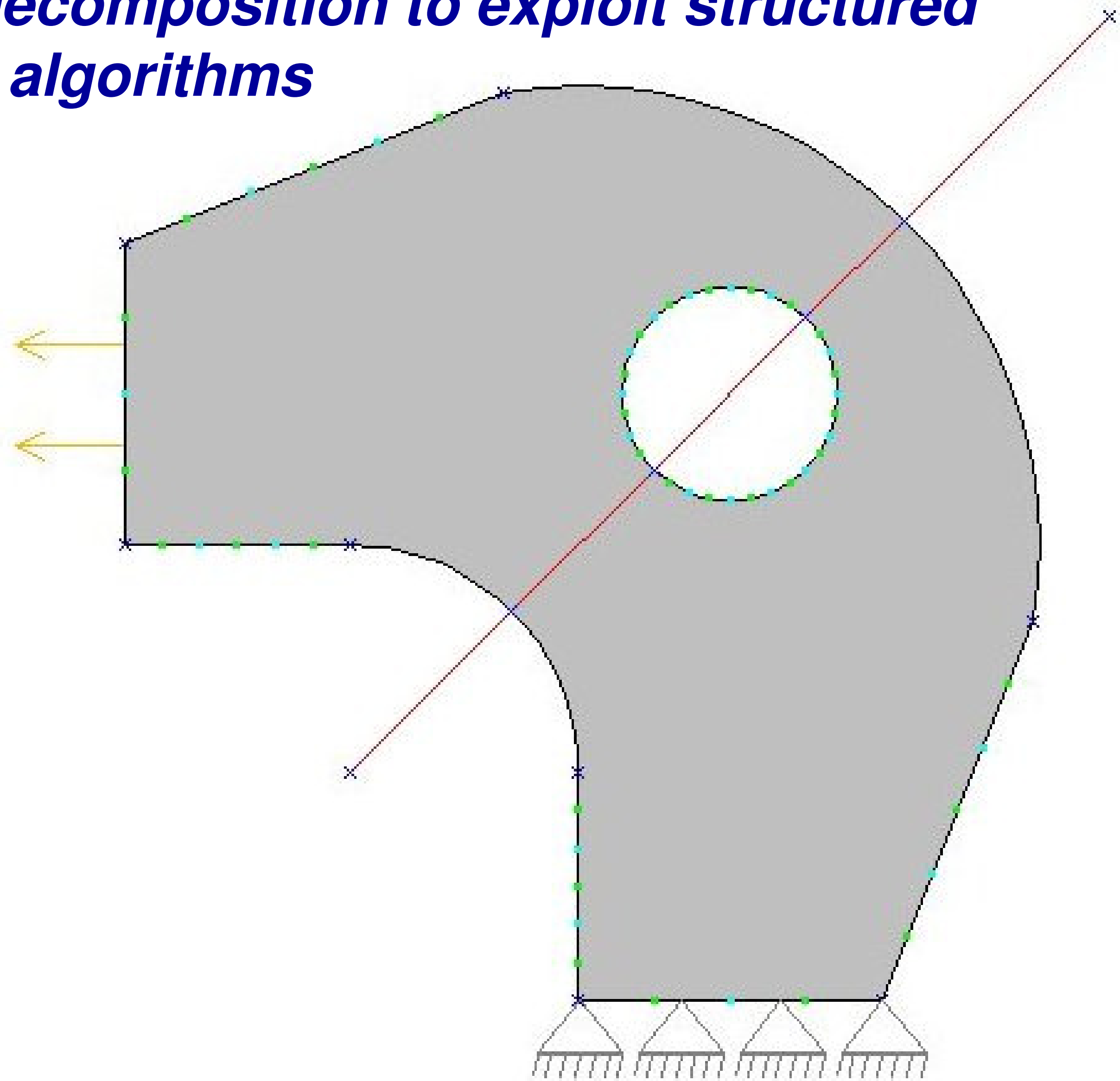
Construction of a simple FE model

Attributes automatically assigned to mesh entities

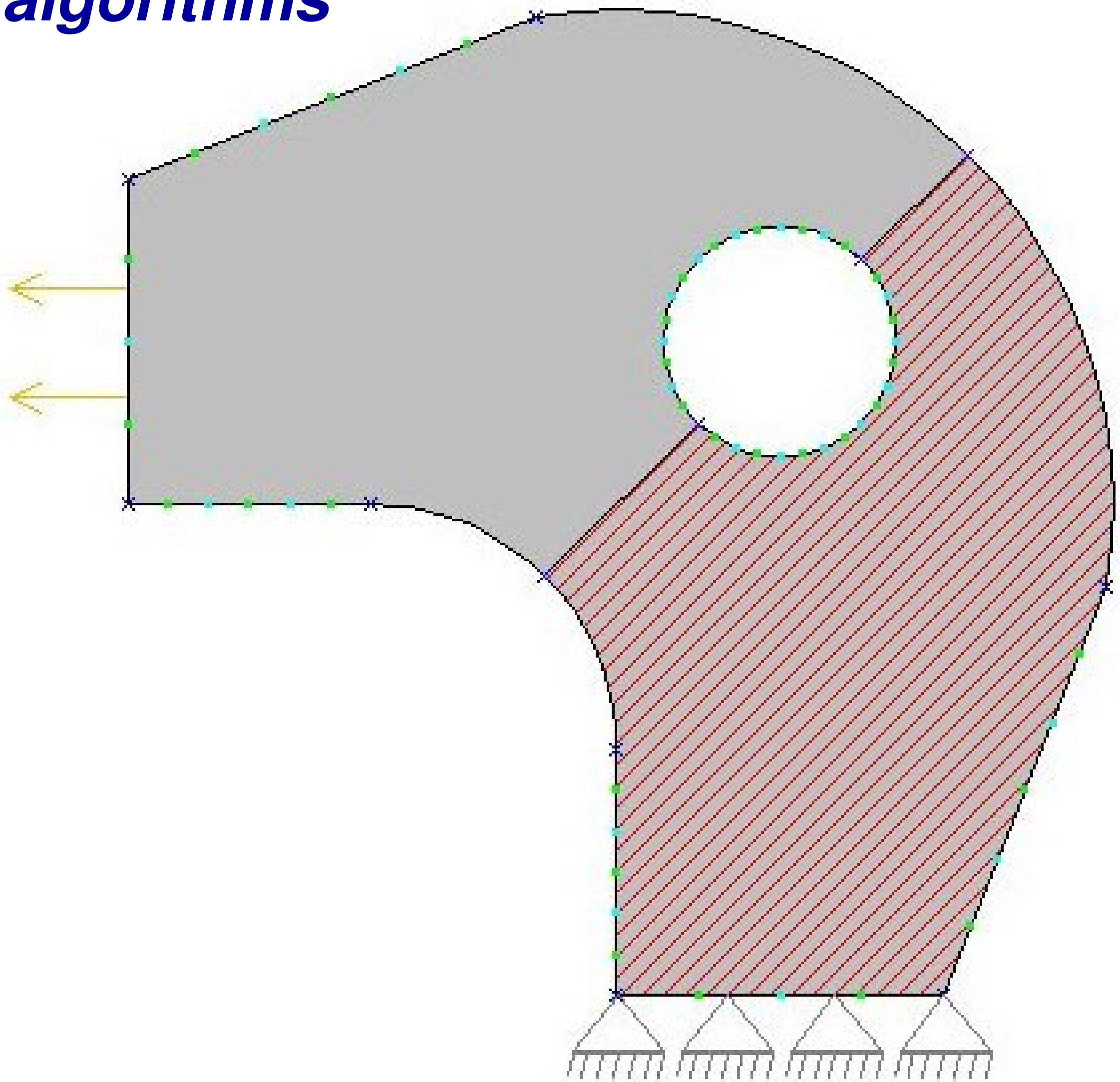


Construction of a simple FE model

Region decomposition to exploit structured meshing algorithms

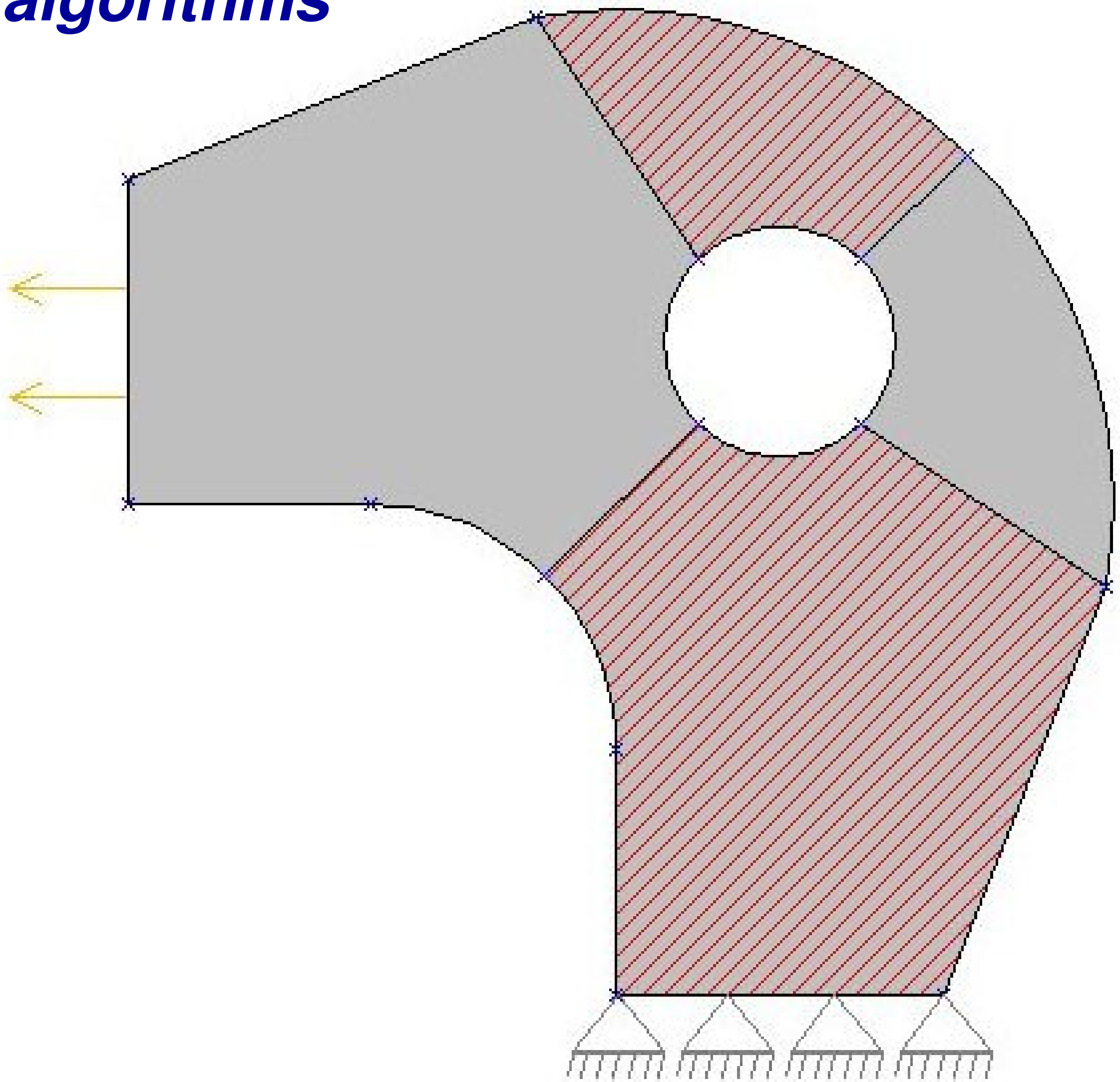


Region decomposition to exploit structured meshing algorithms



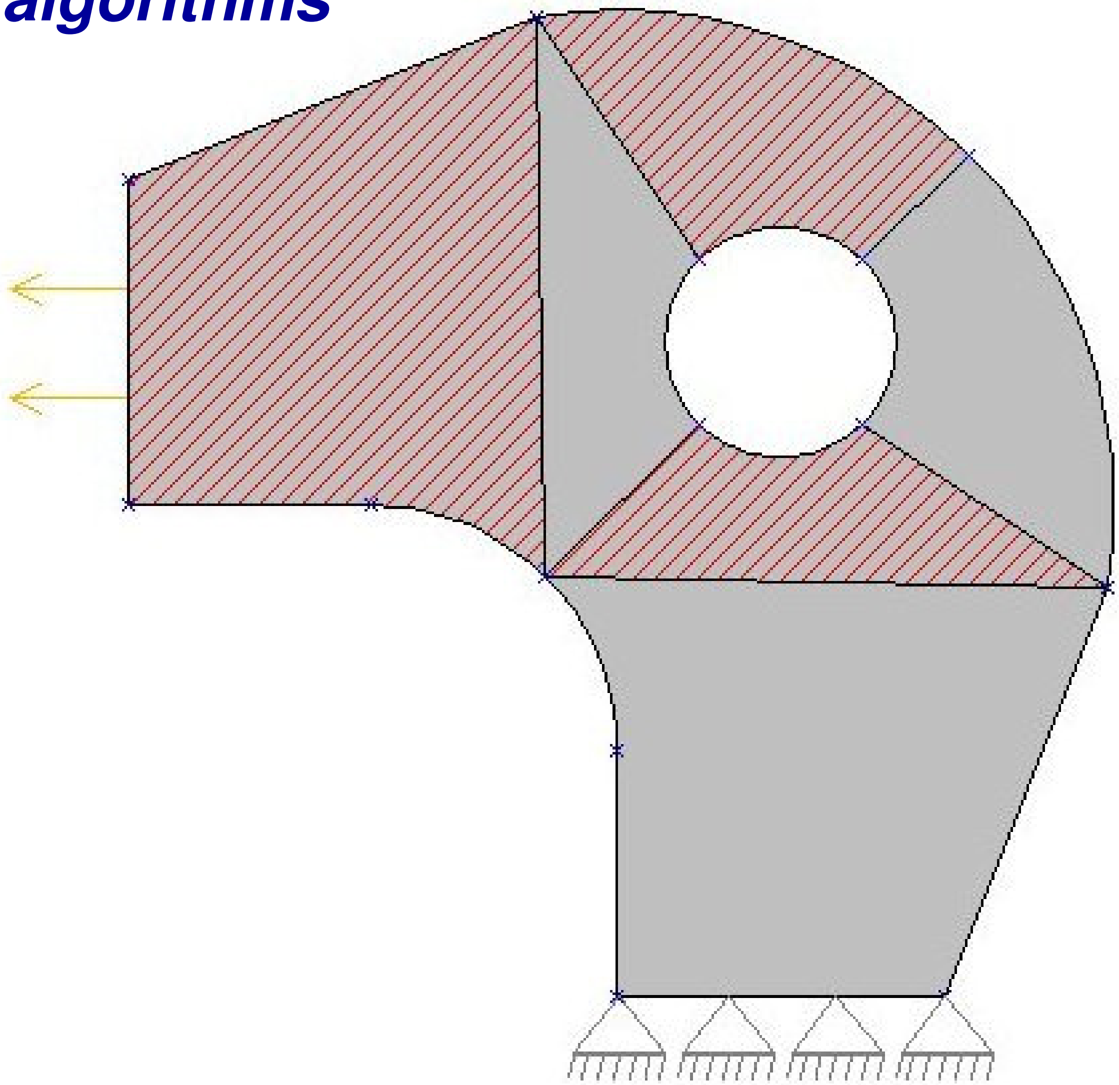
Construction of a simple FE model

Region decomposition to exploit structured meshing algorithms



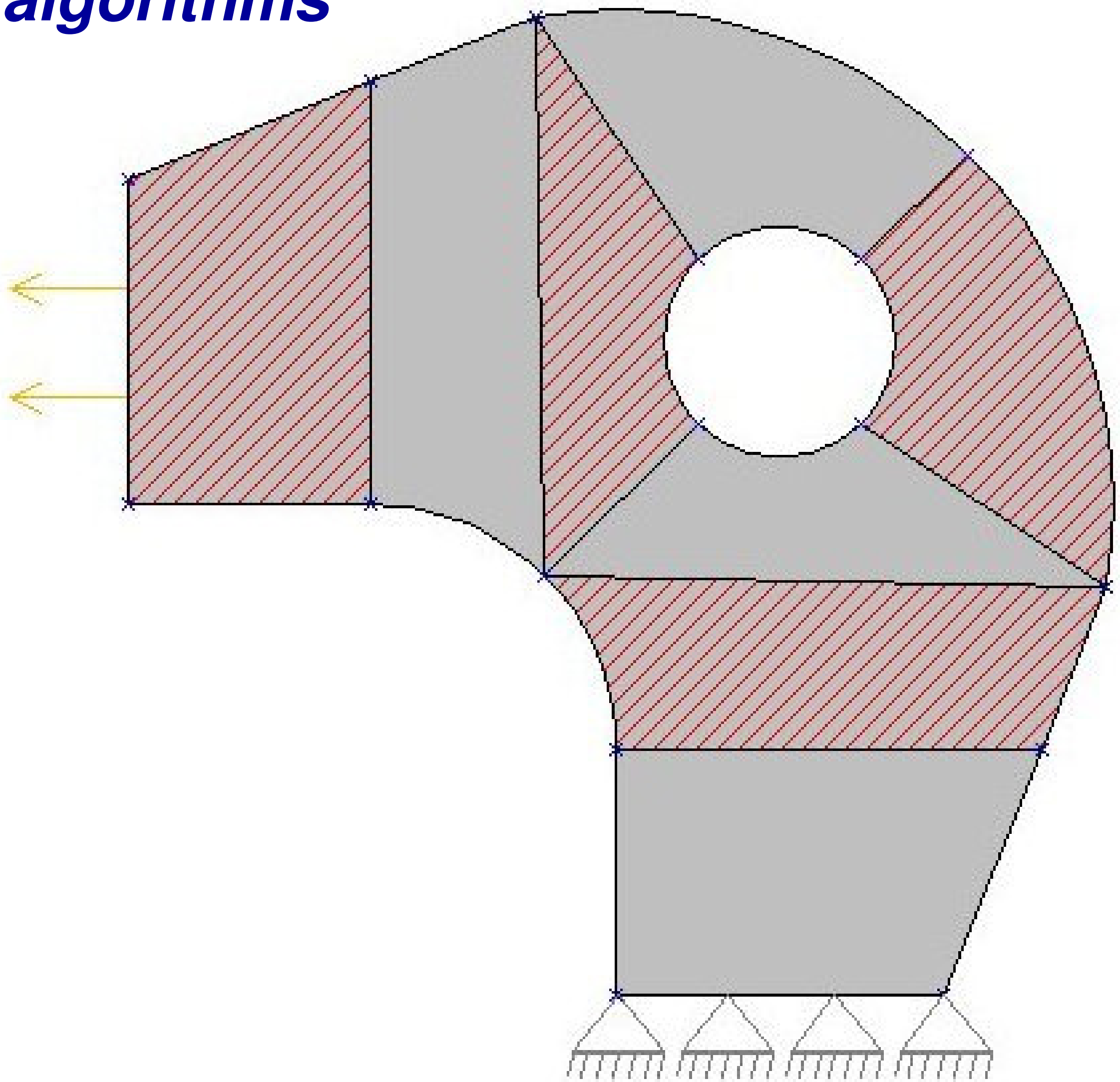
Construction of a simple FE model

Region decomposition to exploit structured meshing algorithms



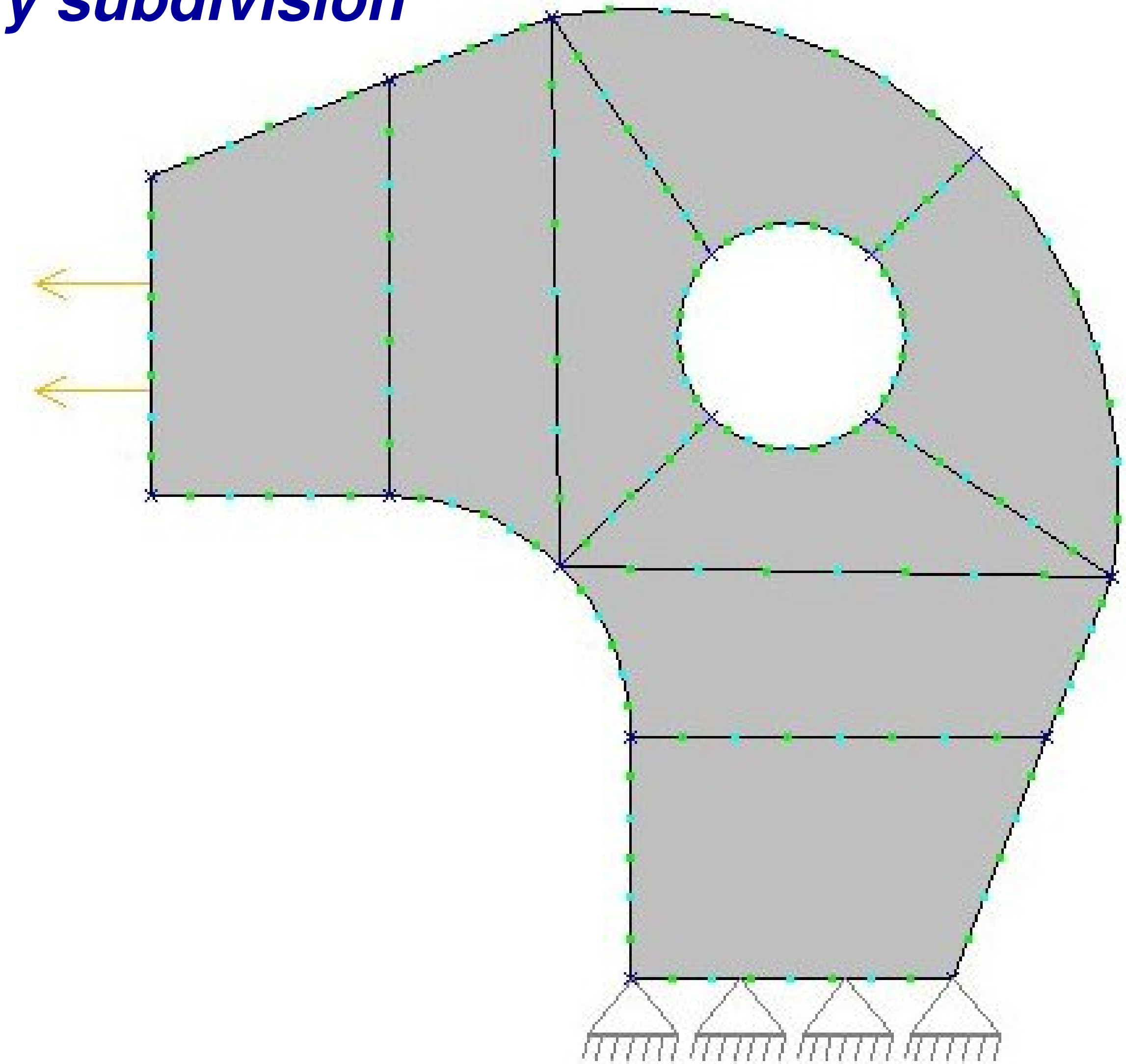
Construction of a simple FE model

Region decomposition to exploit structured meshing algorithms

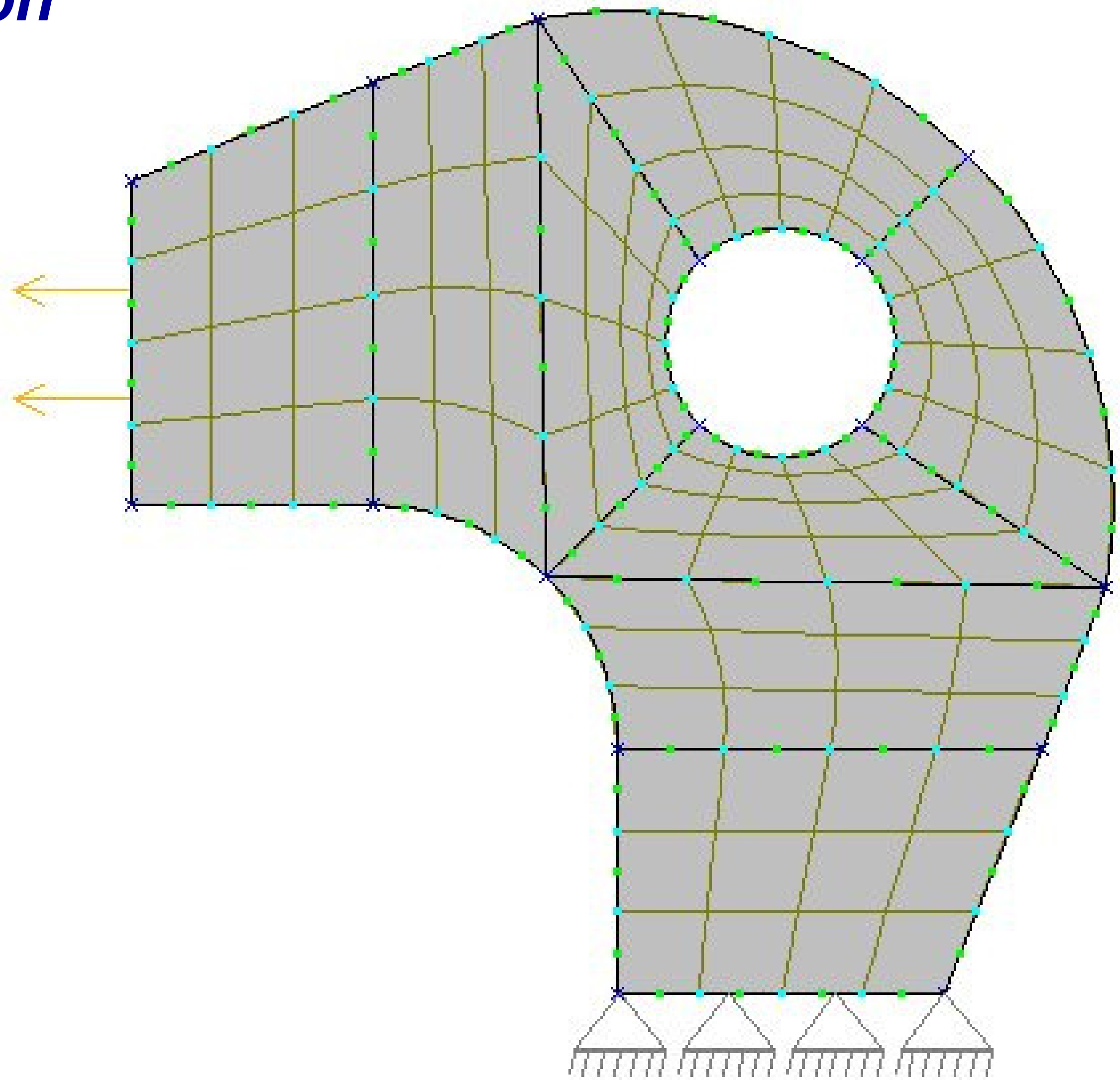


Construction of a simple FE model

Defining meshing refinement parameters: boundary subdivision

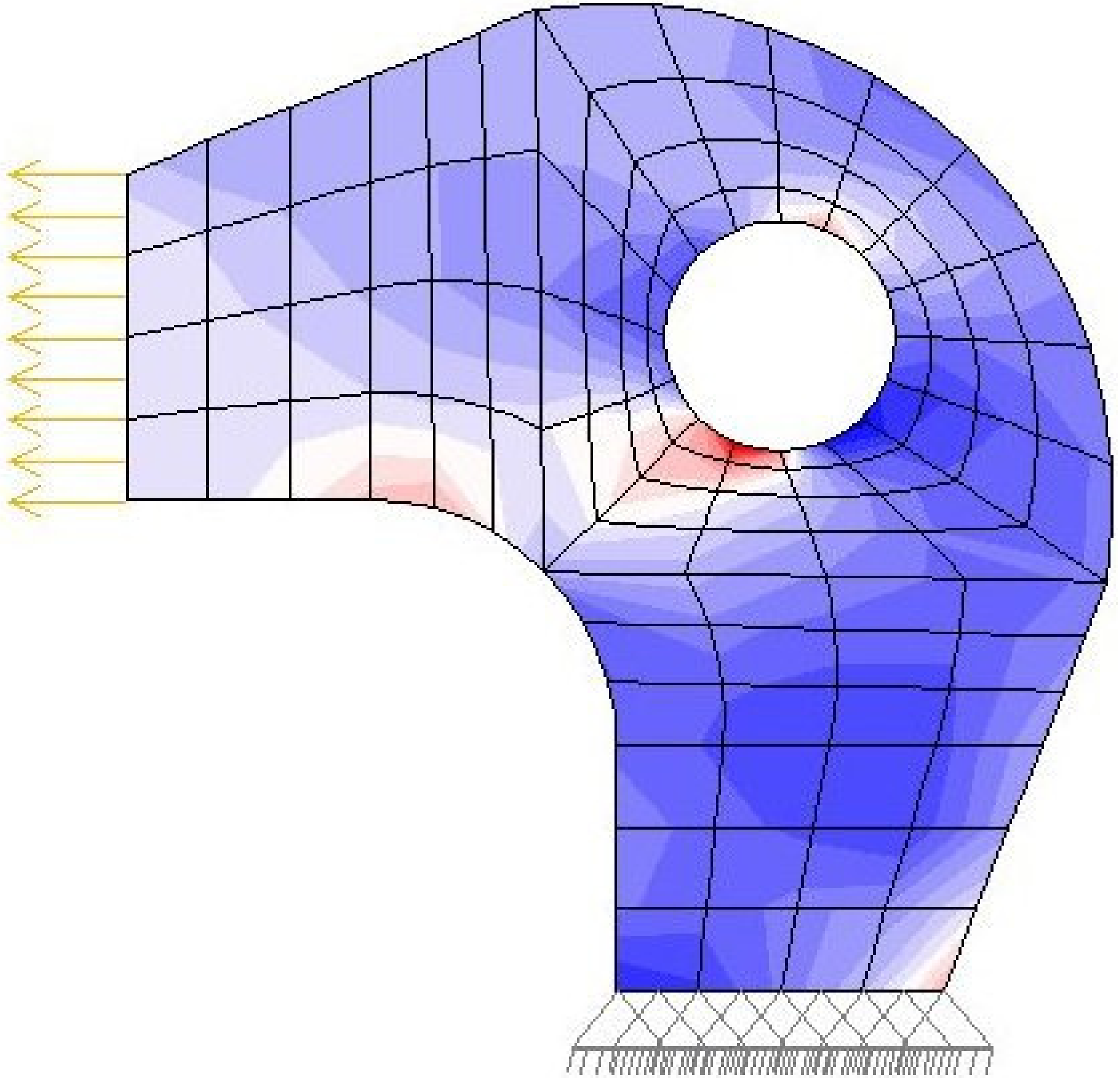


Automatic structured mesh generation



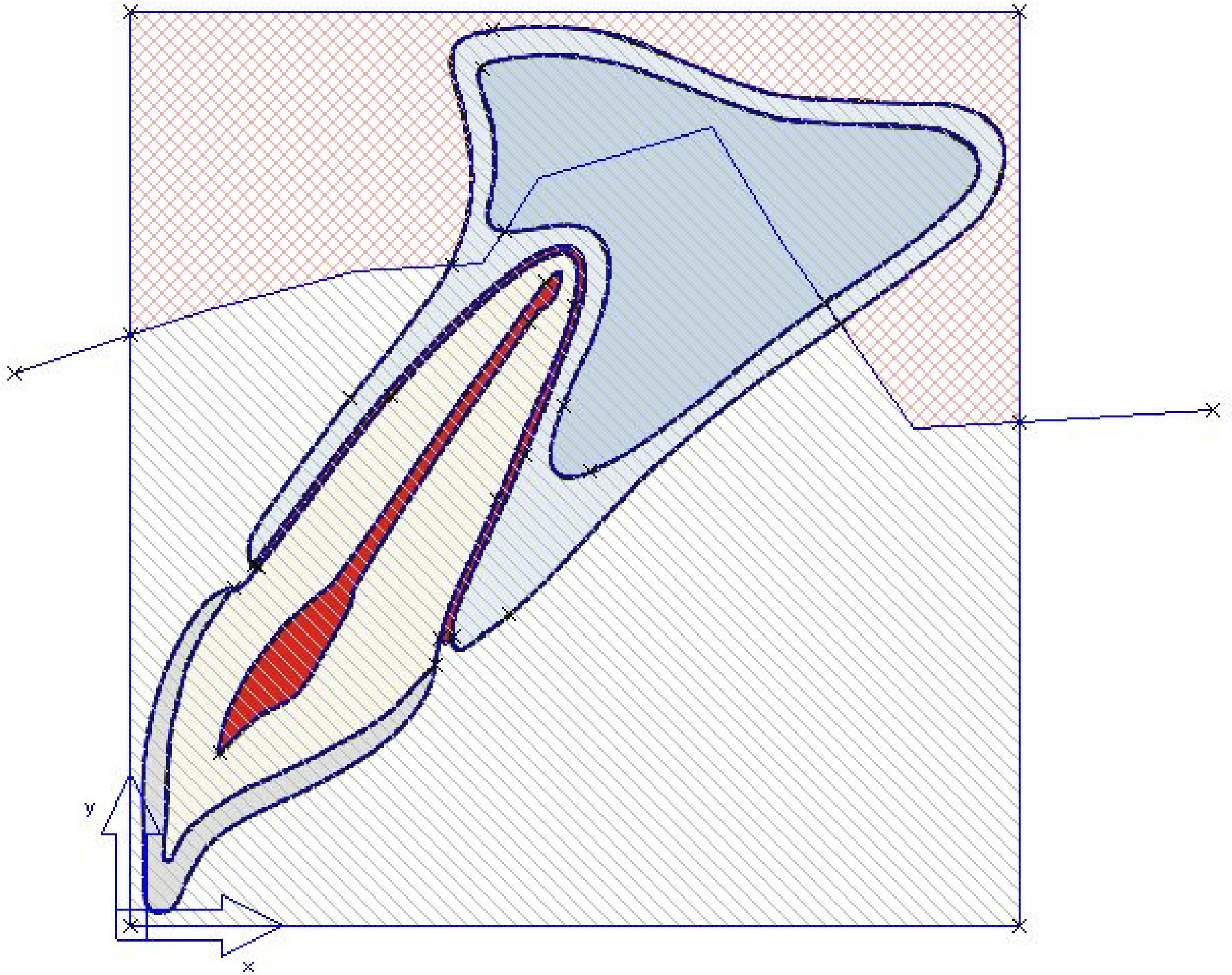
Construction of a simple FE model

What is the technology behind this?
What issues we have to address?

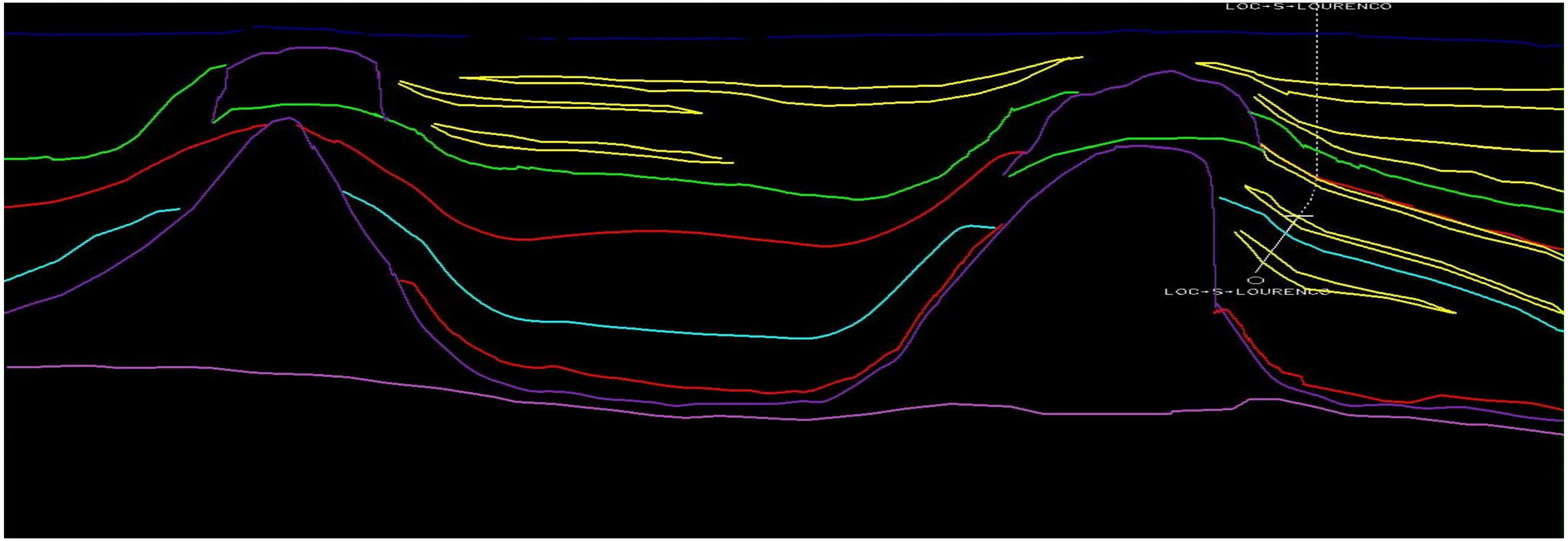


Generic space subdivision: many applications

An environment in which curves and surfaces are inserted randomly.
Automatic region recognition and full adjacency information.

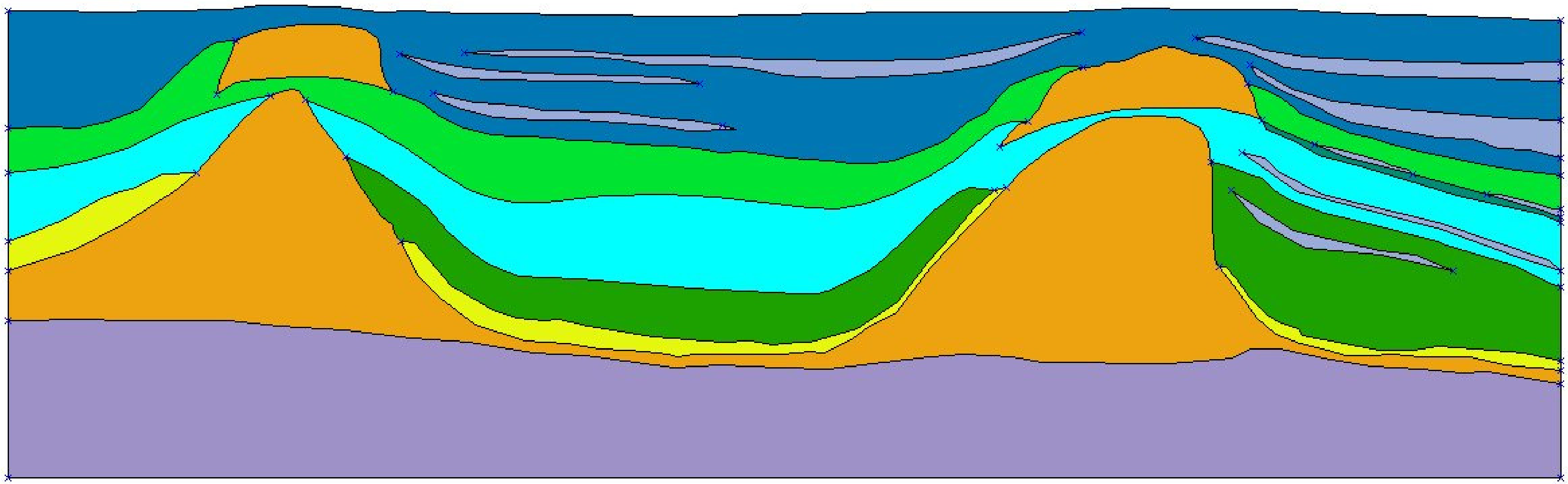


2D subsurface simulation modeling



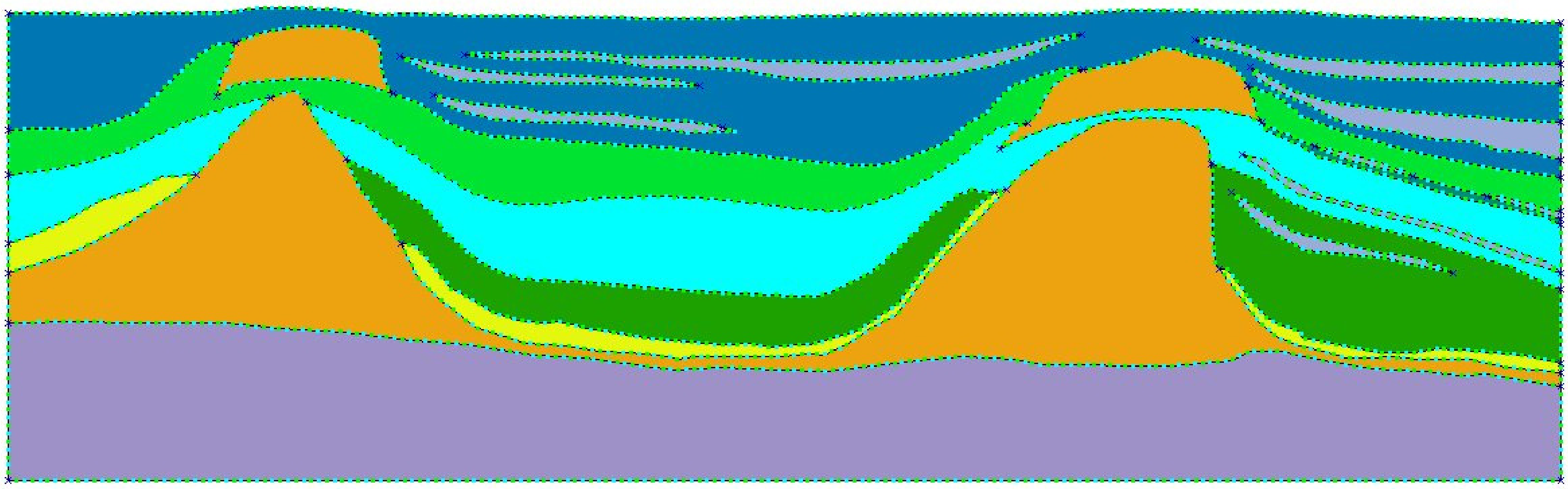
2D subsurface simulation modeling

Curve digitalization



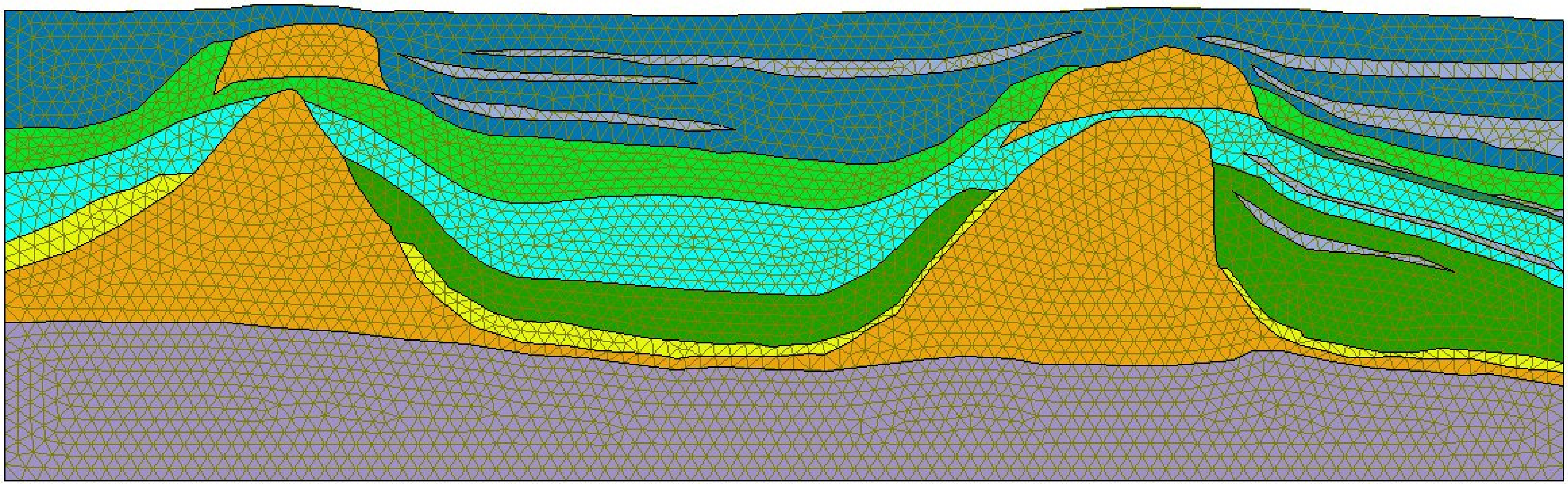
2D subsurface simulation modeling

Curve subdivision



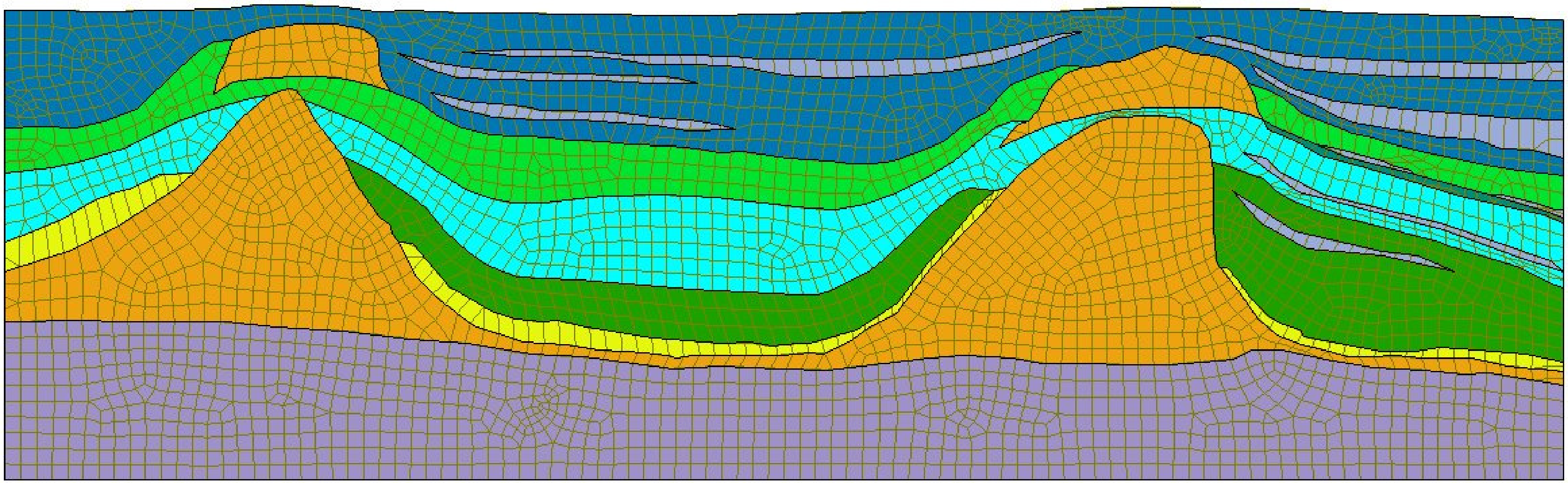
2D subsurface simulation modeling

Mesh generation: triangular elements



2D subsurface simulation modeling

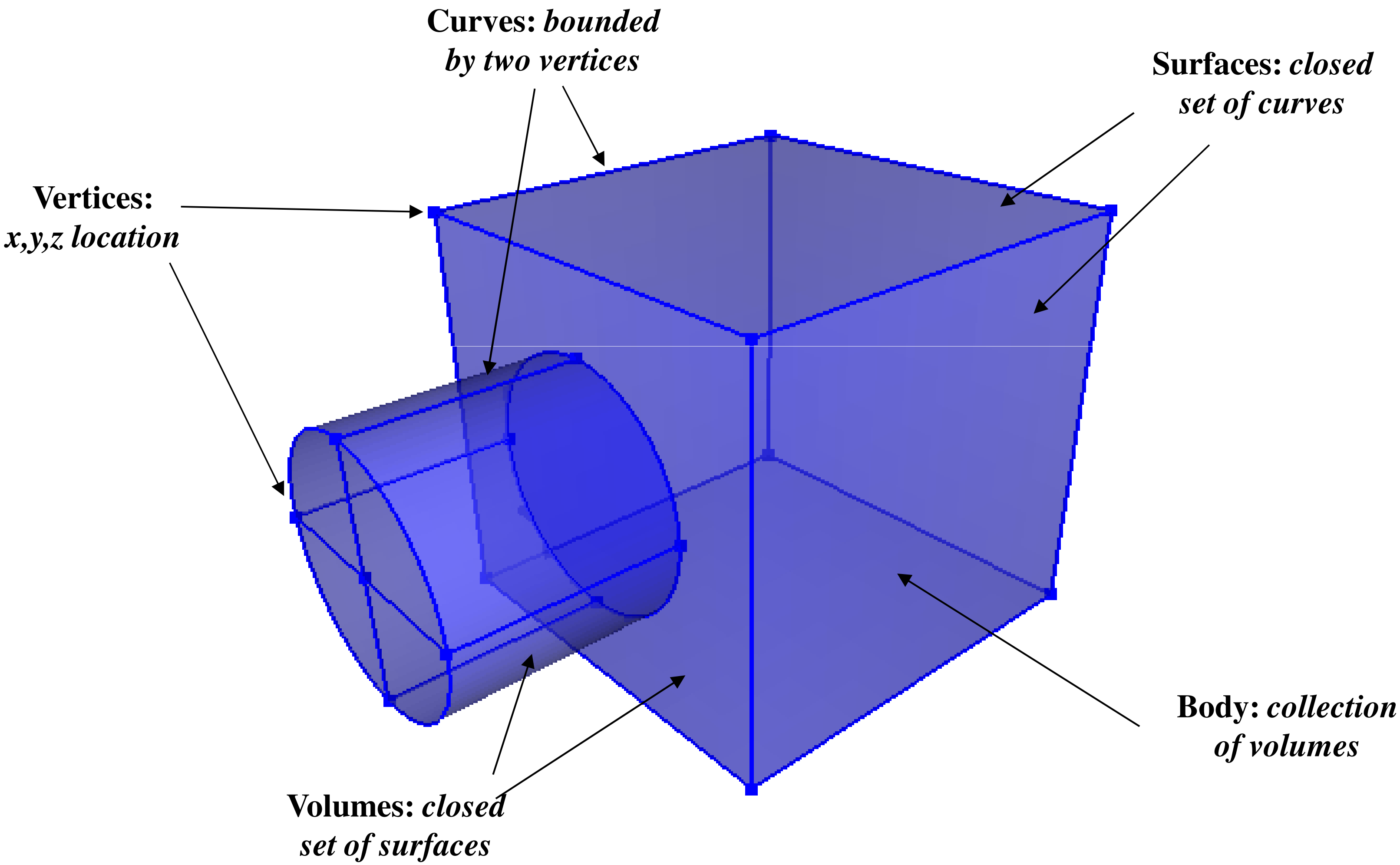
Mesh generation: quadrilateral elements



Geometric modeling

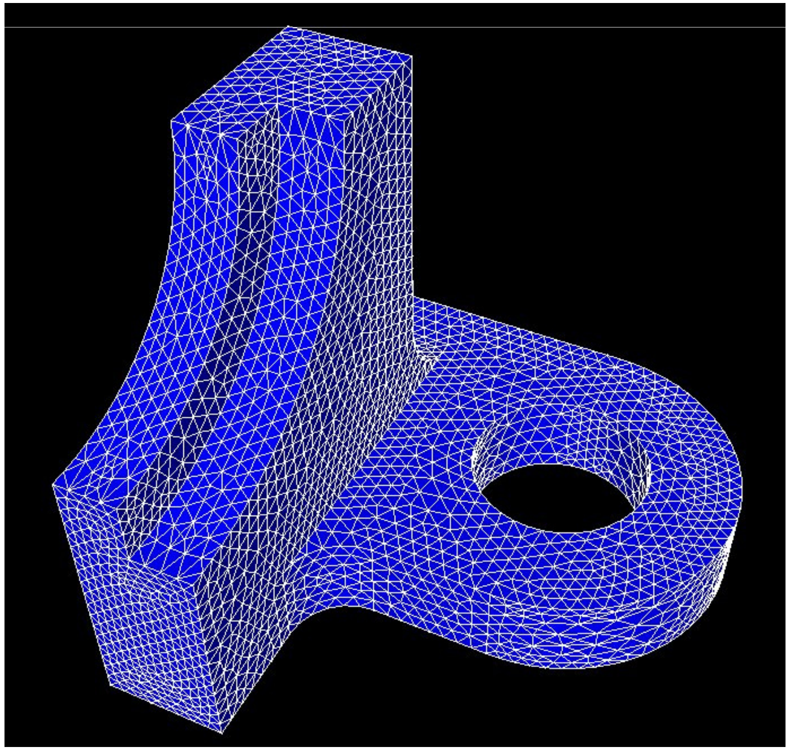
Geometric modeling

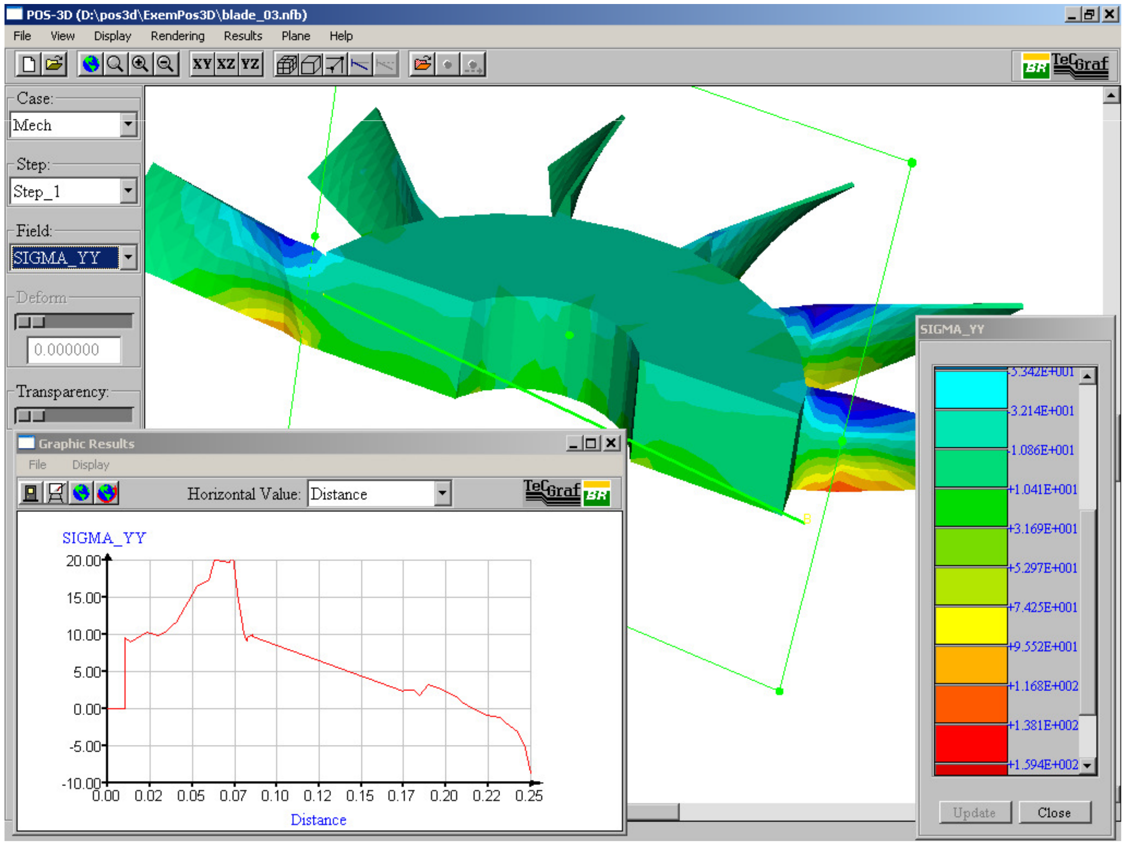
Geometry definitions



Requirements for underlying data representation

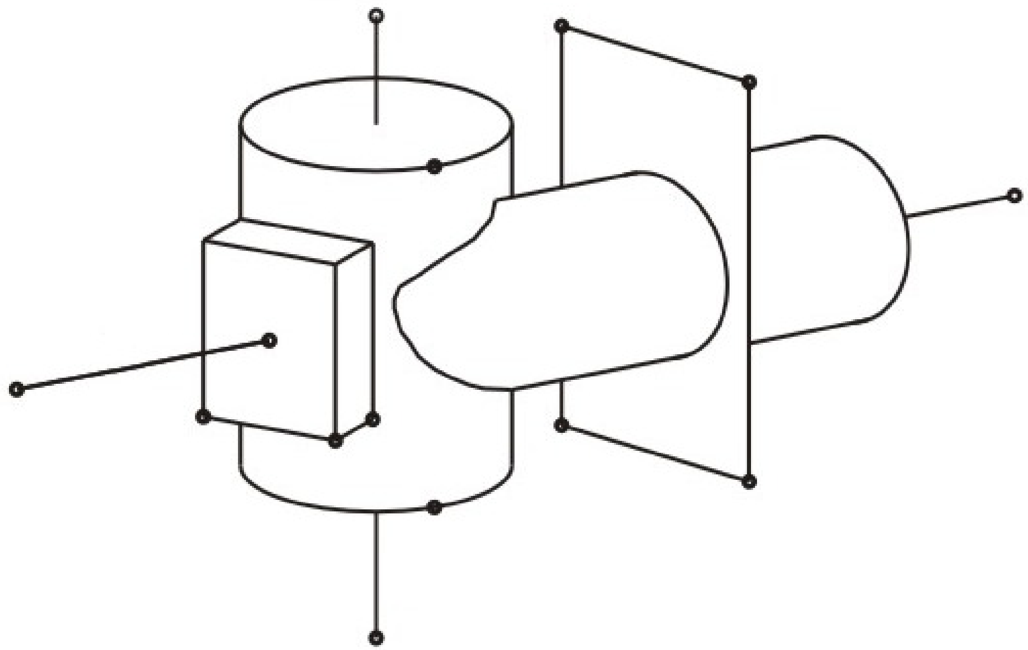
– ***The data structures must provide a natural navigation across all phases of a simulation: pre-processing (model creation), numerical analysis, and post-processing (model results visualization).***





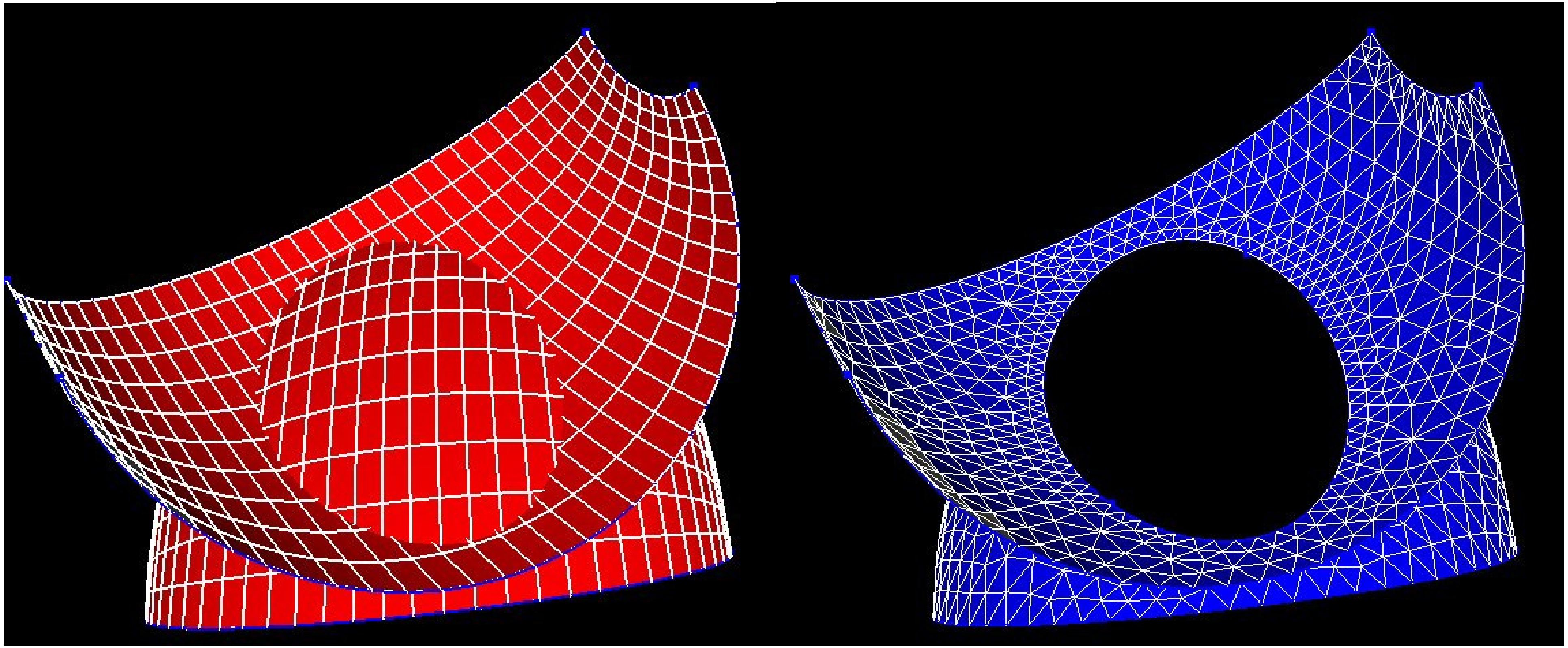
Requirements for underlying data representation (cont.)

– ***The data structures must take into account that the simulation may induce, at least temporarily during model creation, geometric objects (curves and surfaces) that are inconsistent with the target final model. This requires a non-manifold topology representation capability.***



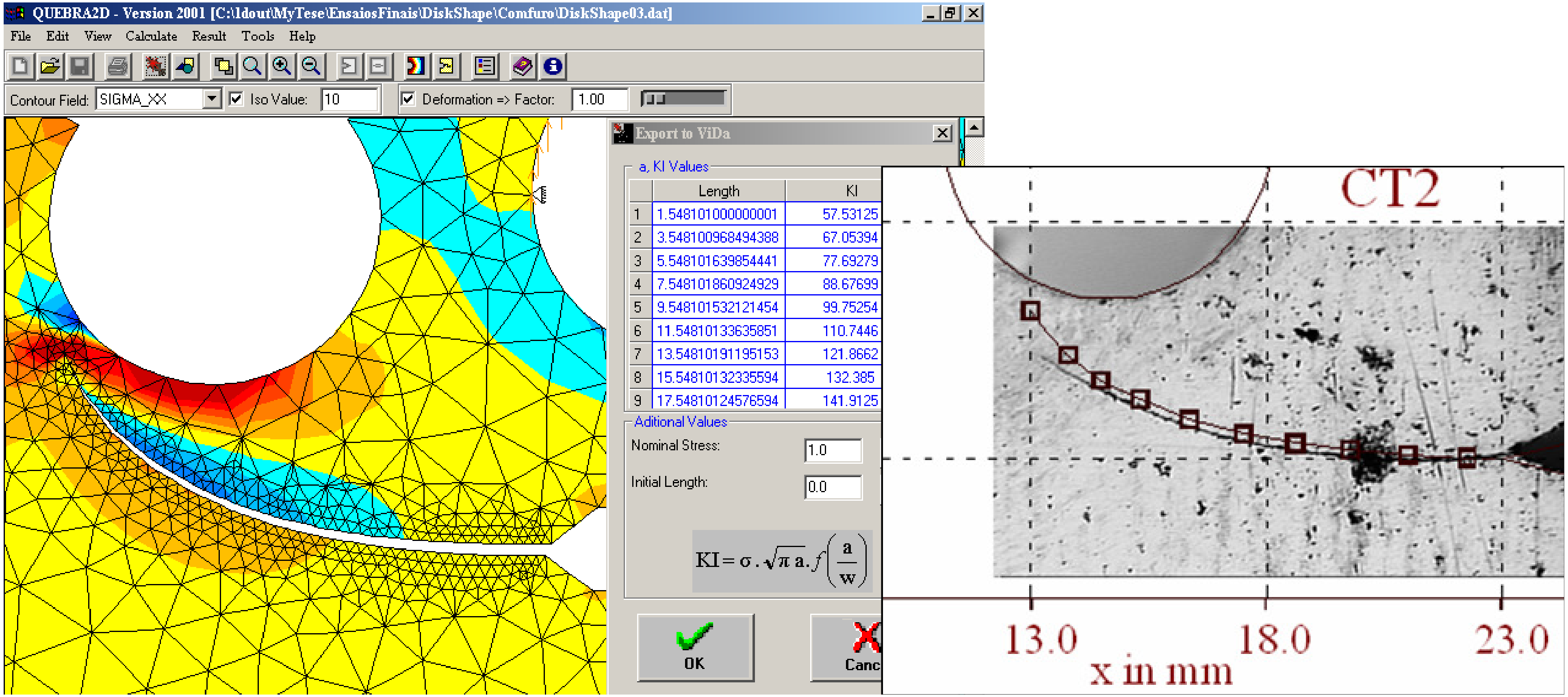
Requirements for underlying data representation (cont.)

– ***The data structure should aid in key aspects of geometric modeling, such as surface intersection and automatic region recognition, as well as in surface and solid finite element mesh generation in arbitrary domains.***



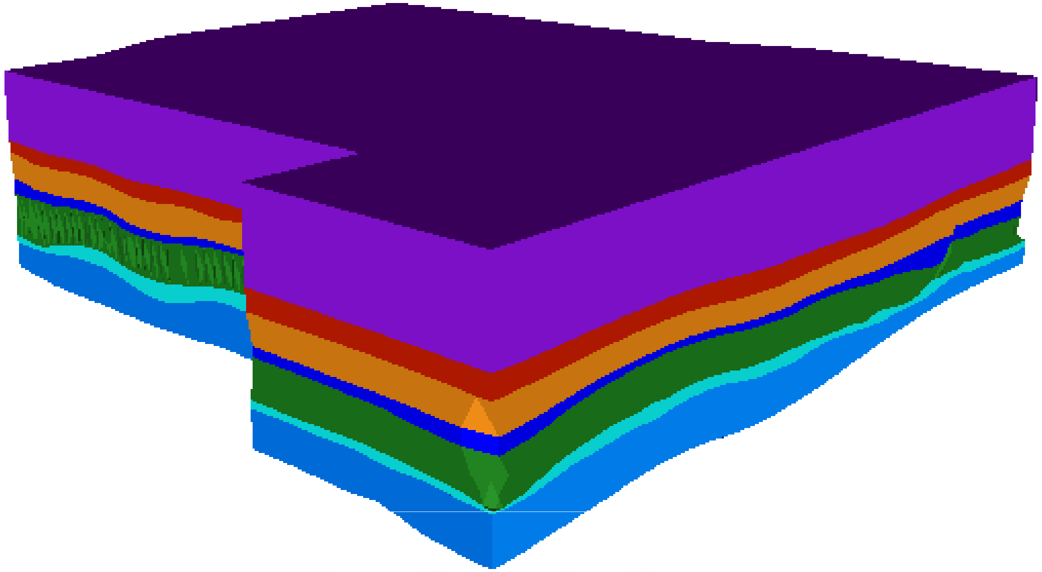
Requirements for underlying data representation (cont.)

– ***The data structure must provide for efficient geometric operators, including automatic intersection detection and processing. This is necessary in simulations with evolving topology and geometry.***

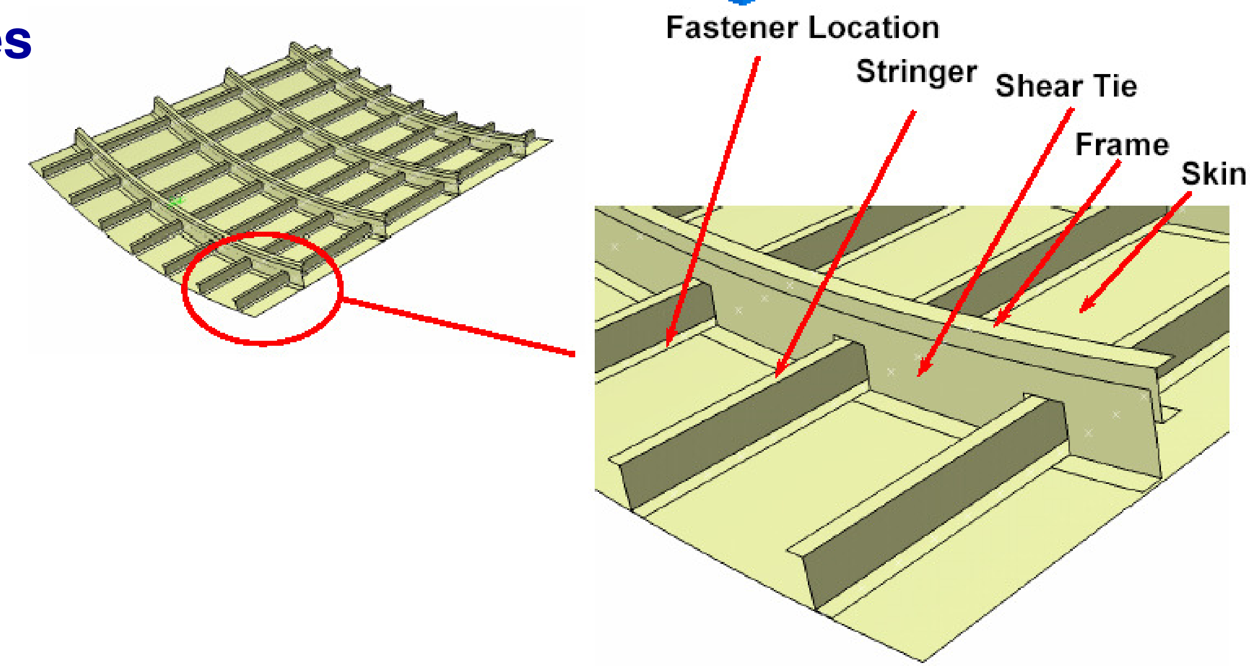


The need for non-manifold modeling

Multi-region modeling

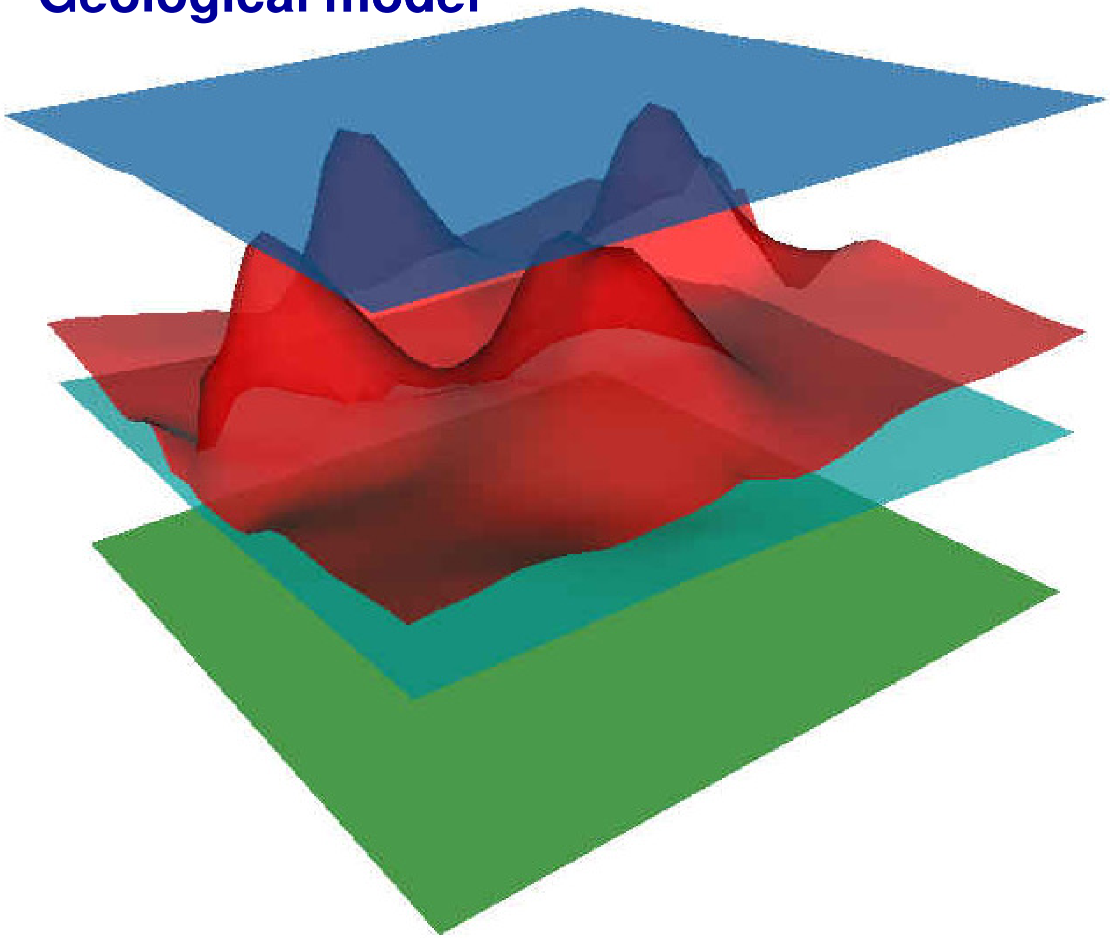


Degenerated structures

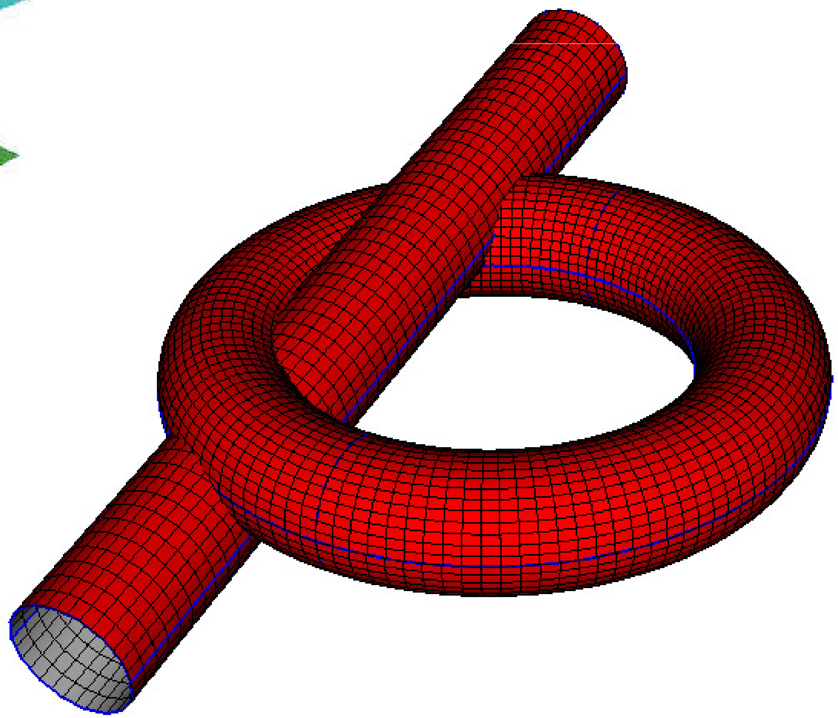


Natural modeling: surface patches as primitives

Geological model

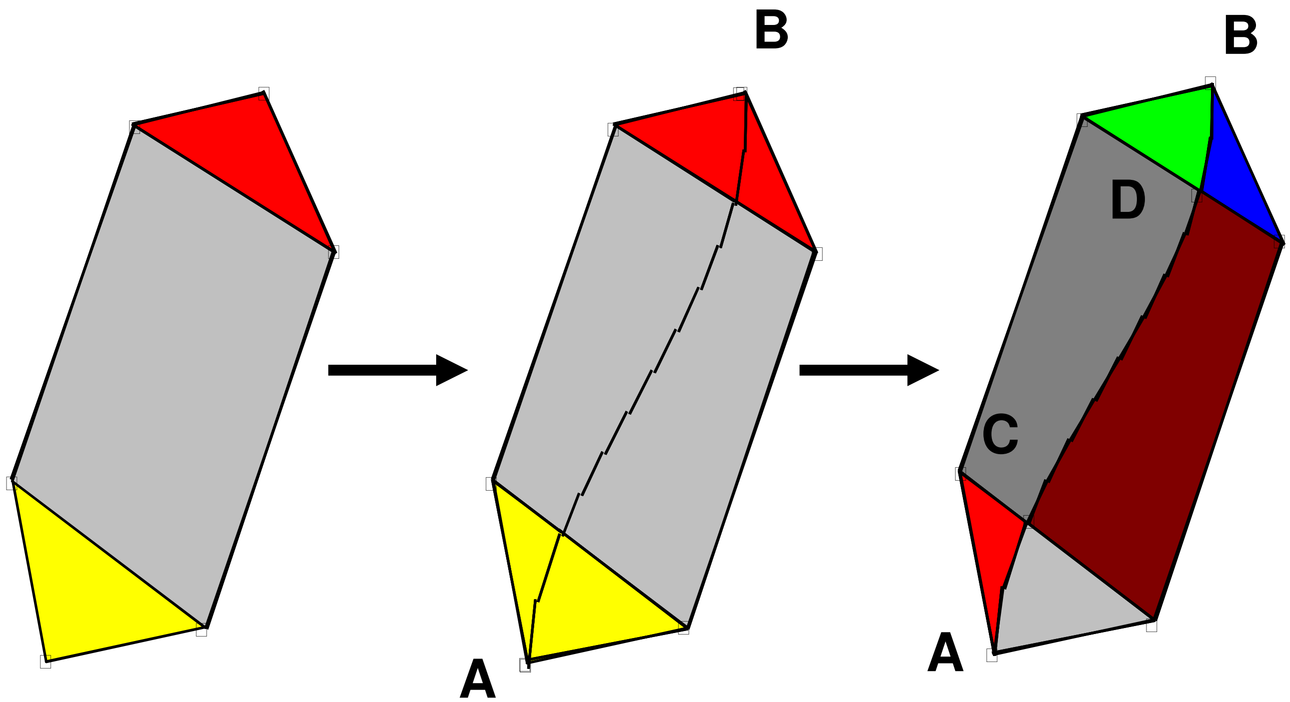


Manufactured model



Ideal environment: complete space subdivision

Space subdivision in 2D: high level operations



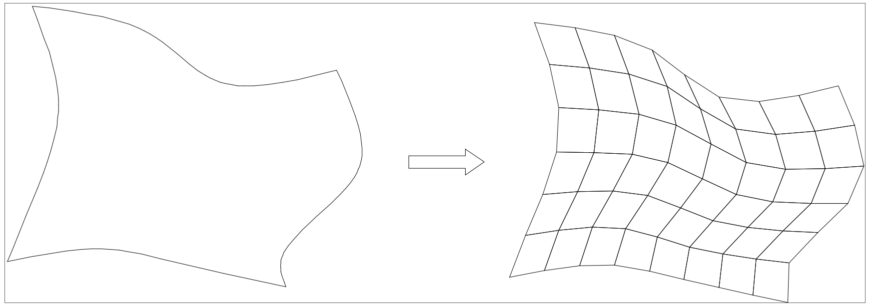
User action
+ basic function

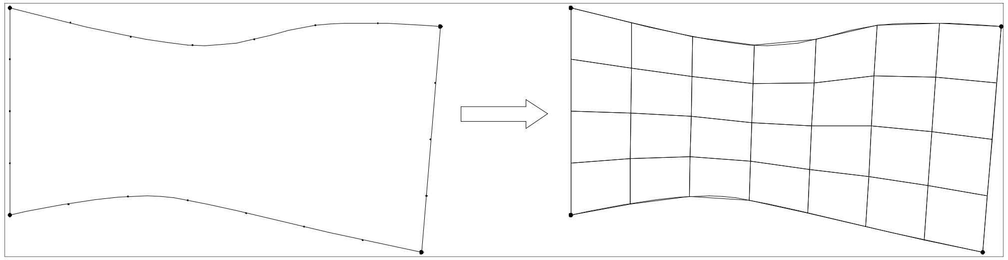
System
response

Mesh generation

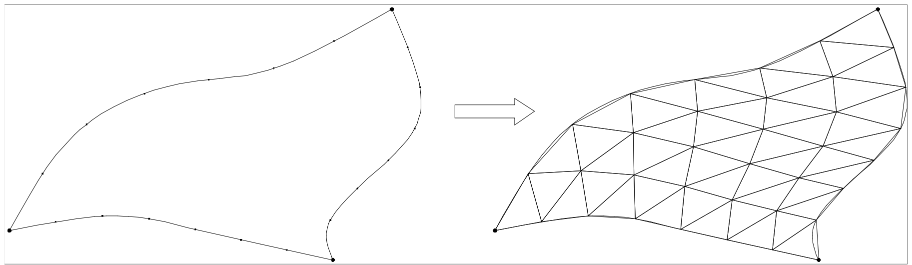
Library of mesh generation algorithms

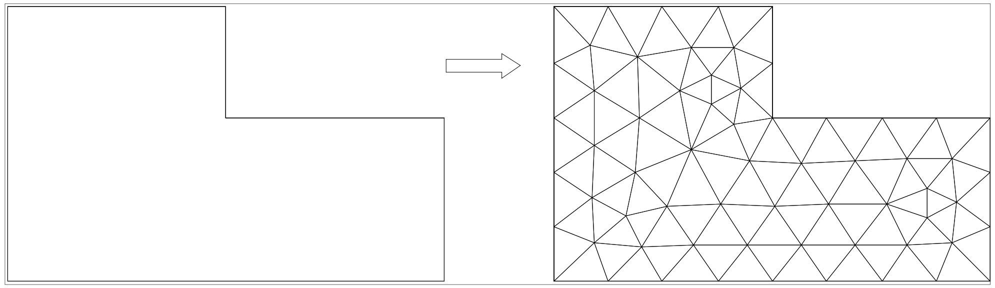
2D structured meshes





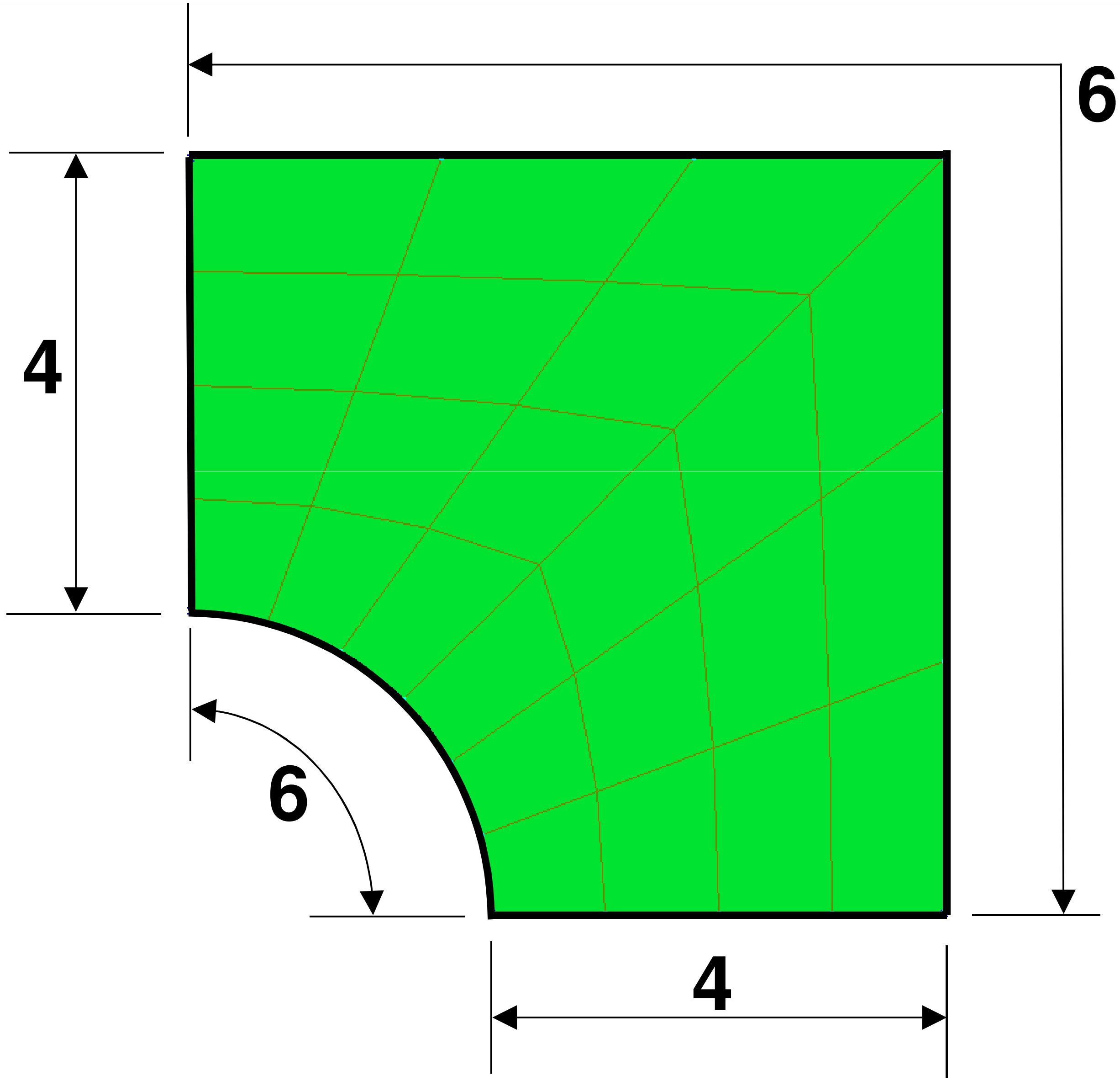
2D structured and non-structured meshes





Structured mesh – 2D Mapping

- **Geometry Requirements**
  - **4 topological sides**
  - **Opposite sides must have similar discretization**



Structured mesh – 2D Mapping

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 17, 1015–1044 (1981)

A GENERAL TWO-DIMENSIONAL, GRAPHICAL FINITE ELEMENT PREPROCESSOR UTILIZING DISCRETE TRANSFINITE MAPPINGS

ROBERT HABER‡
University of Illinois, Urbana, Illinois, U.S.A.

MARK S. SHEPHARD‡
Rensselaer Polytechnic Institute, Troy, New York, U.S.A.

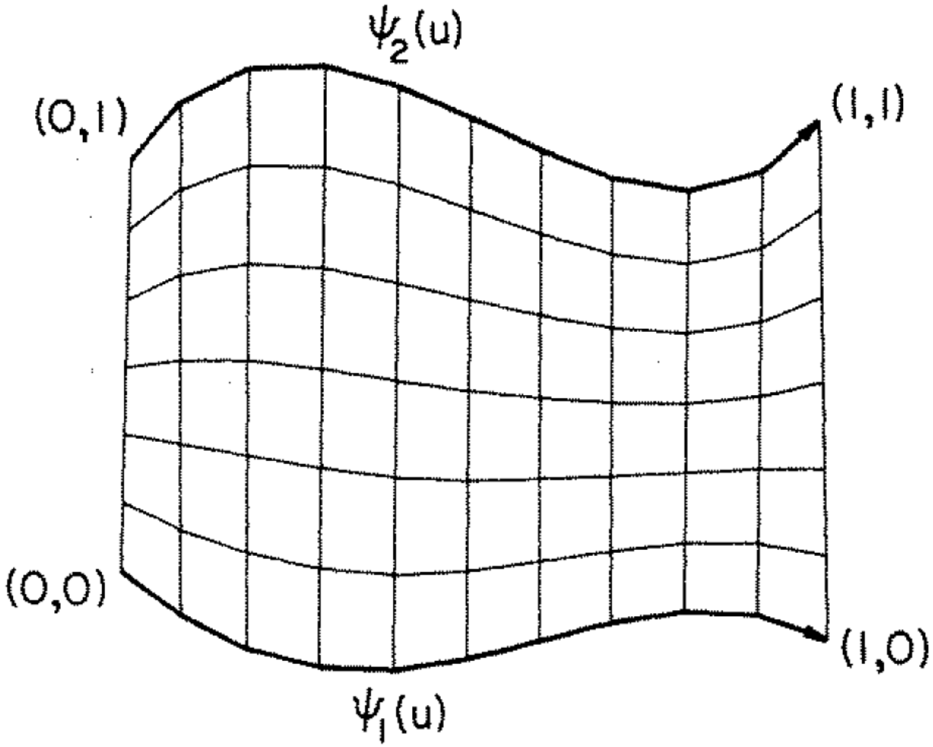
JOHN F. ABEL§
Cornell University, Ithaca, New York, U.S.A.

RICHARD H. GALLAGHER‖
University of Arizona, Tucson, Arizona, U.S.A.

AND

DONALD P. GREENBERG¶
Cornell University, Ithaca, New York, U.S.A.

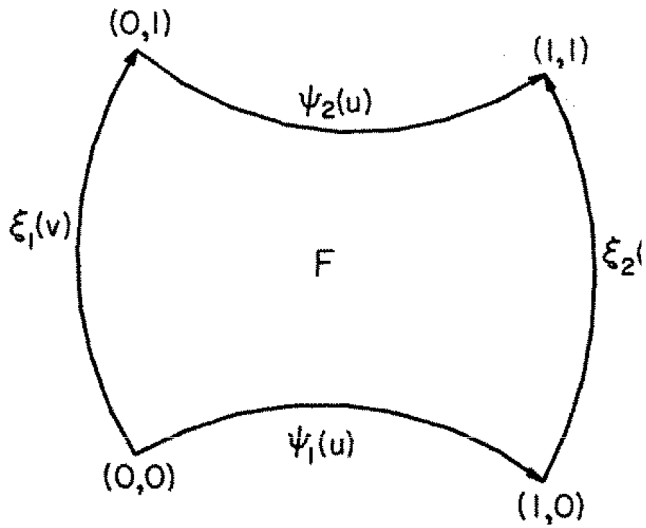
Structured mesh – 2D Mapping



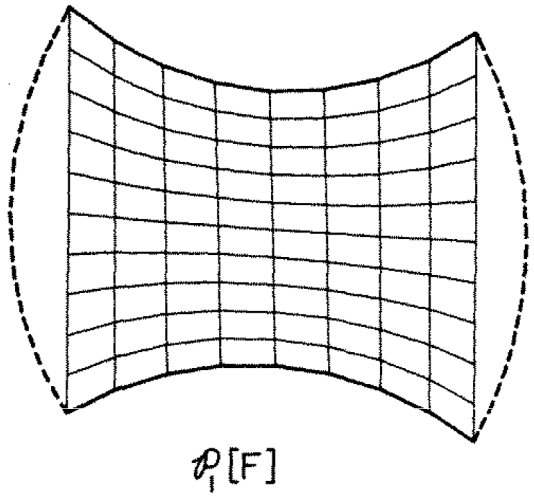
Lofting projector

$$\mathscr{P}_1[F] \equiv P_2(u, v) = (1 - v)\psi_1(u) + v\psi_2(u) \qquad 0 \leqslant u \leqslant 1$$

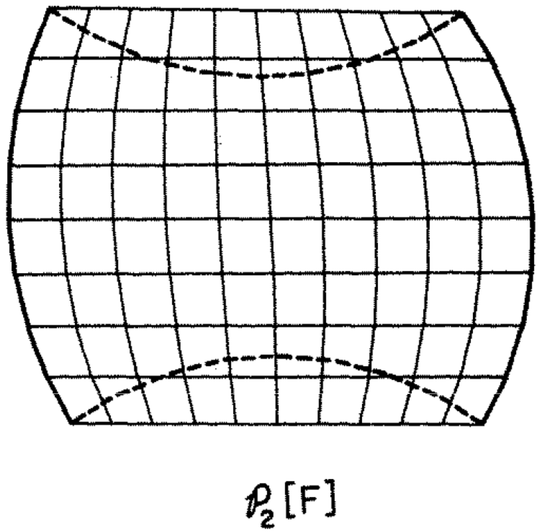
Structured mesh – 2D Mapping



Bilinear projector: co-ordinate system and boundary curves



Bilinear projector: $\mathscr{P}_1$

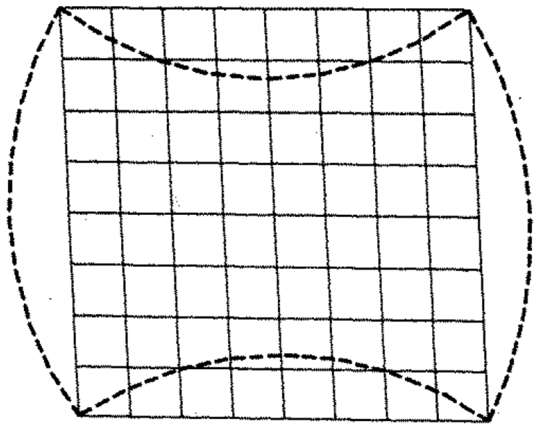


Bilinear projector: $\mathscr{P}_2$

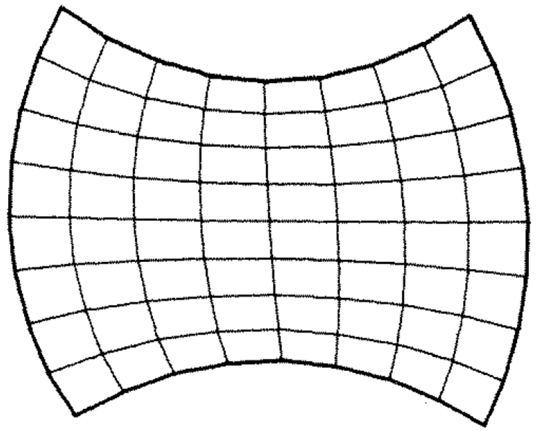
$$\mathscr{P}_1[F] \equiv P_2(u, v) = (1-v)\psi_1(u) + v\psi_2(u) \qquad 0 \leqslant u \leqslant 1$$

$$\mathscr{P}_2[F] \equiv P_2(u, v) = (1-u)\xi_1(v) + u\xi_2(v) \qquad 0 \leqslant v \leqslant 1$$

Structured mesh – 2D Mapping



Bilinear projector: $\mathcal{P}_1\mathcal{P}_2$



Bilinear projector: $\mathcal{P}_1 \oplus \mathcal{P}_2$

$$(\mathcal{P}_1 \oplus \mathcal{P}_2)[F] \equiv \mathcal{P}_1[F] + \mathcal{P}_2[F] - \mathcal{P}_1\mathcal{P}_2[F]$$

$$= P_{\mathrm{B}}(u, v)$$

$$= (1-v)\psi_1(u) + v\psi_2(u) + (1-u)\xi_1(v) + u\xi_2(v)$$

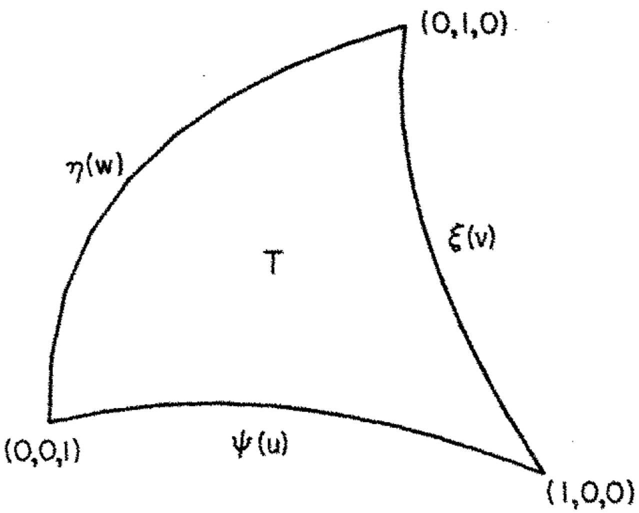
$$-(1-u)(1-v)F(0,0) - (1-u)vF(0,1)$$

$$-uvF(1,1) - u(1-v)F(1,0) \qquad 0 \leqslant u \leqslant 1, 0 \leqslant v \leqslant 1$$

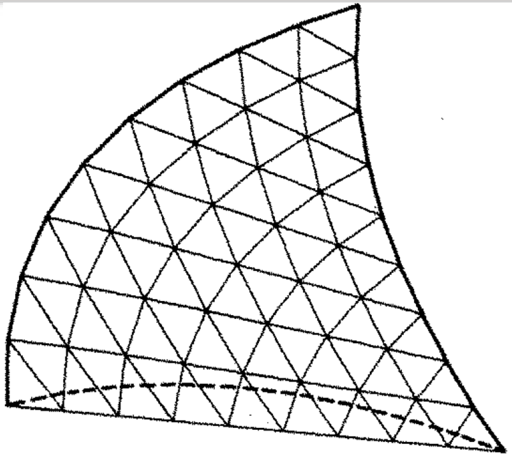
Assumed discrete representation of curves:

$$\{\xi_1(v_i), \xi_2(v_i)\} i = 1, n, \qquad \{\psi_1(u_j), \psi_2(u_j)\} j = 1, m$$

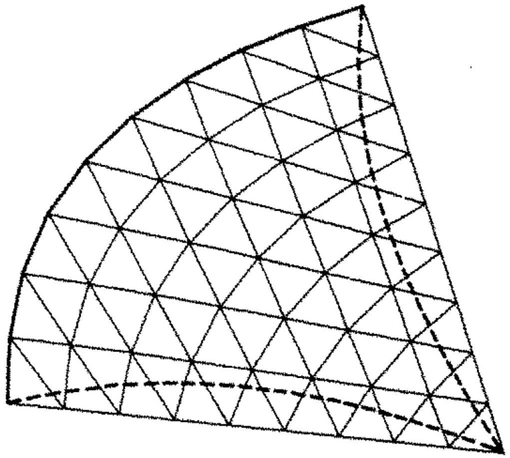
Structured mesh – 2D Mapping



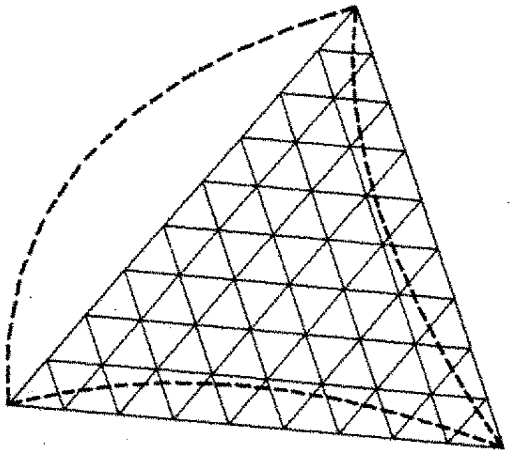
Trilinear projector: co-ordinate system and boundary curves



Trilinear projector: $\mathcal{N}_1$



Trilinear projector: $\mathcal{N}_1\mathcal{N}_2$



Trilinear projector: $\mathcal{N}_1\mathcal{N}_2\mathcal{N}_3$

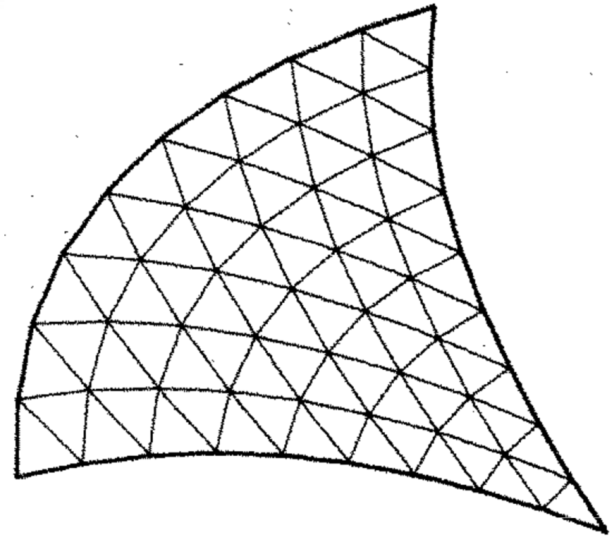
$$\mathcal{N}_1 \equiv N_1(u, v, w) = \left(\frac{u}{1-v}\right)\xi(v) + \left(\frac{w}{1-v}\right)\eta(1-v)$$

$$\mathcal{N}_2 \equiv N_2(u, v, w) = \left(\frac{v}{1-w}\right)\eta(w) + \left(\frac{u}{1-w}\right)\psi(1-w)$$

$$\mathcal{N}_3 \equiv N_3(u, v, w) = \left(\frac{w}{1-u}\right)\psi(u) + \left(\frac{v}{1-u}\right)\xi(1-u)$$

$$0 \leqslant u \leqslant 1, \quad 0 \leqslant v \leqslant 1, \quad 0 \leqslant w \leqslant 1, \quad u+v+w=1$$

Structured mesh – 2D Mapping



Q [T]

Trilinear projector: $\mathcal{Q}$

$$\mathcal{Q} \equiv Q(u, v, w) = \frac{1}{2}\Big[\Big(\frac{u}{1-v}\Big)\xi(v) + \Big(\frac{w}{1-v}\Big)\eta(1-v) + \Big(\frac{v}{1-w}\Big)\eta(w) + \Big(\frac{u}{1-w}\Big)\psi(1-w)$$

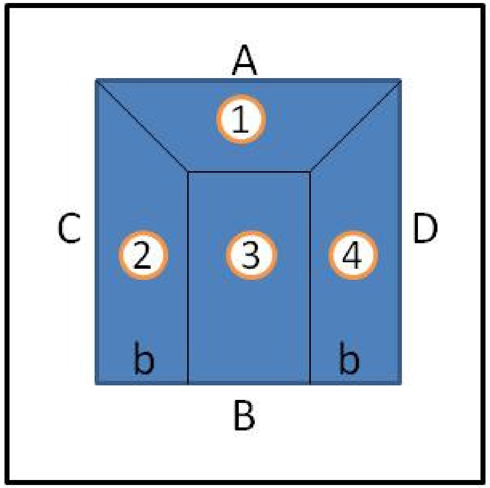
$$+\Big(\frac{w}{1-u}\Big)\psi(u) + \Big(\frac{v}{1-u}\Big)\xi(1-u) - w\psi(0) - u\xi(0) - v\eta(0)\Big]$$

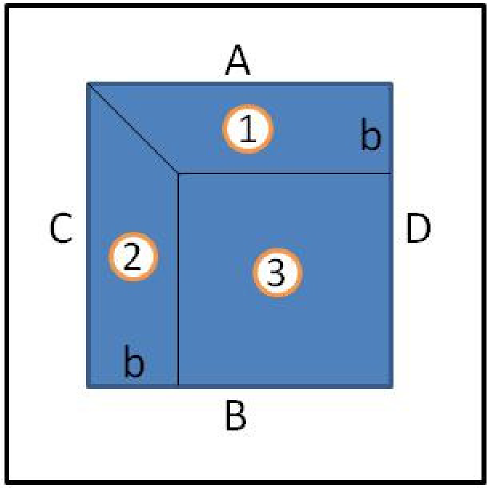
Assumed discrete representation of curves:

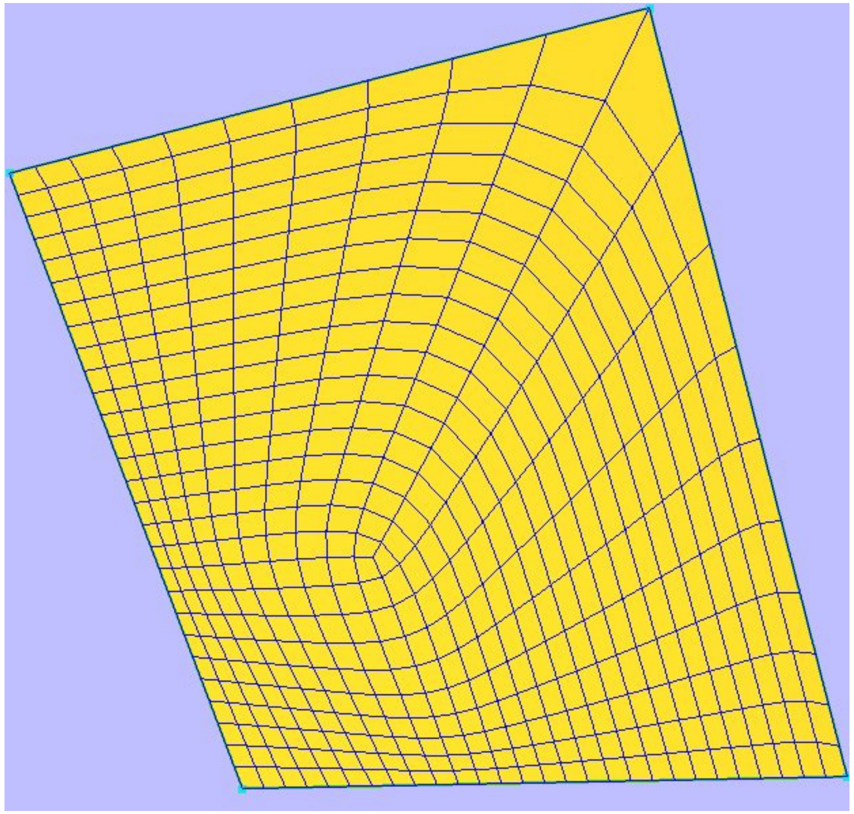
$$\{\psi(u_i), \xi(v_i), \eta(w_i); i = 1, n\}$$

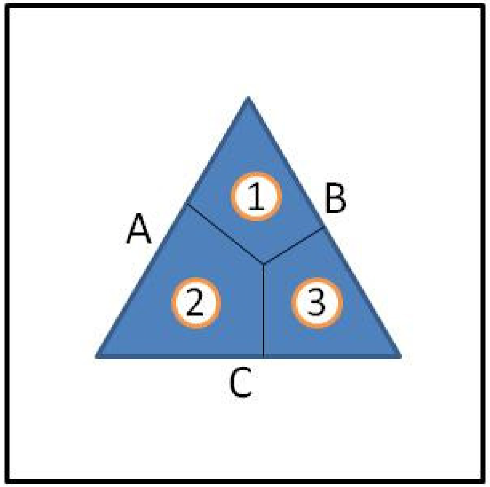
Library of mesh generation algorithms

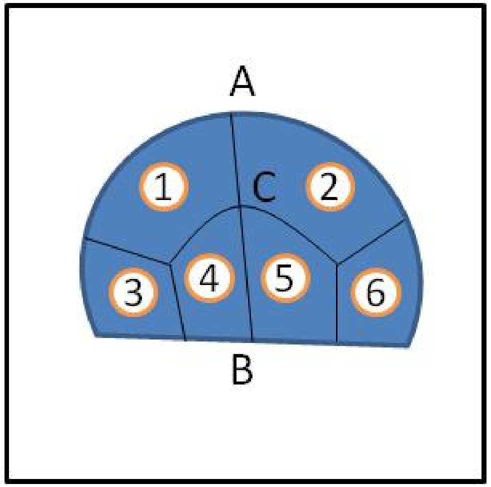
Quadrilateral template (new)





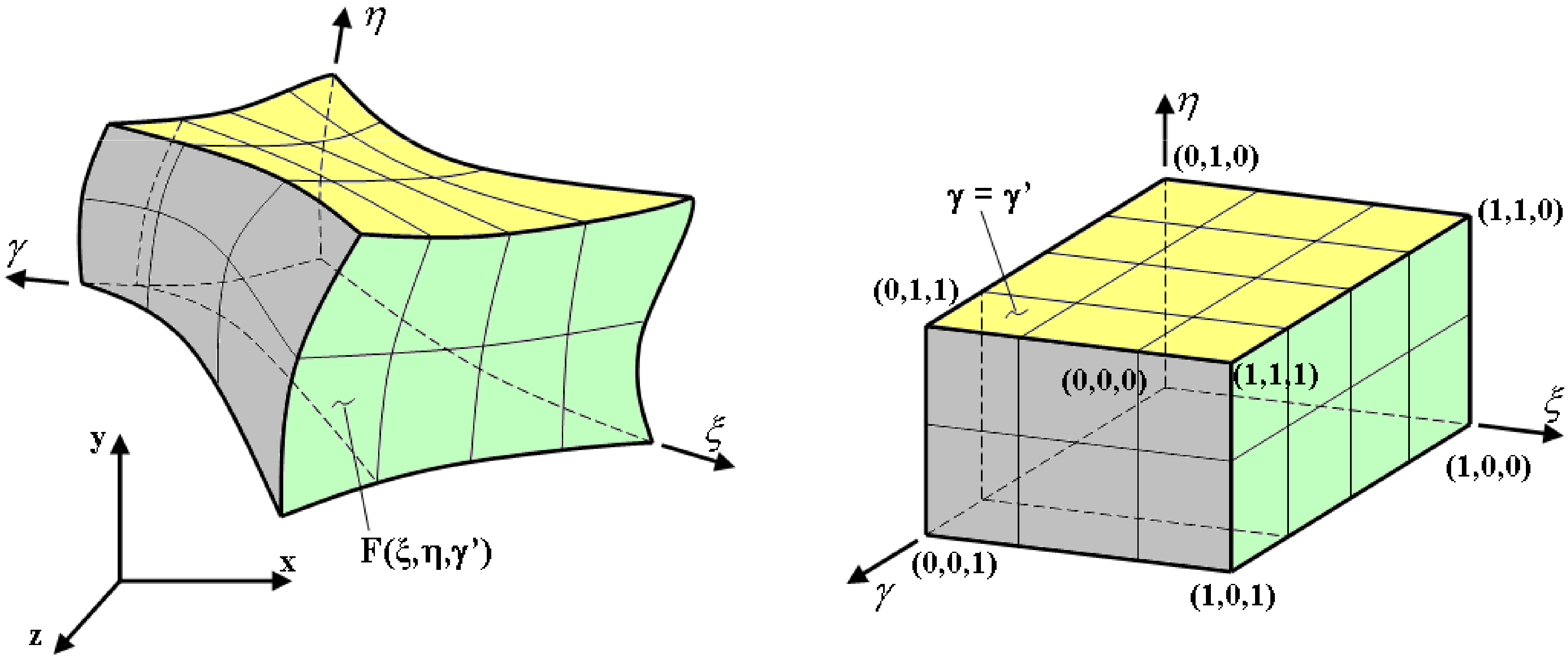






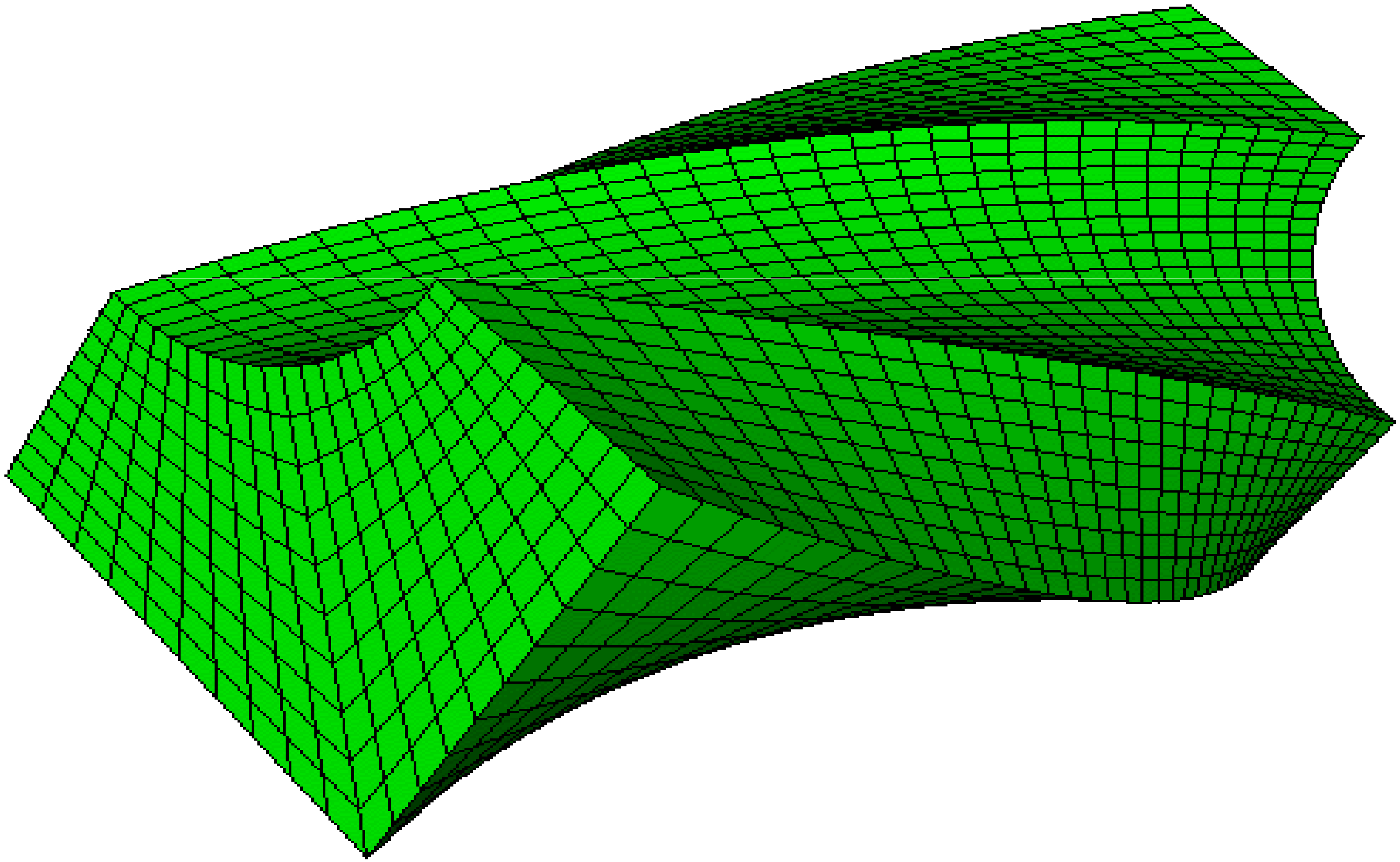
Structured mesh – 3D Mapping

- **Geometry Requirements**
  - **6 topological surfaces**
  - **Opposite surfaces must have similar mapped meshes**



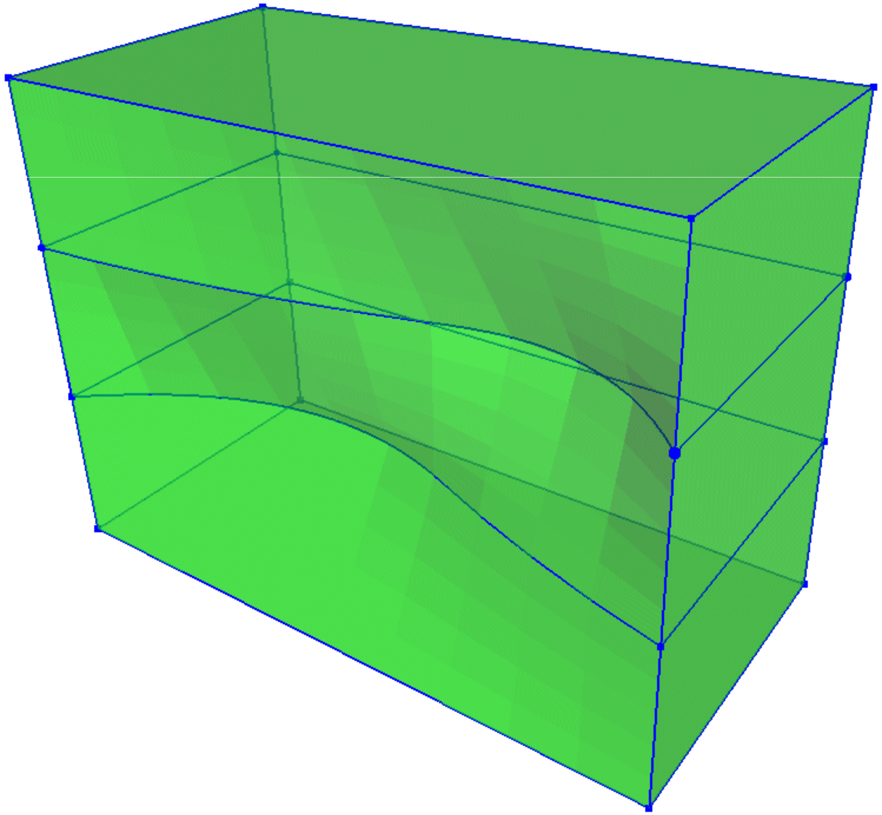
Structured mesh – 3D Mapping

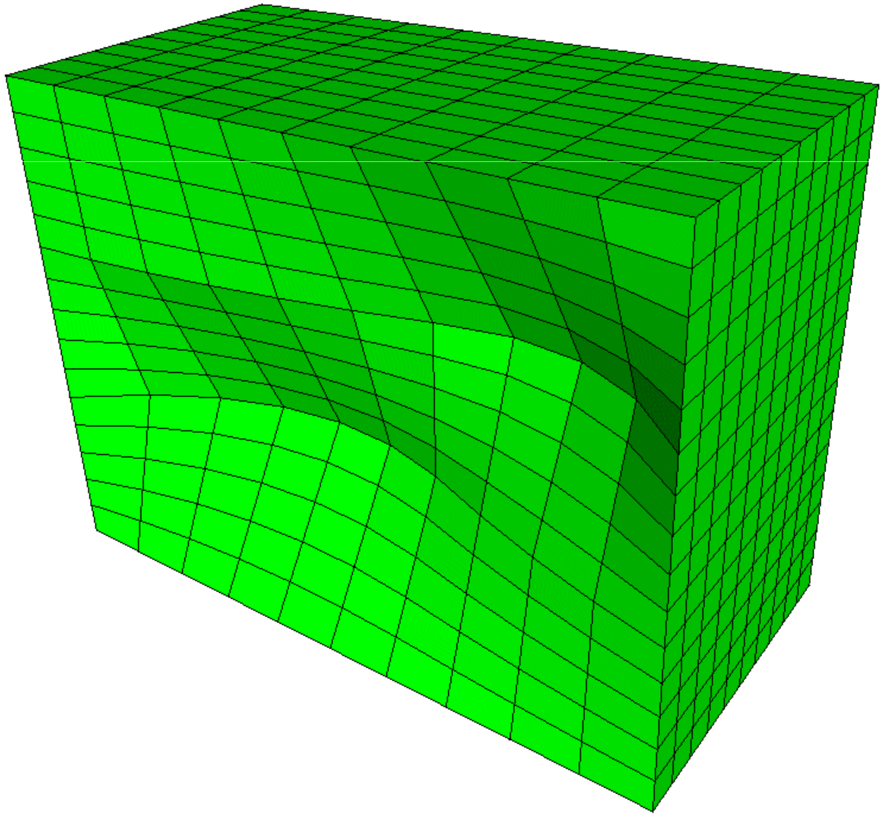
- **Many complex domains can be mapped**



Structured mesh – 3D Mapping

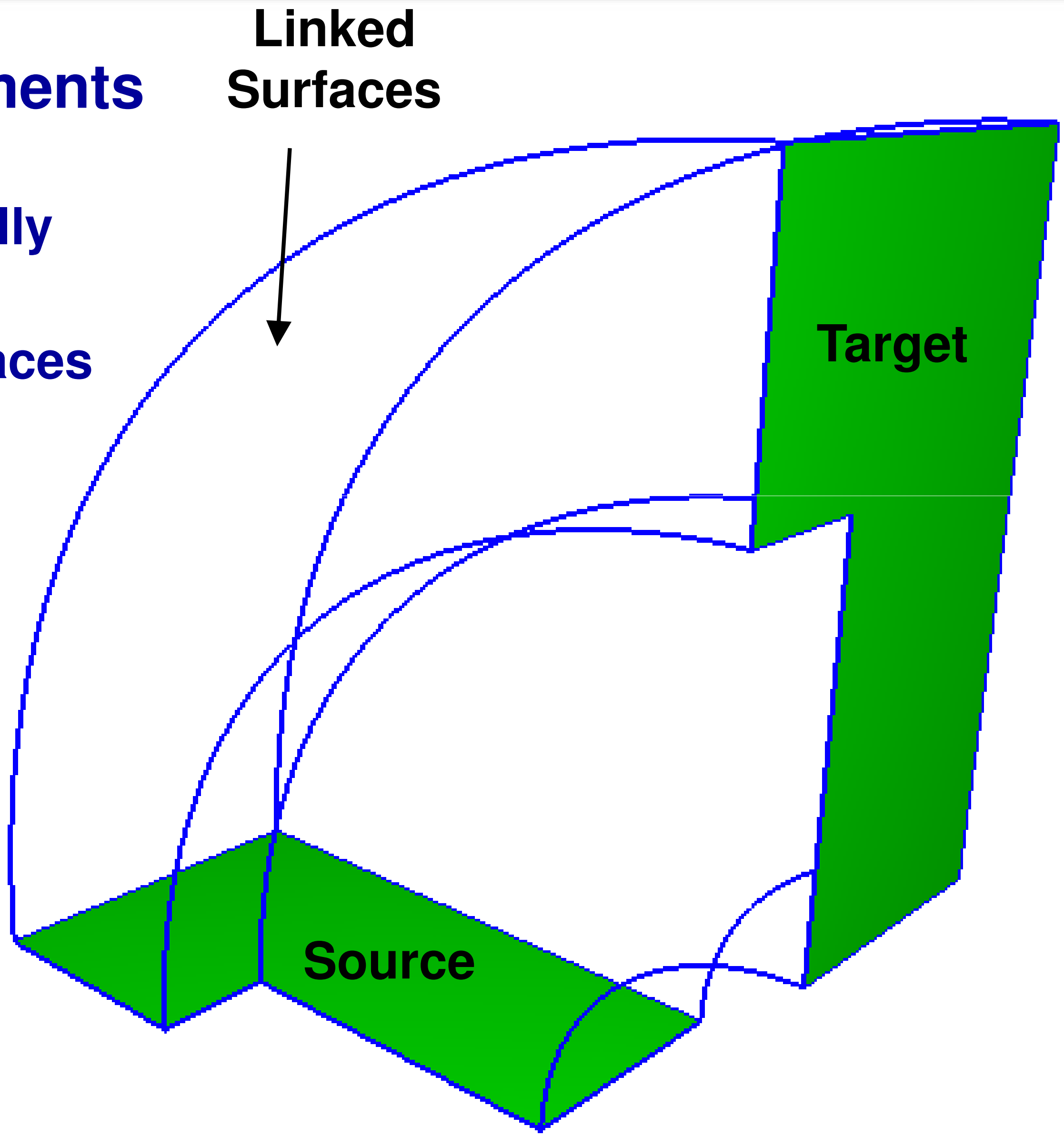
- **Algorithm must deal with:**
  - **Multiple surfaces on boundary**
  - **Concave surfaces**





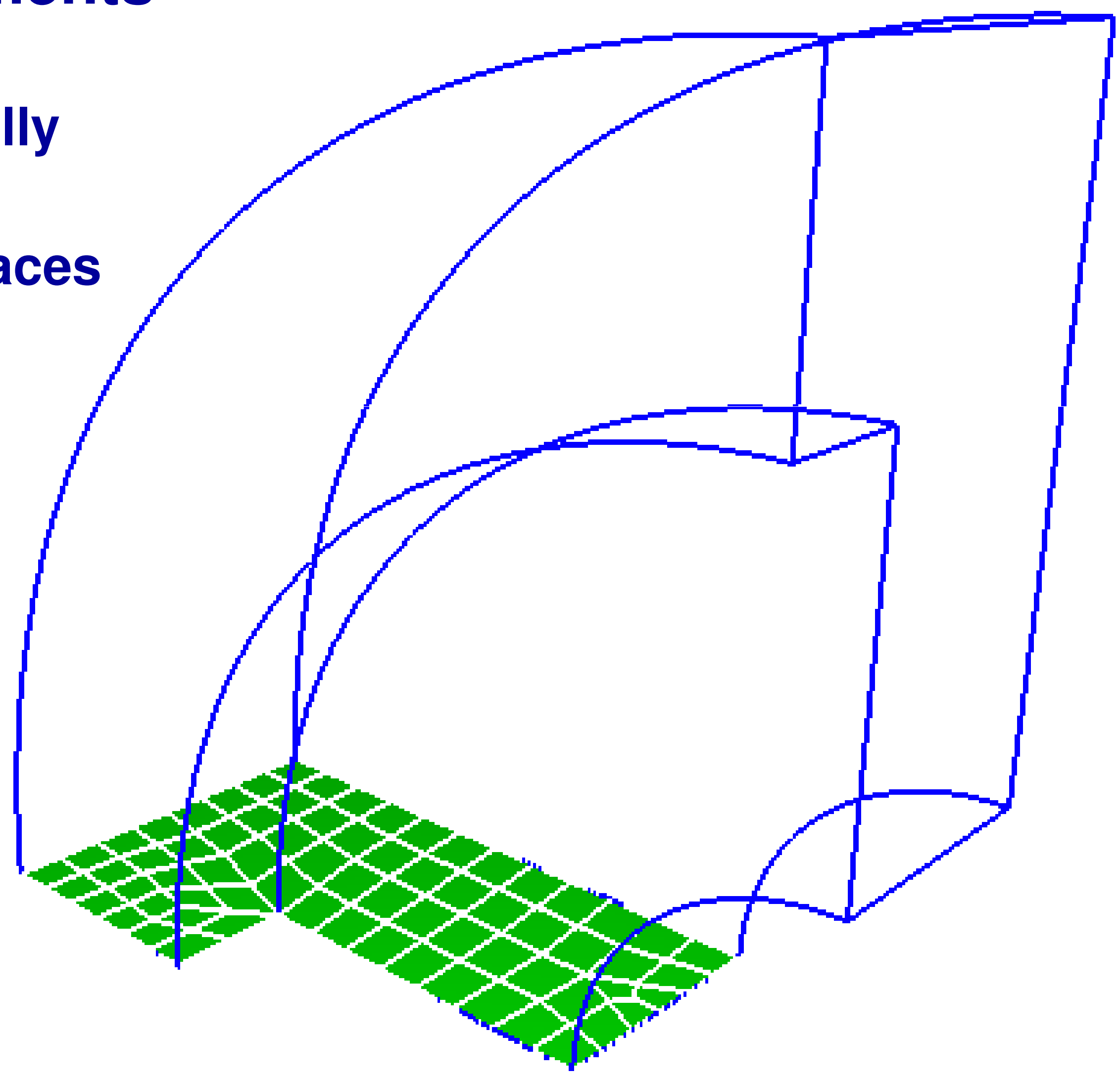
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



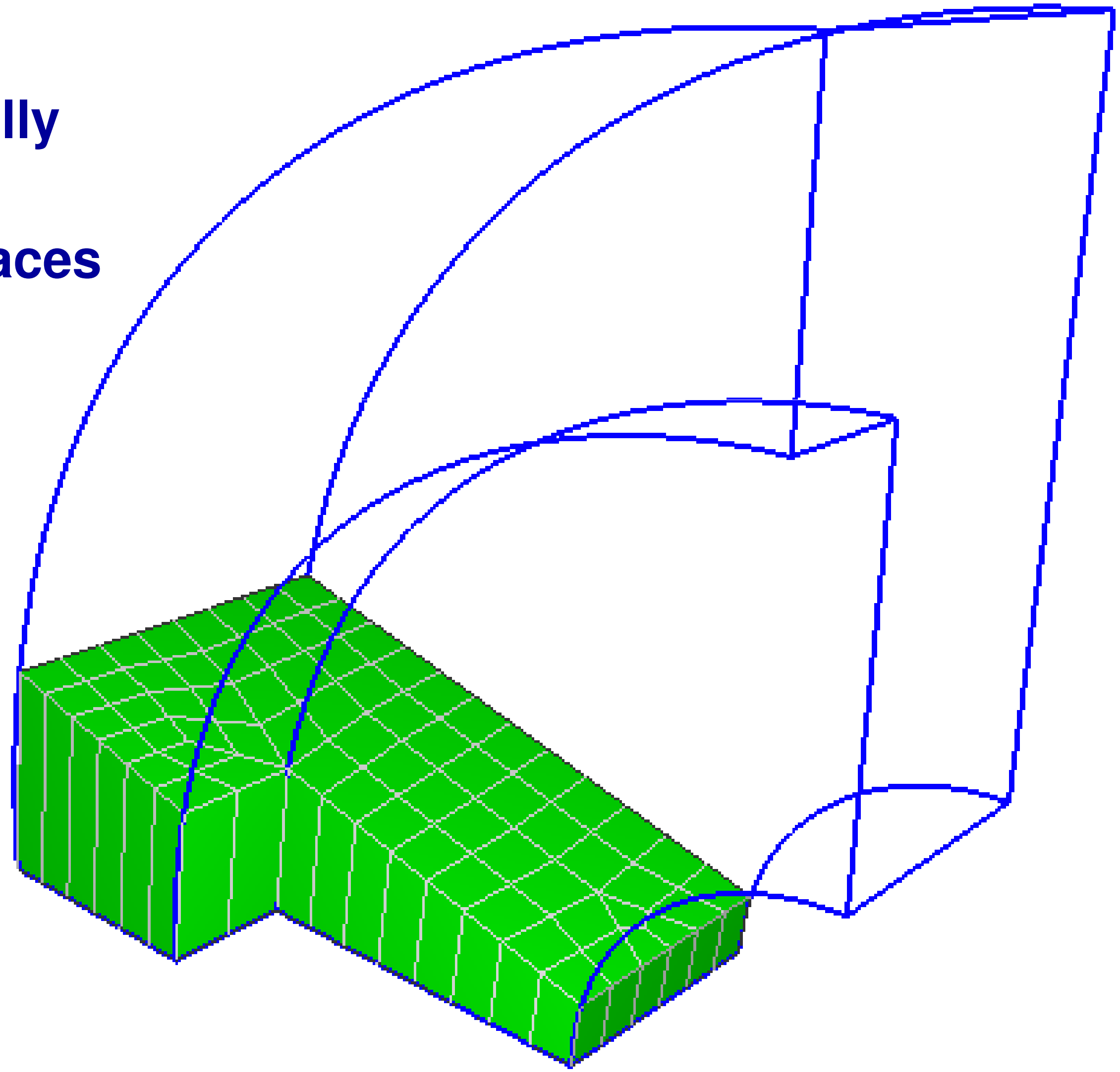
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



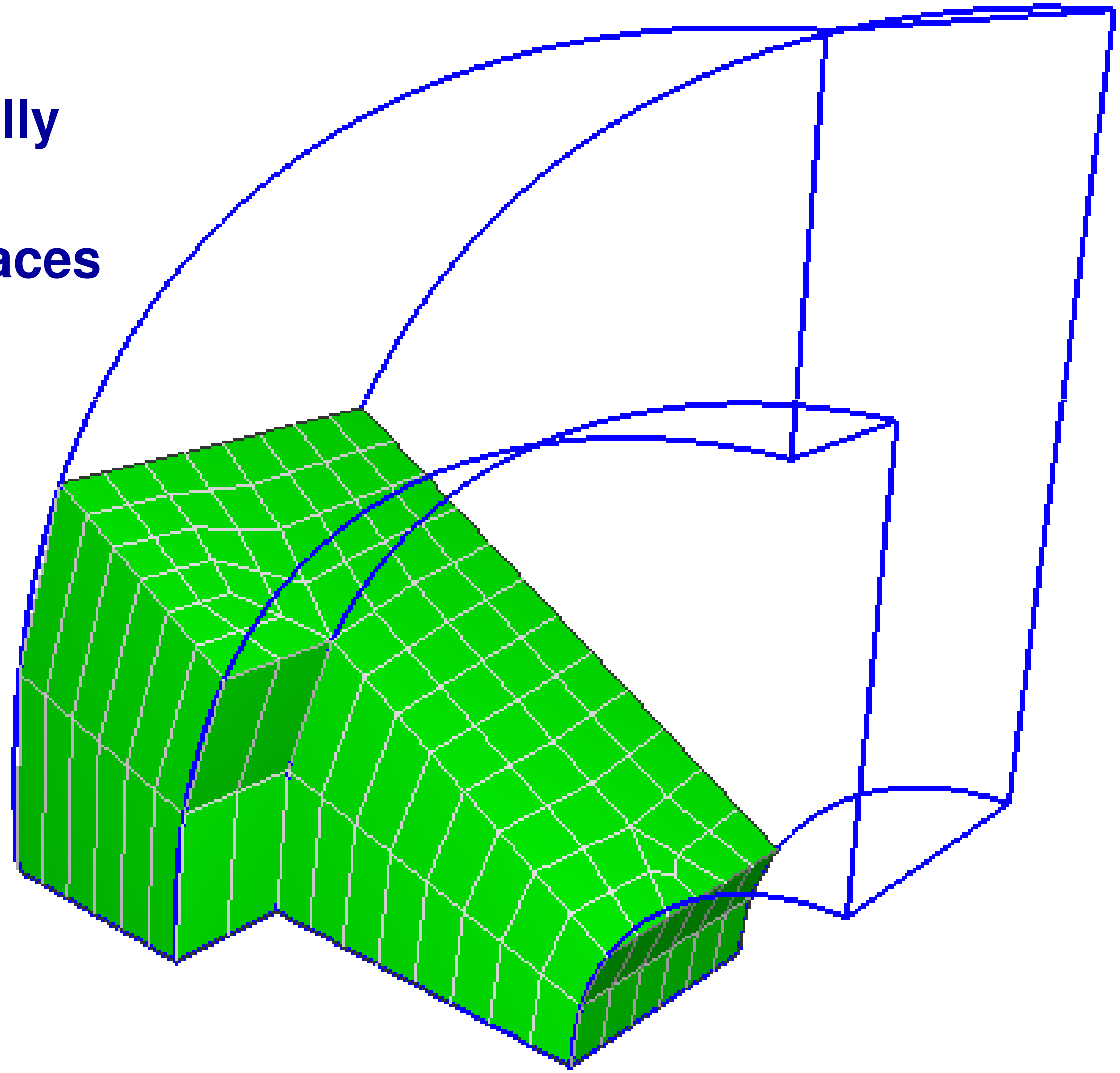
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



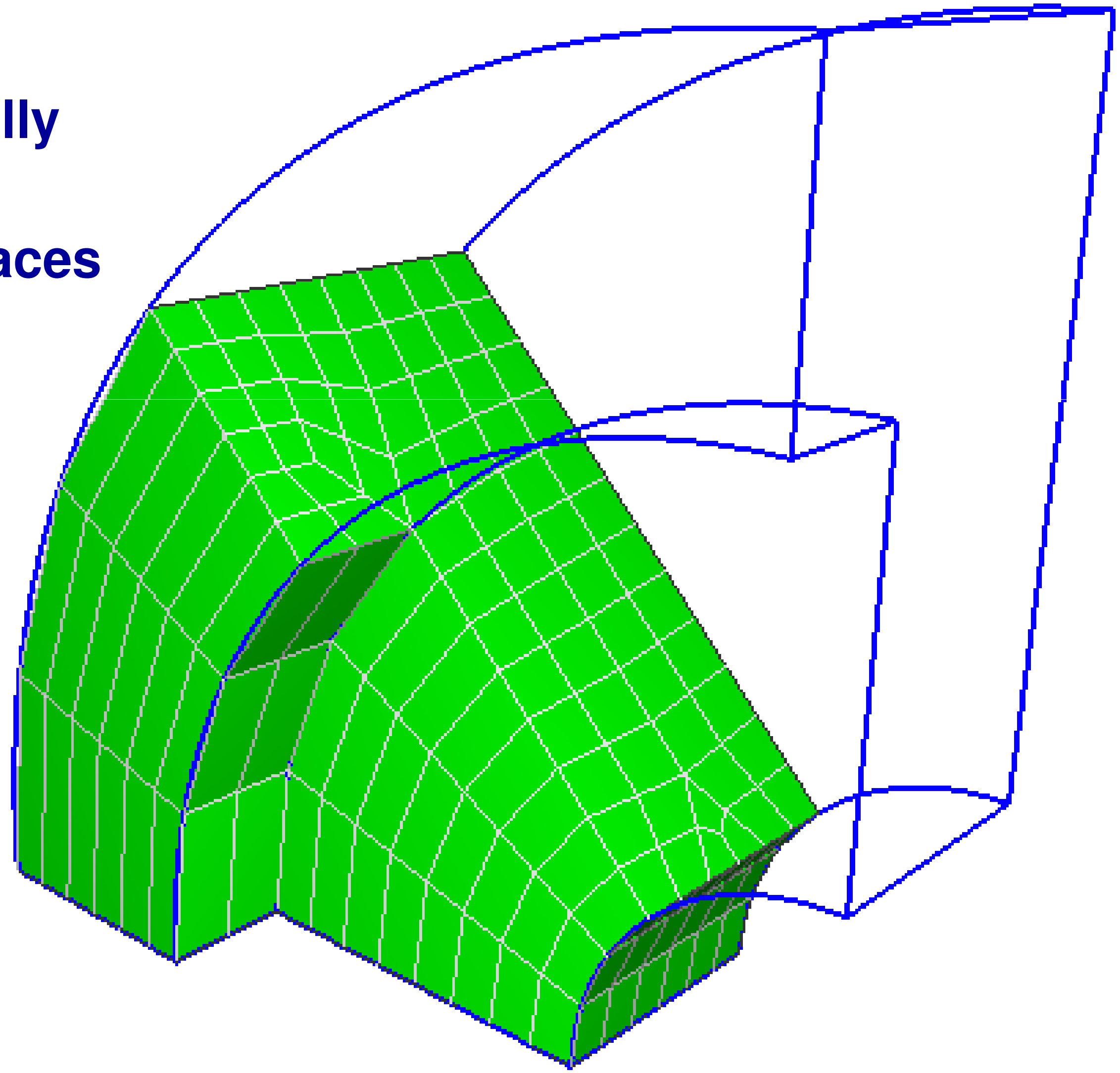
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



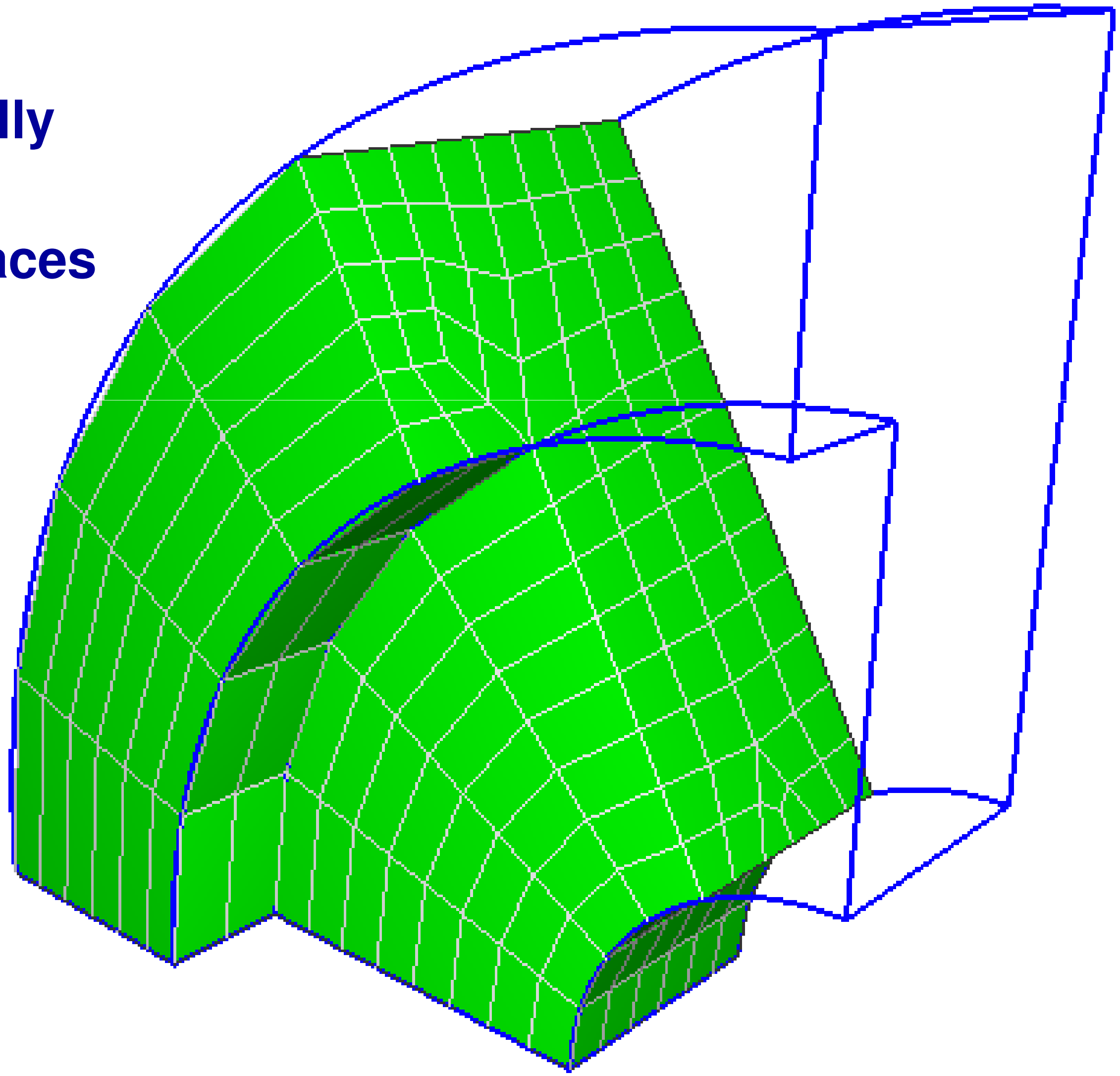
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



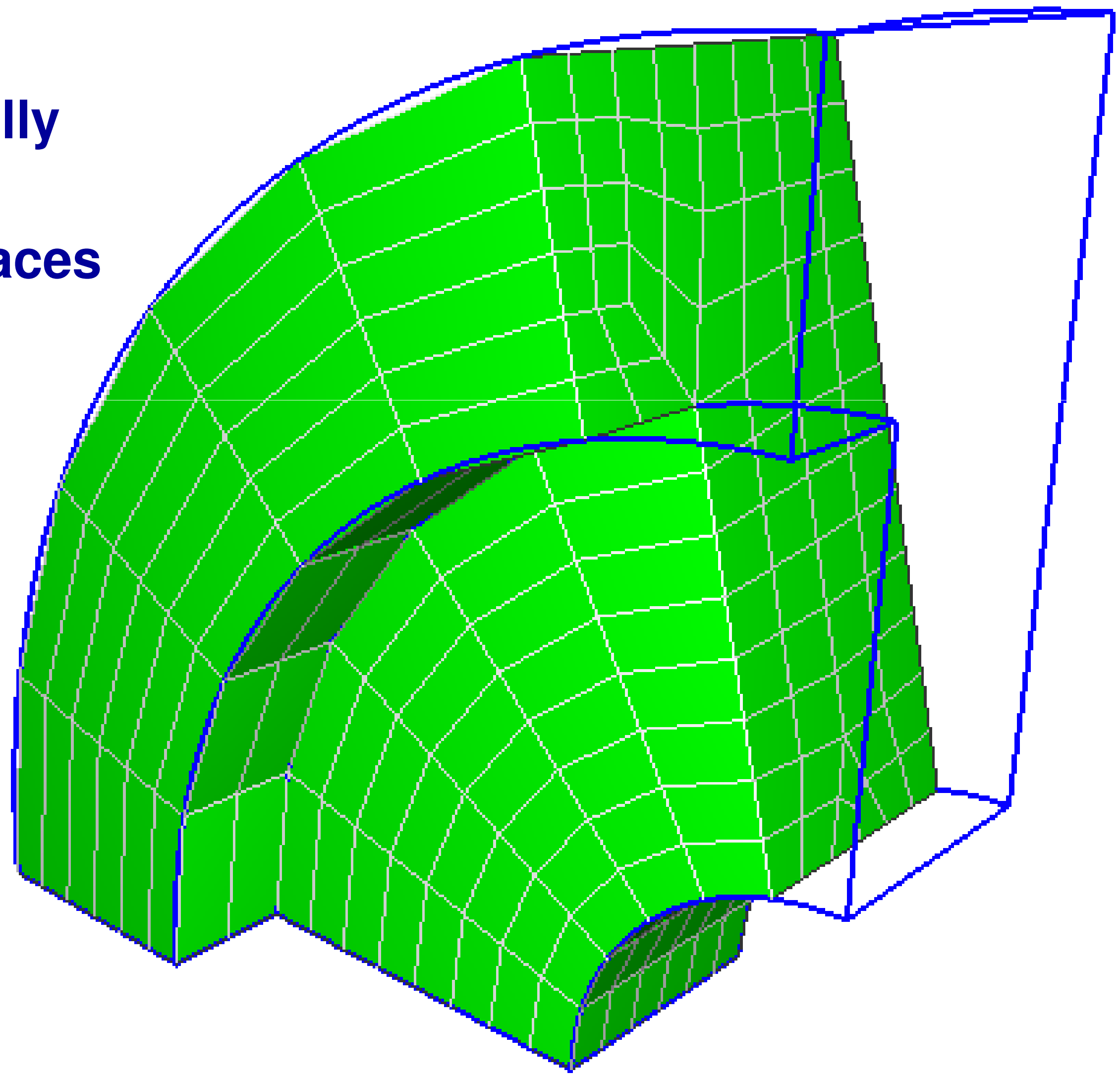
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



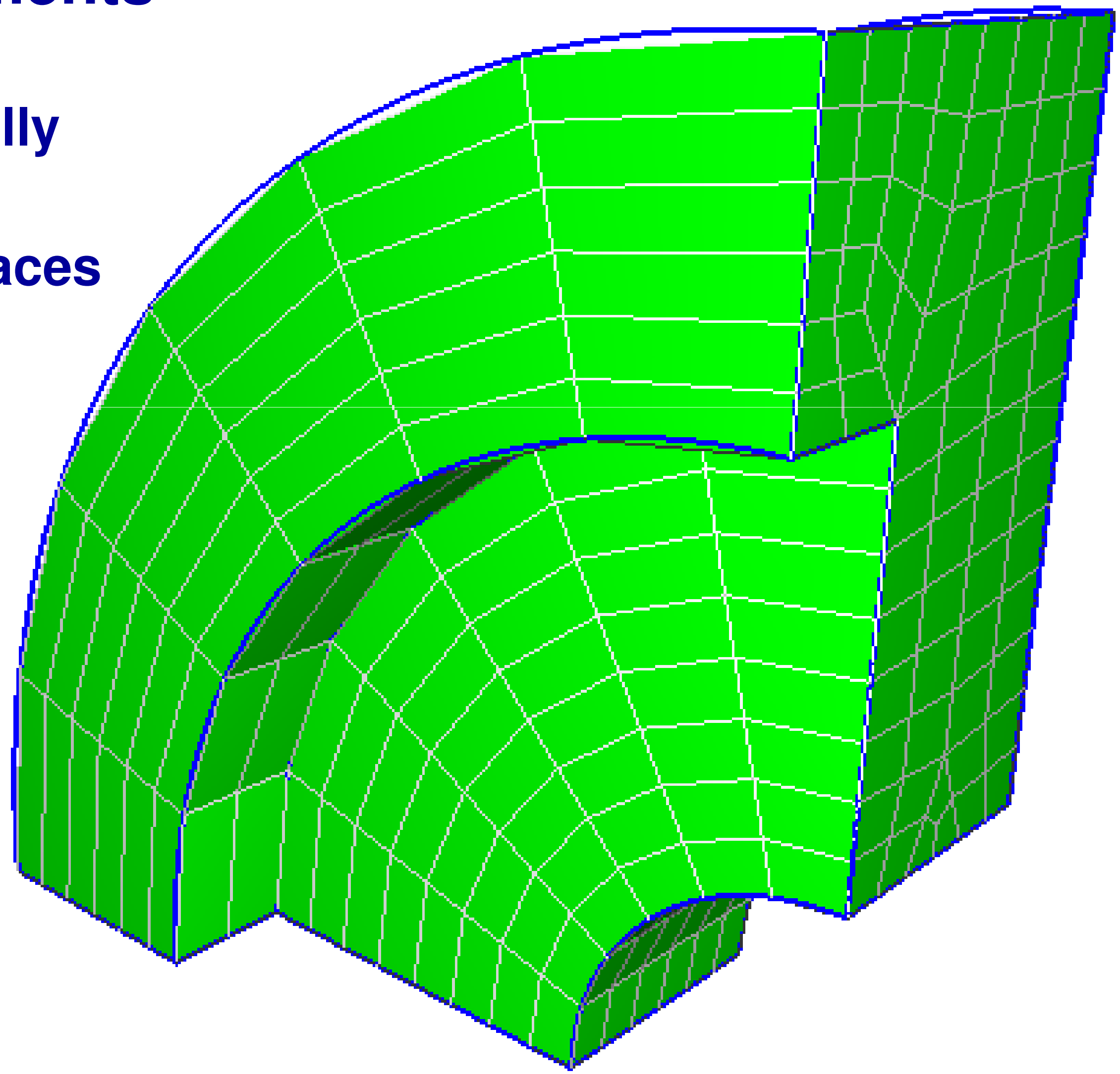
Structured mesh – Sweeping

- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**



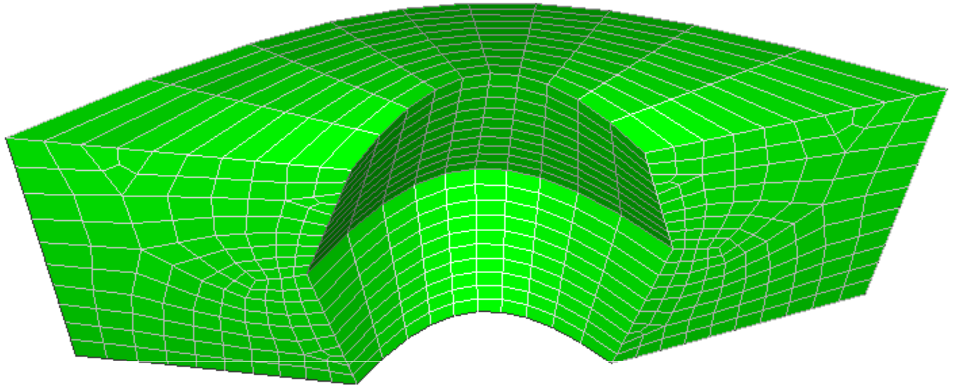
Structured mesh – Sweeping

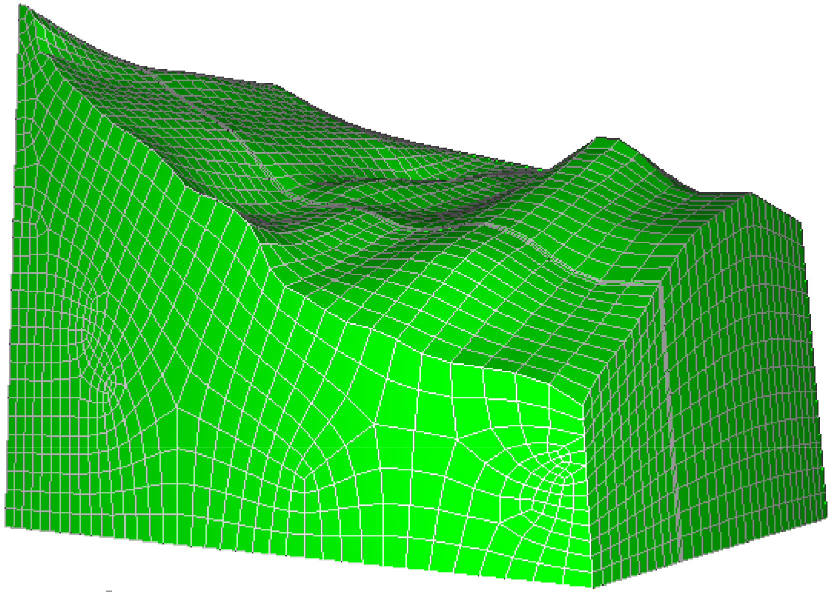
- **Geometry Requirements**
  - **Source and target surfaces topologically similar**
  - **Mapped linked surfaces**

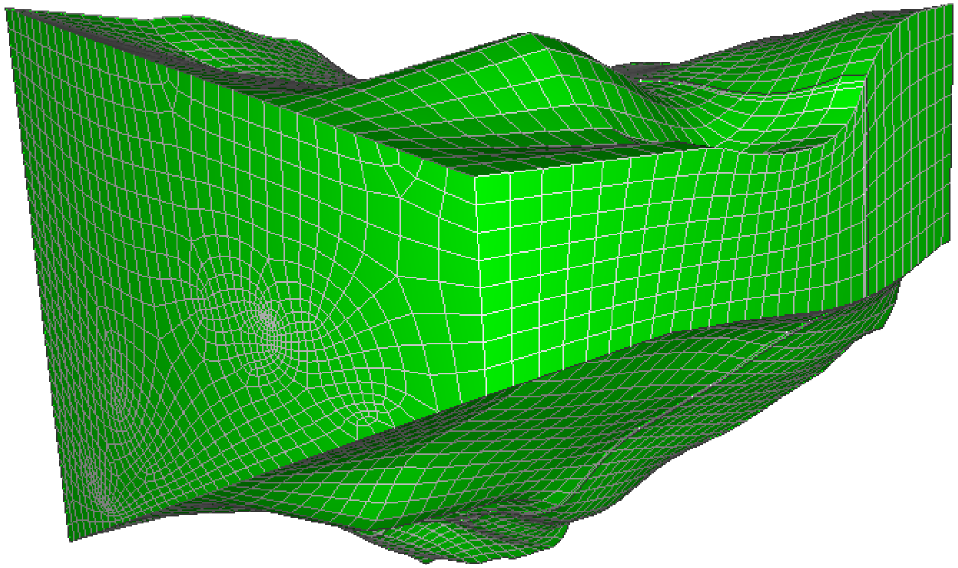


Structured mesh – Sweeping

- **Examples**

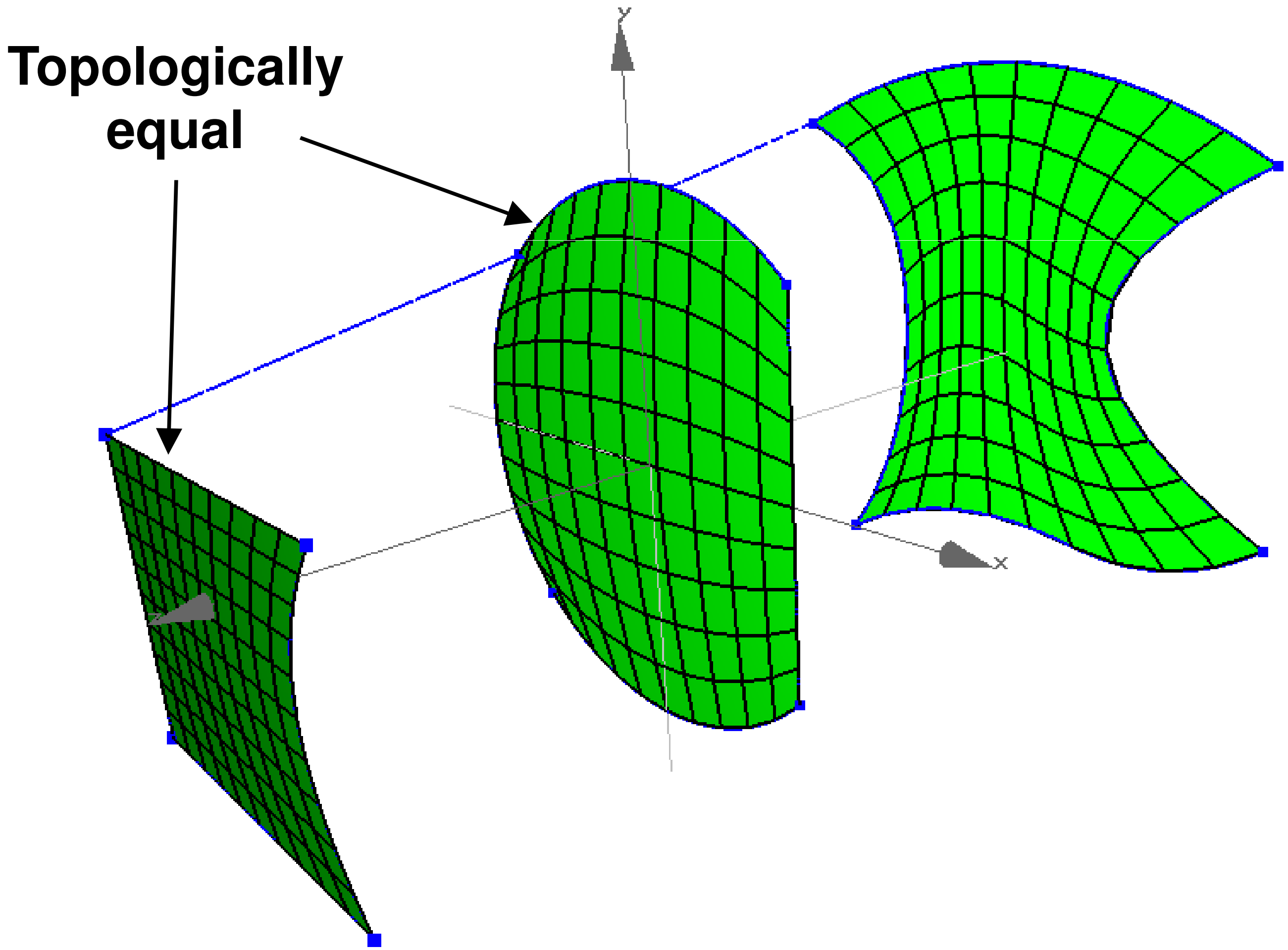






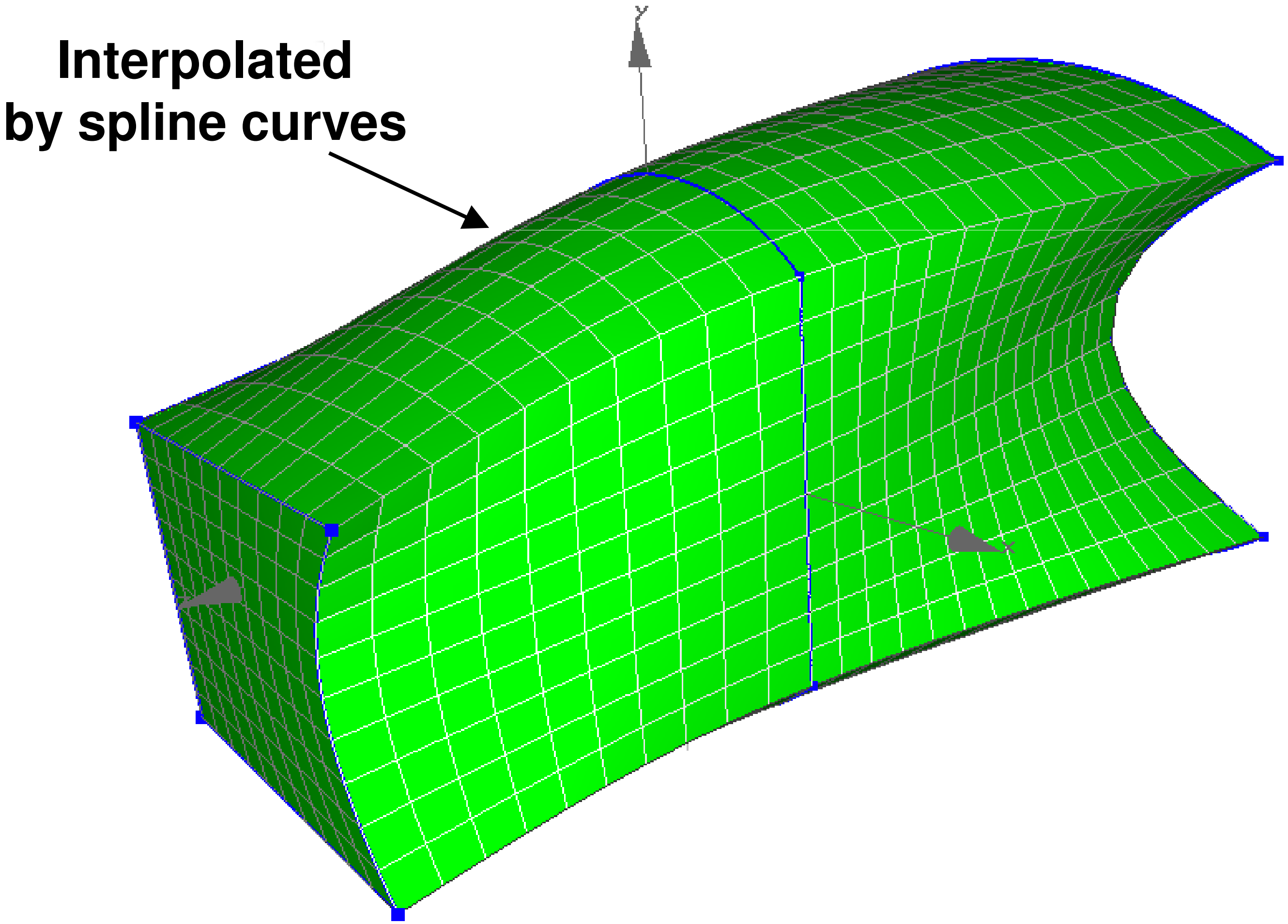
Structured mesh – Spline Sweeping

- **Geometry Requirements**
  - **Sequence of sections**
  - **Meshes must be topologically equal**



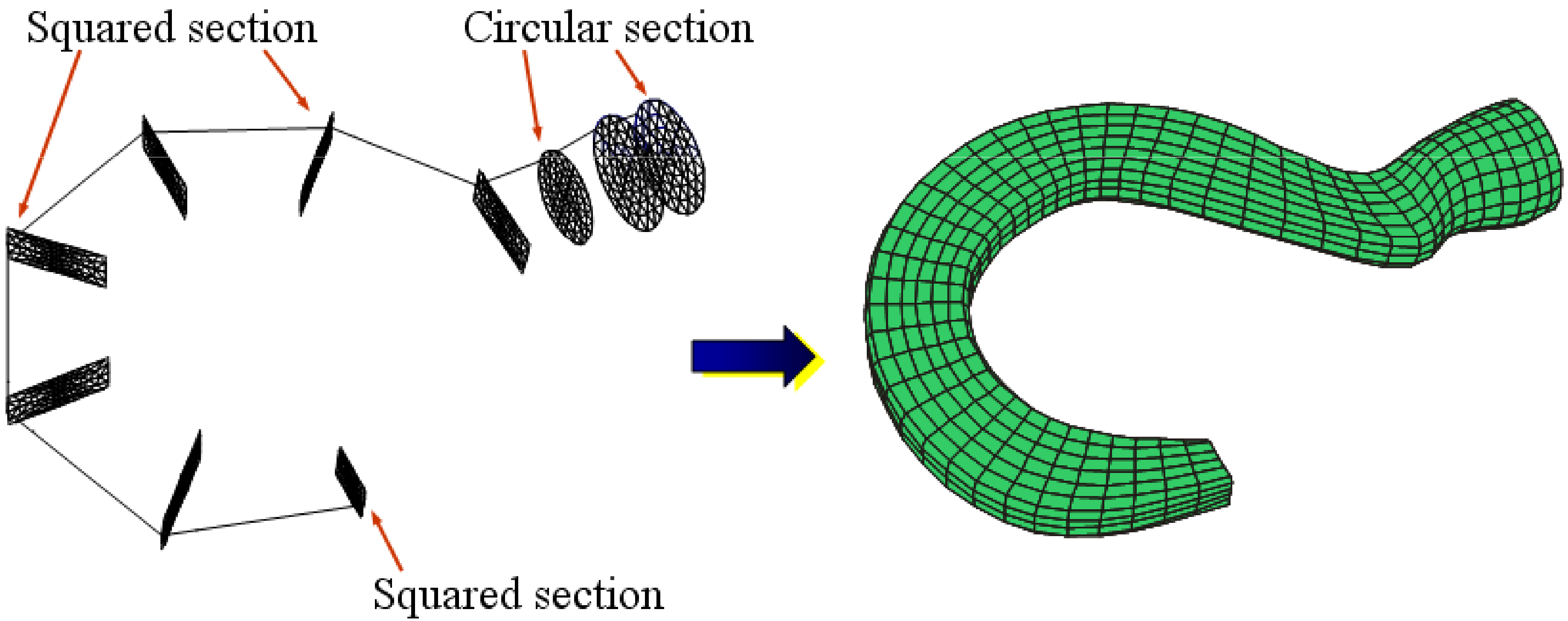
Structured mesh – Spline Sweeping

- **Geometry Requirements**
  - **Sequence of sections**
  - **Meshes must be topologically equal**



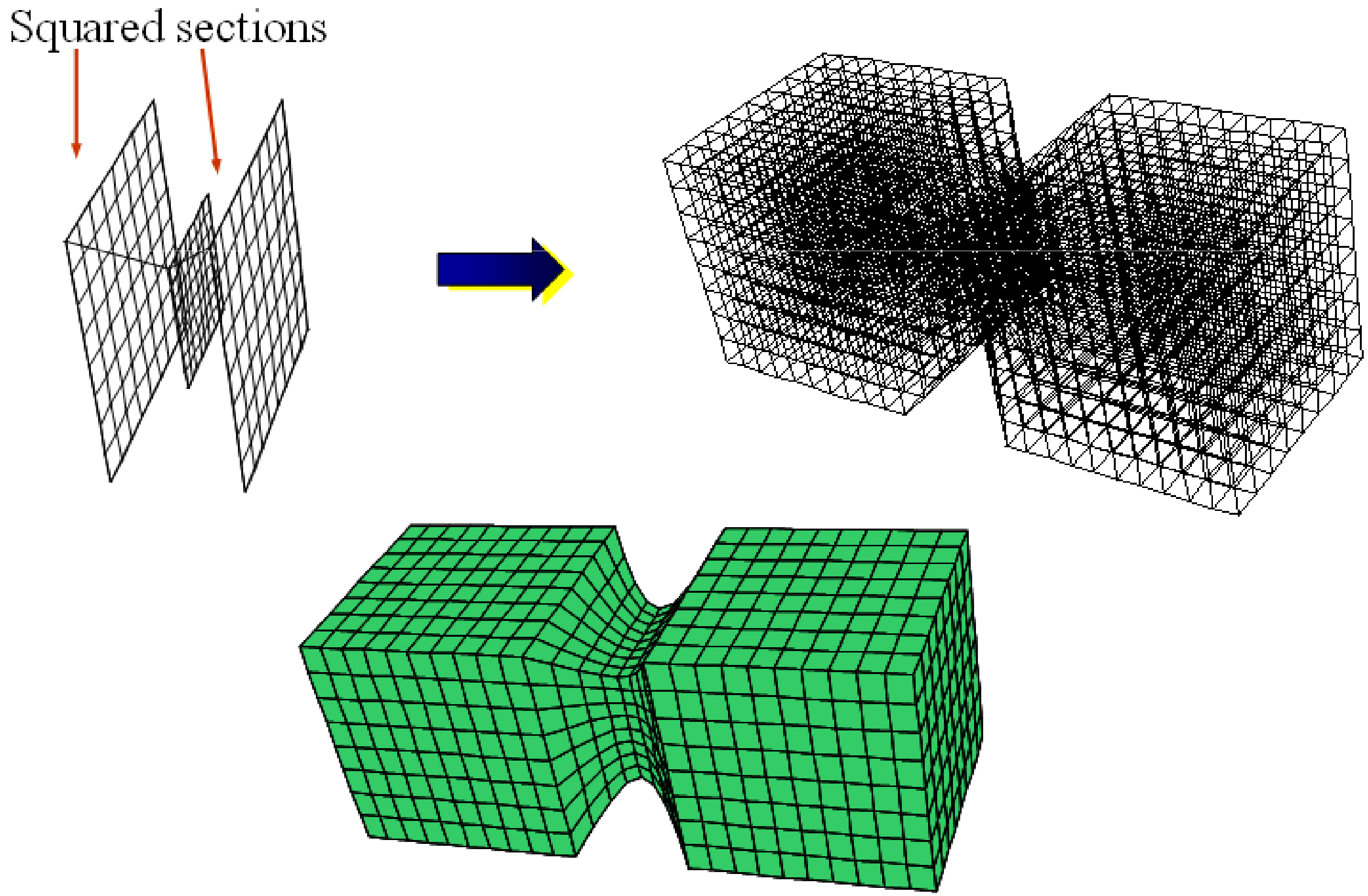
Structured mesh – Spline Sweeping

- **Geometry Requirements**
  - **Sequence of sections**
  - **Meshes must be topologically equal**



Structured mesh – Spline Sweeping

- **Geometry Requirements**
  - **Sequence of sections**
  - **Meshes must be topologically equal**



Unstructured mesh – Requirements

- **Specific algorithm requirements inherited from its ancestor**
  J-Mesh (Joaquim Cavalcante-Neto, Wawrzynek, Carvalho, Martha & Ingraffea; 2001):
  - **Generation of well-shaped elements**
  - **Ability to conform to an existing refinement at the boundary of region**
  - **Ability to transition well between regions with different element sizes**
  - **Capability for modeling discontinuities (internal restriction and cracks)**

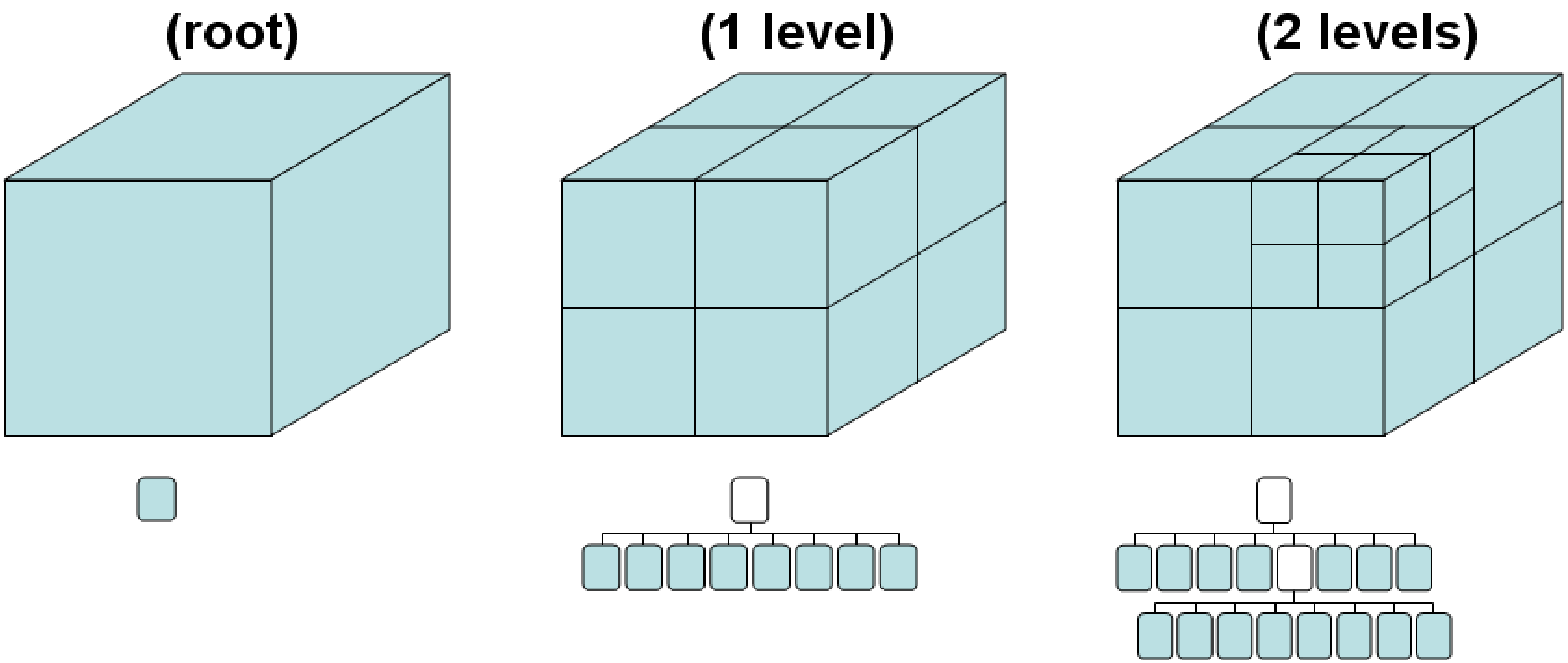
- **Additional requirements for surfaces**
  - **Locally refine the mesh in regions with curvatures**

Unstructured mesh generation outline

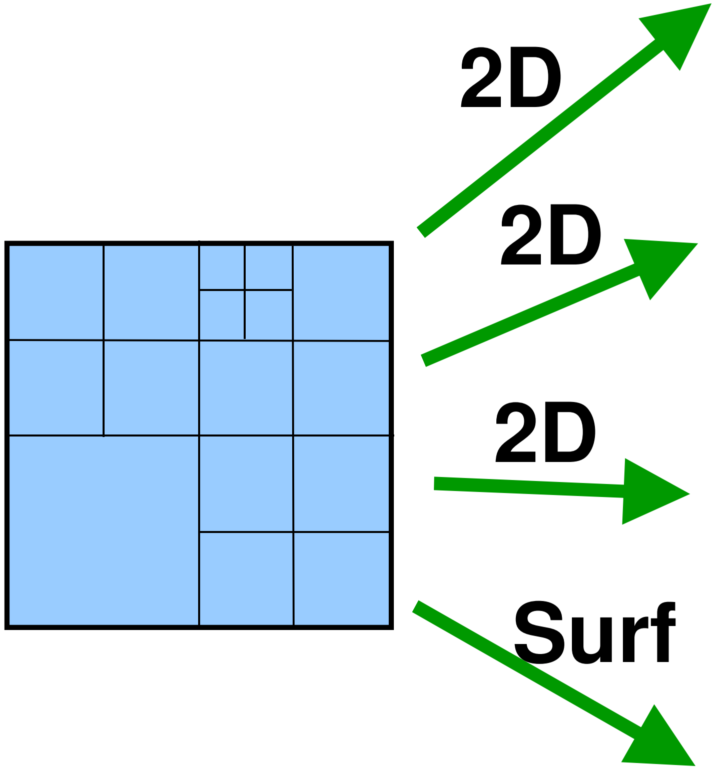
- **Background mesh generation – quadtree/octree**
  - **Initialization based on boundary mesh.**
  - **Refinement to force a maximum cell size.**
  - **Refinement to provide minimum size disparity for adjacent cells.**
- **Advancing-front procedure**
  - **Geometry-based element generation**
  - **Topology-based element generation**
  - **Element generation based on back-tracking with face deletion.**
- **Local mesh improvement**
  - **Laplacian smoothing,**
  - **Local back-tracking with element deletion, or**
  - **Taubin smoothing (surfaces)**

Unstructured mesh – auxiliary background structure

- **Quadtree and Octree**
  - **Fast search procedures to navigate through end leaves**
  - **Represent the desired size of elements with nearly the same size as the end leaves**



Unstructured mesh – 2D auxiliary background structure



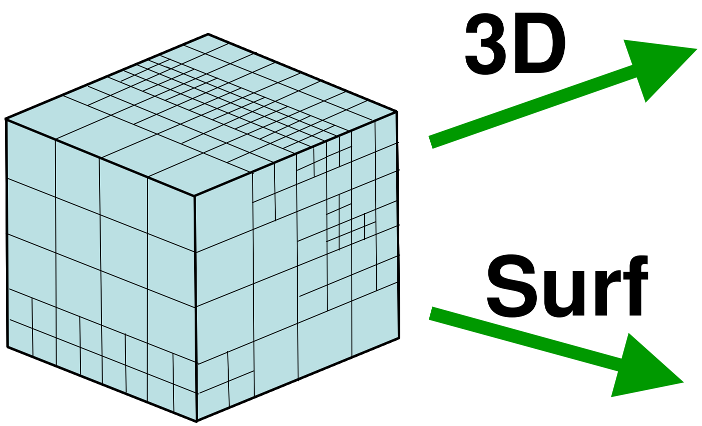
- Create internal points on domain
- Advancing front algorithm

- Create element using patterns in each cells
- Advancing front algorithm near boundary

- Use cell size as guideline to generate new elements
- Advancing front algorithm

- Use cells to store desired sizes for elements and surface metric information
- Advancing front algorithm in parametric space

Unstructured mesh – 3D auxiliary background structure

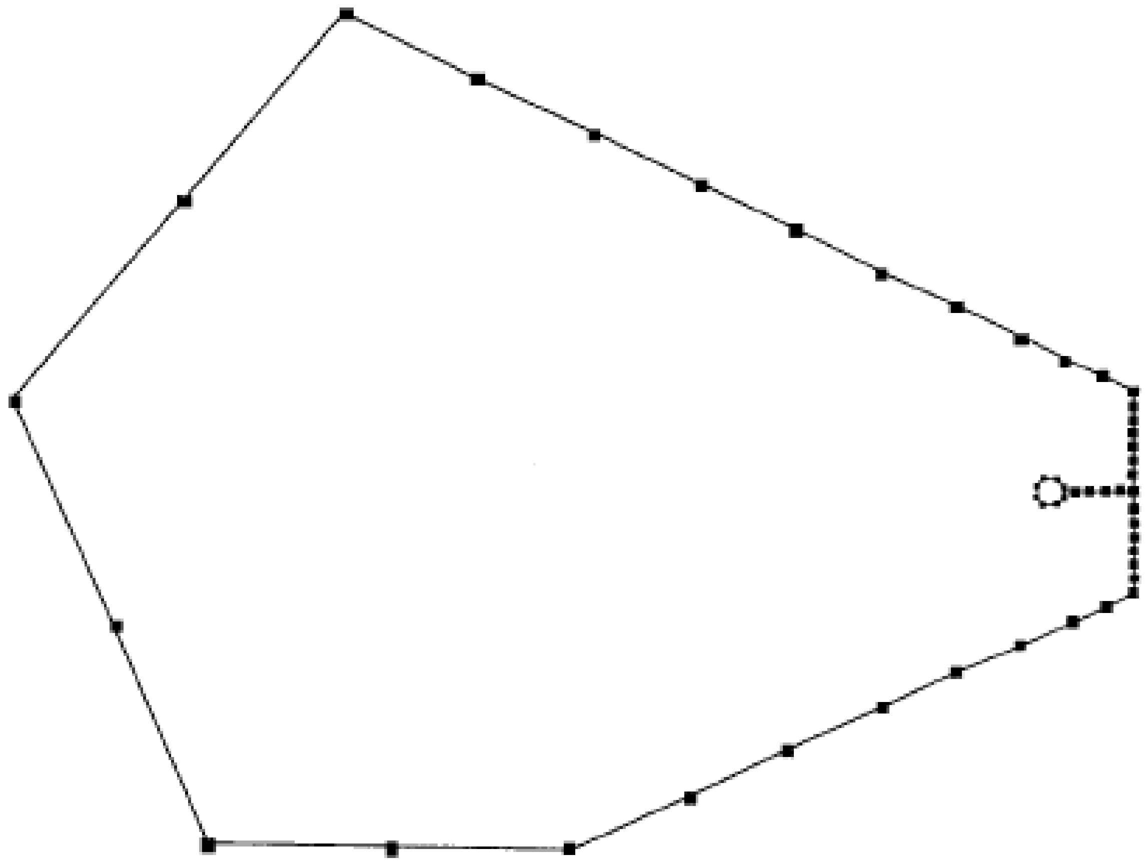


- Use cells to store desired sizes for elements
- Advancing front algorithm

- Use cells to store desired sizes for elements and surface metric information
- Advancing front algorithm direct in 3D space

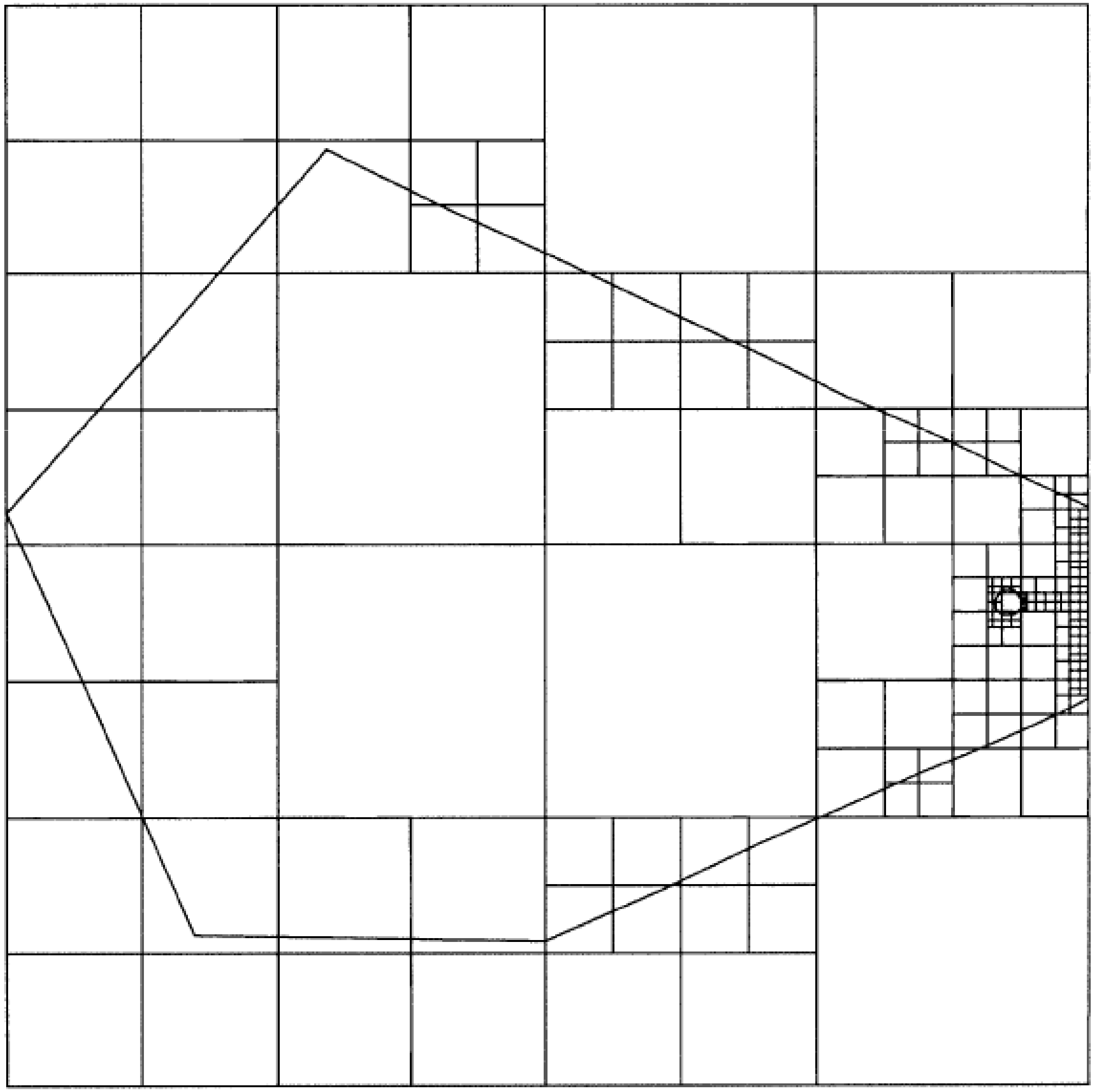
Unstructured mesh – background structure generation

- **Hypothetical 2D model and its boundary refinement**



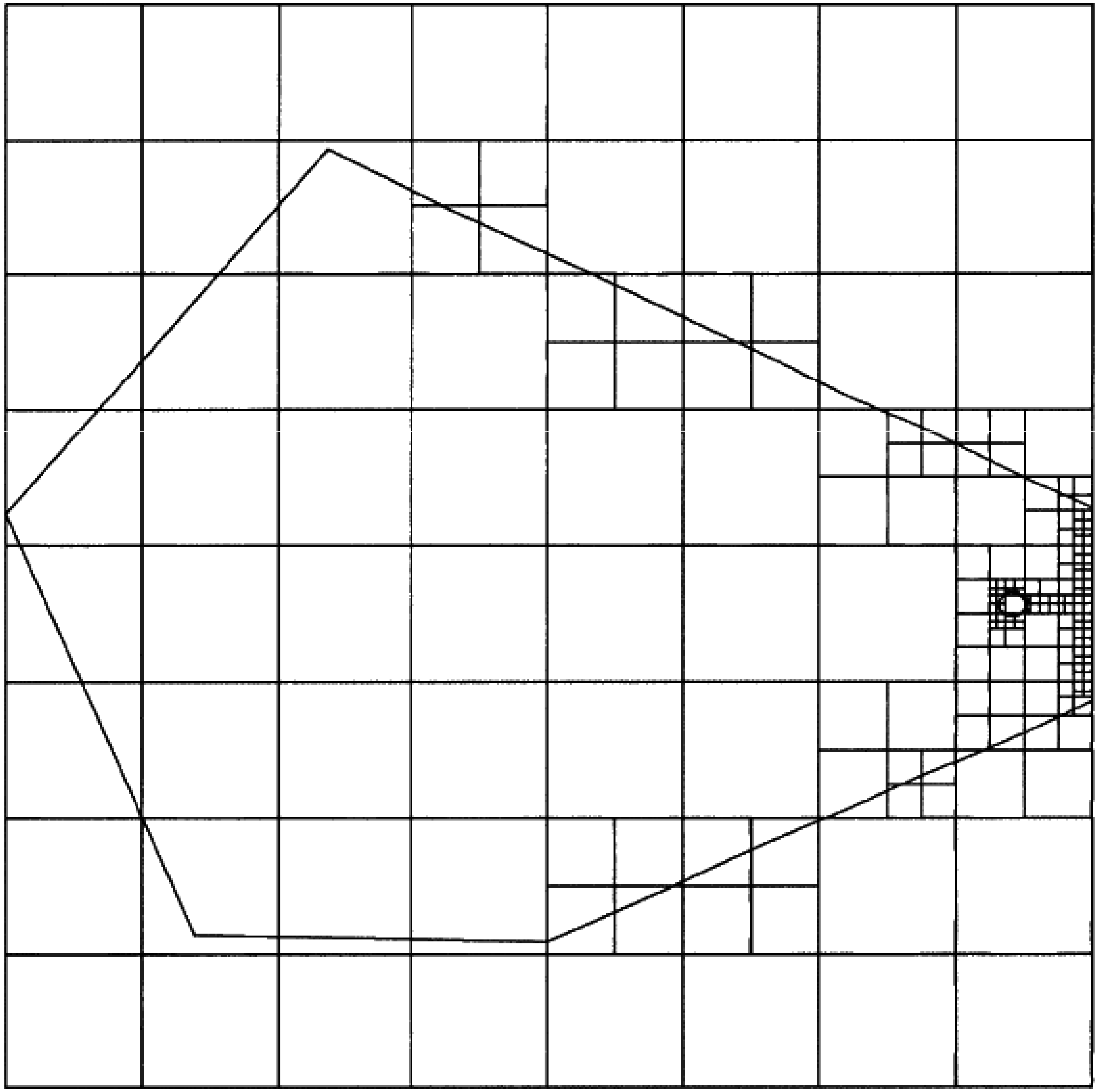
Unstructured mesh – background structure generation

- **Initialization based on boundary mesh**



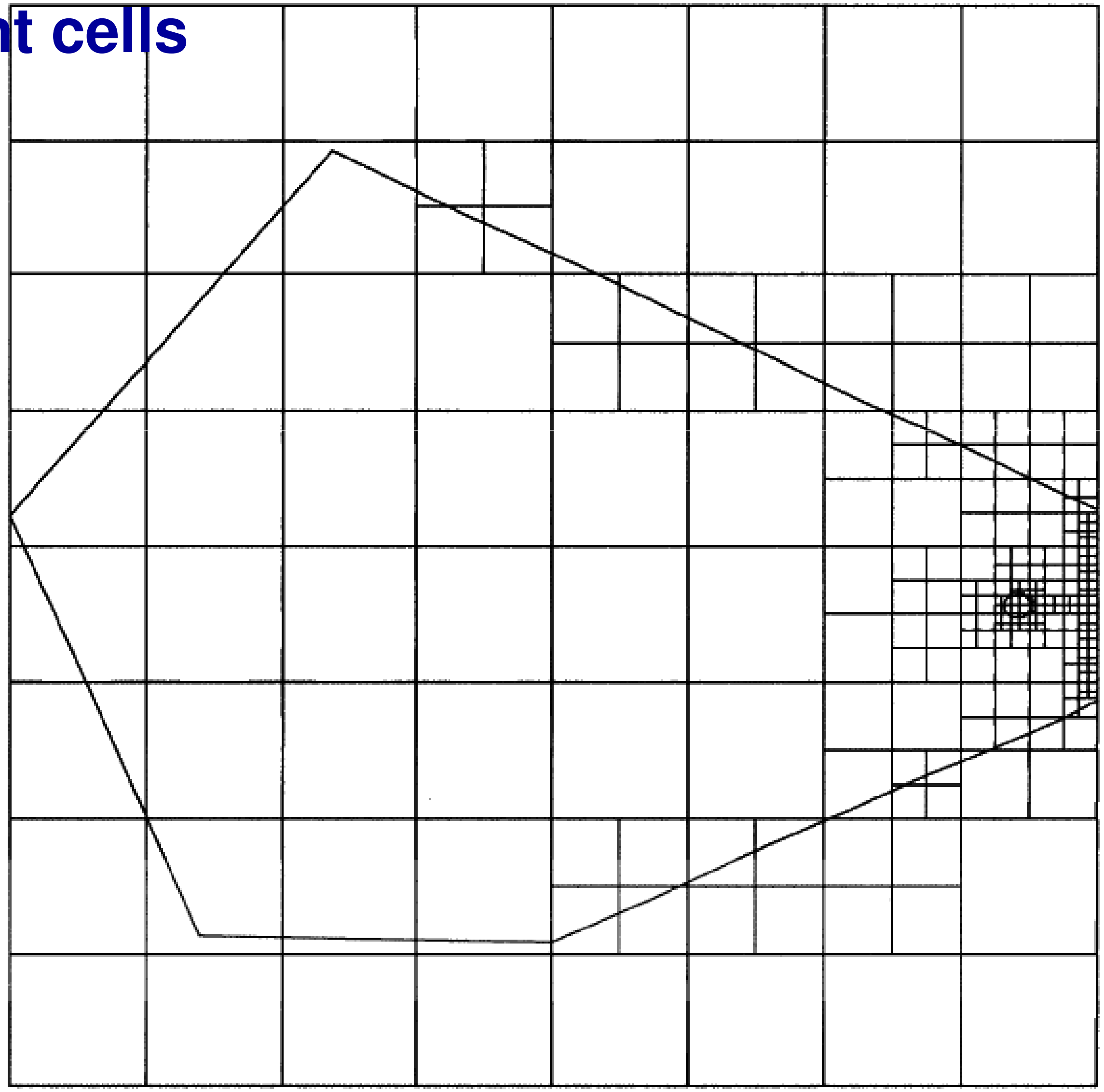
Unstructured mesh – background structure generation

- **Refinement to force a maximum cell size**



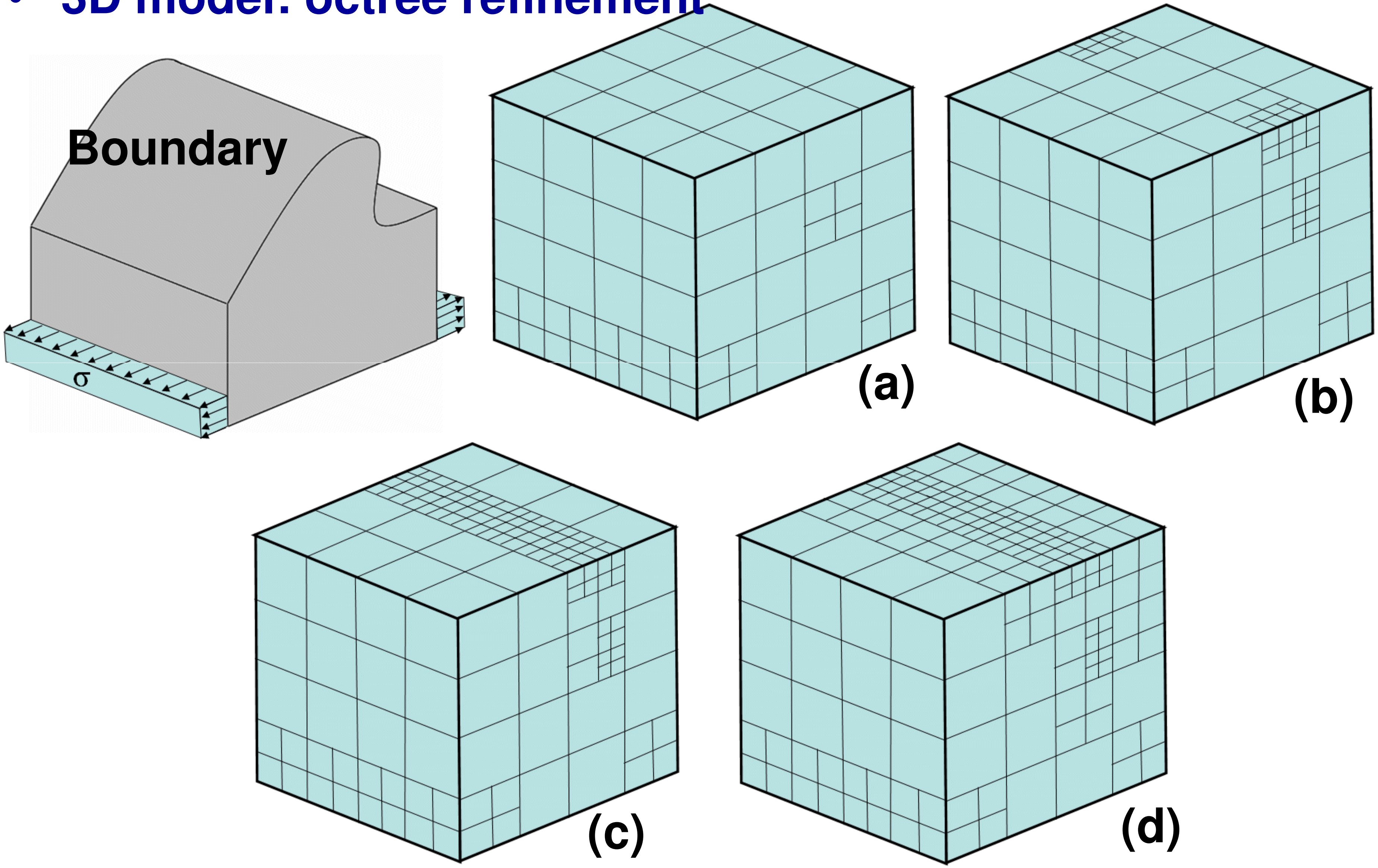
Unstructured mesh – background structure generation

- **Refinement to provide minimum size disparity for adjacent cells**



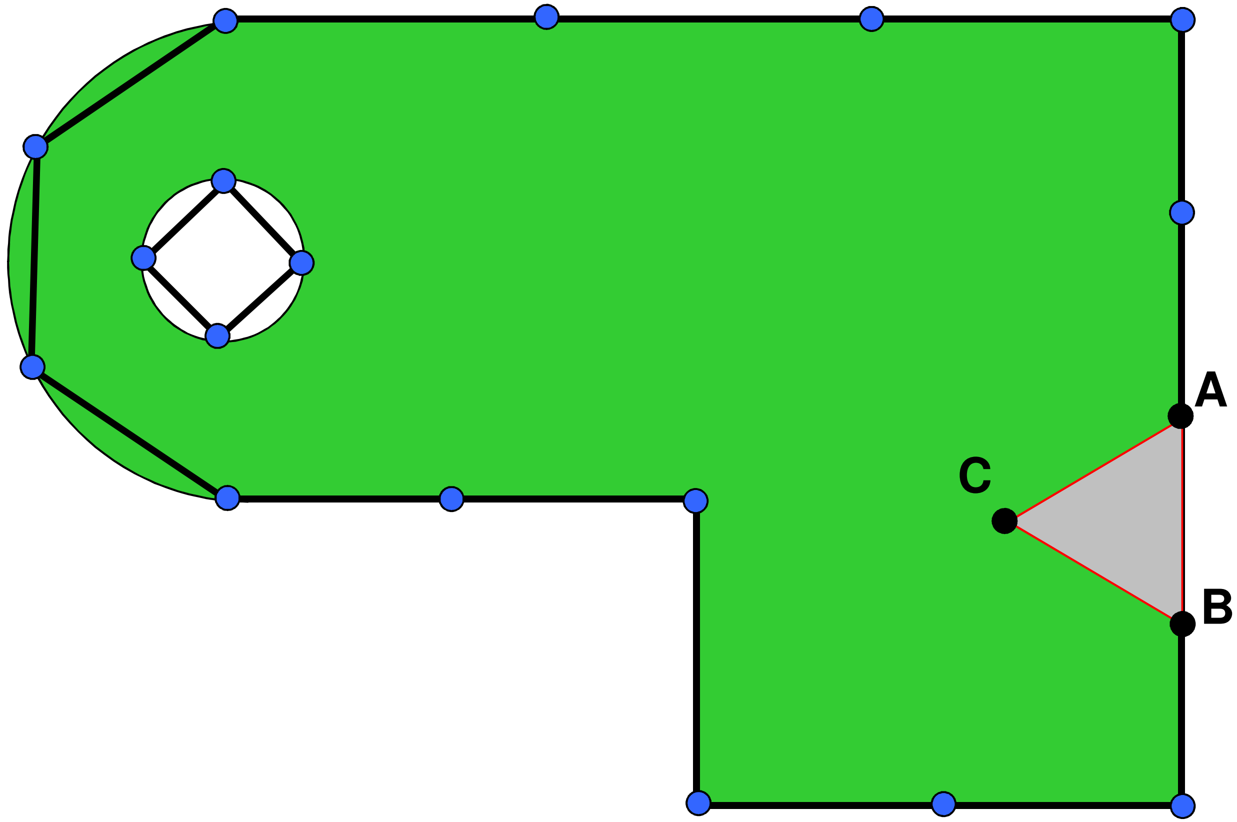
Unstructured mesh – background structure generation

- **3D model: octree refinement**



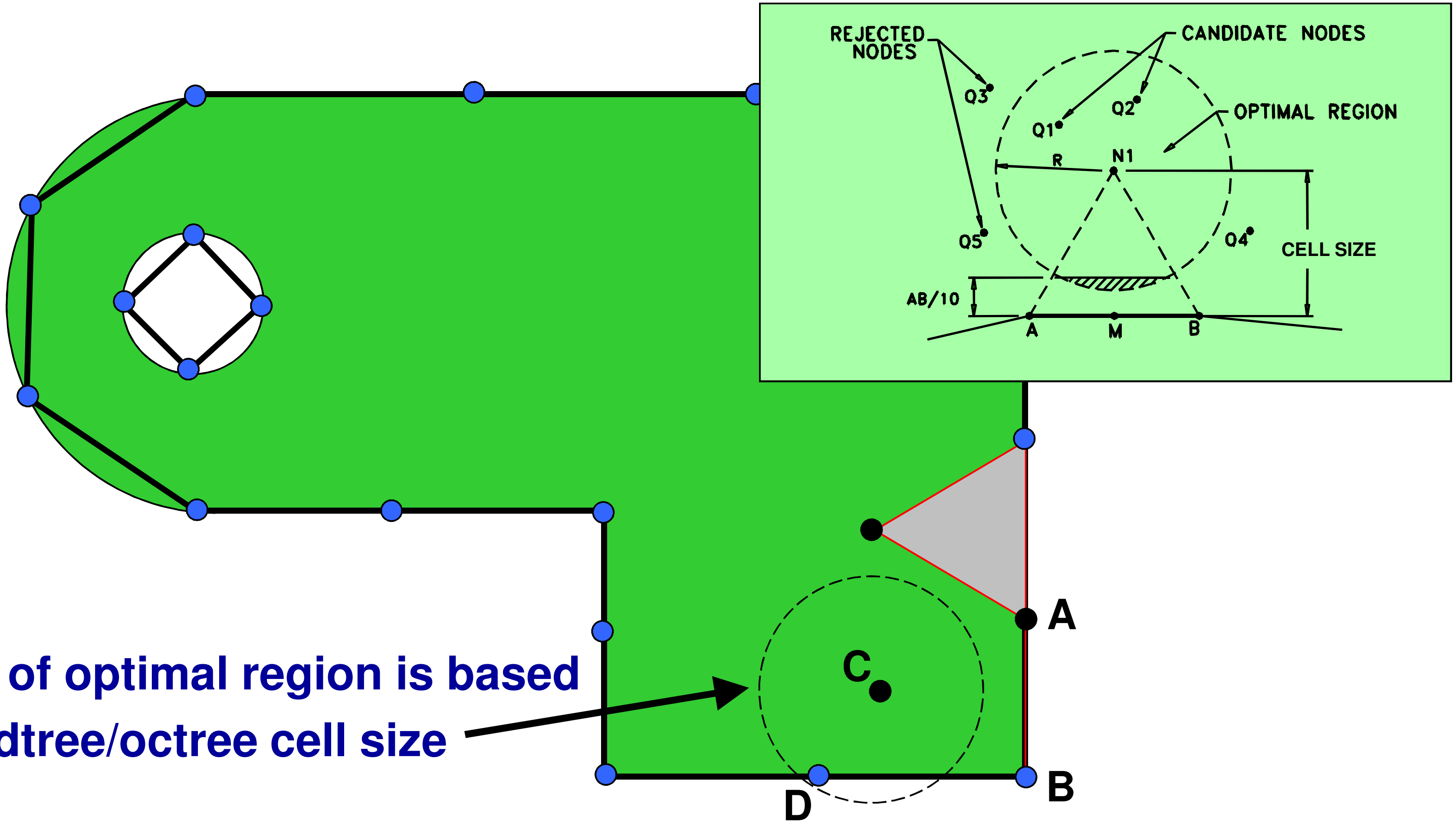
Unstructured mesh – advancing-front technique

- **Advancing front algorithm**
  - **Begin with boundary mesh – define as initial *front***
  - **For each edge (face) on front, locate initial node C based on front AB**



Unstructured mesh – advancing-front technique

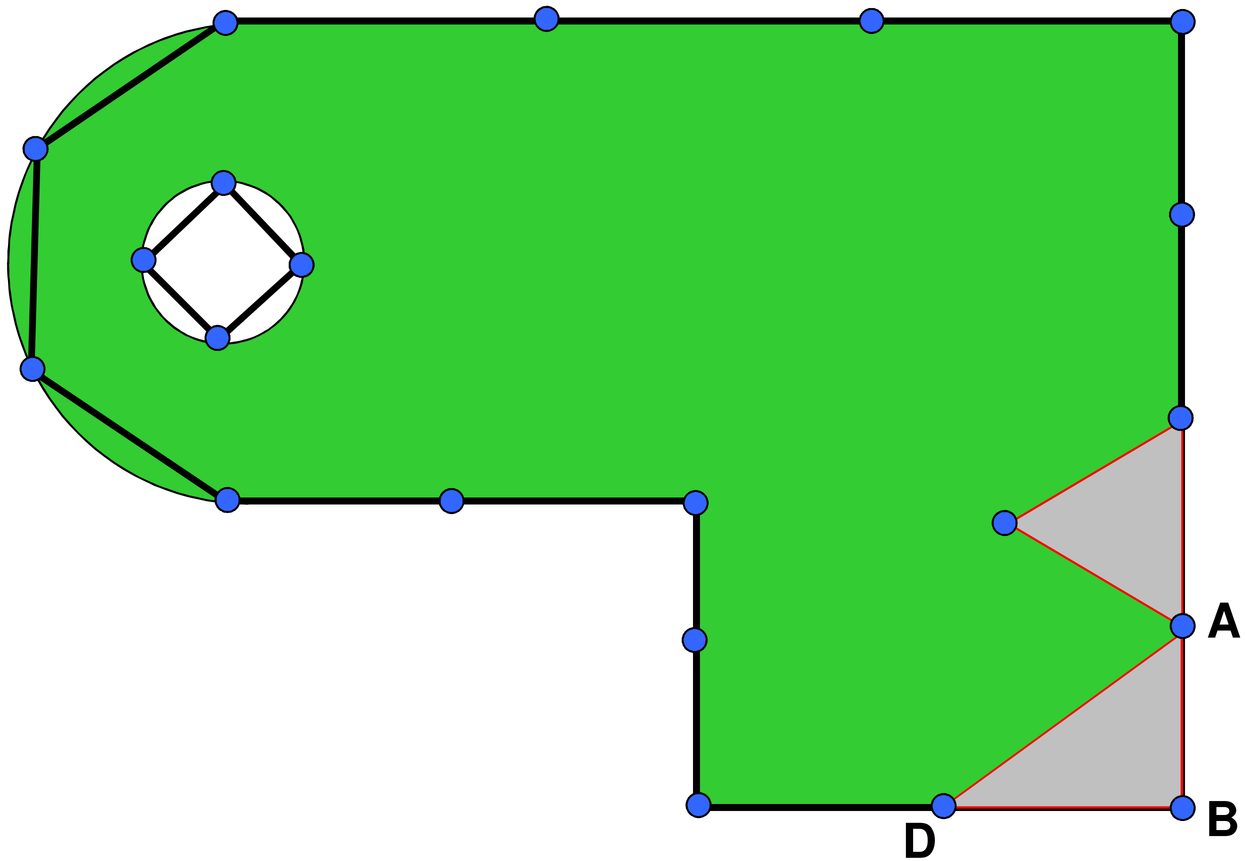
- **Advancing front algorithm**
  - **Determine if any other node on current from are within search radius *r* of ideal location C (Choose D instead of C)**



Radius of optimal region is based on quadtree/octree cell size

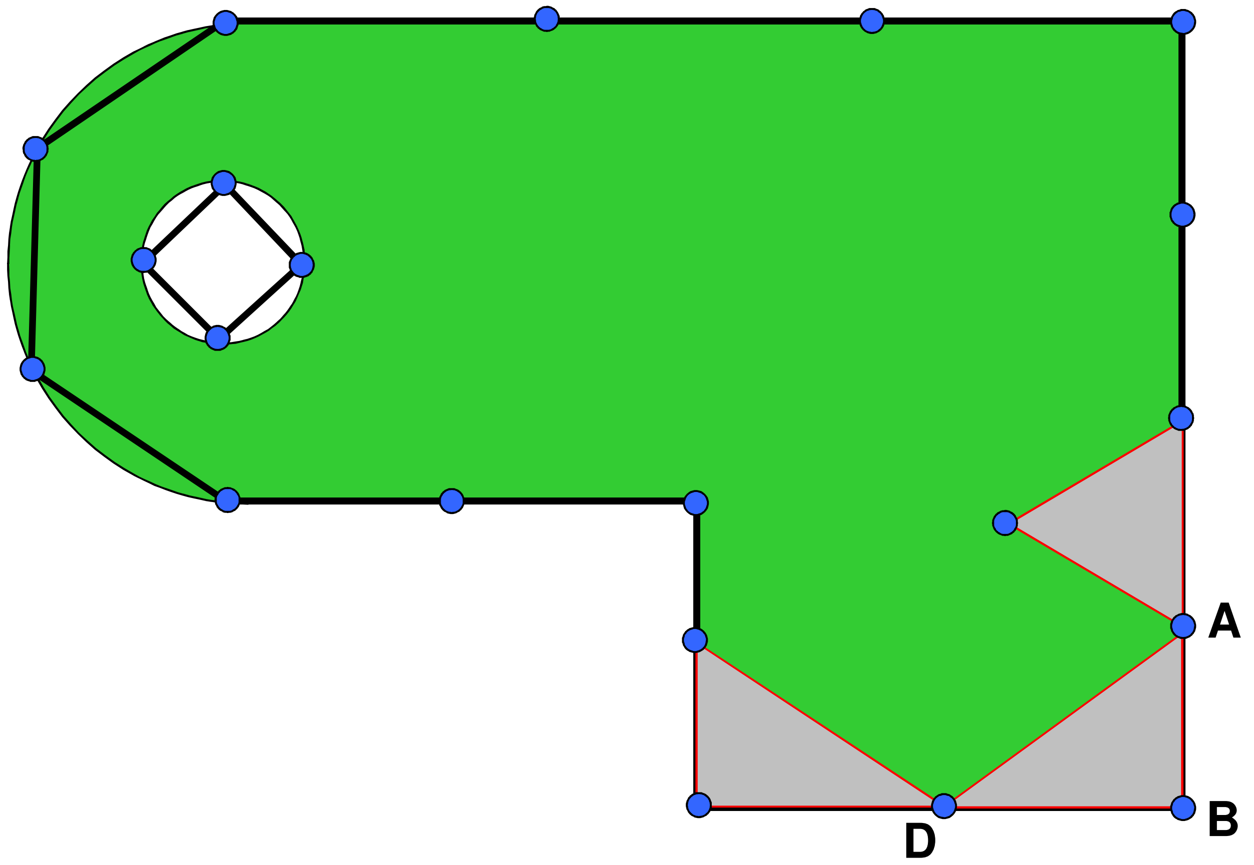
Unstructured mesh – advancing-front technique

- **Advancing front algorithm**
  - **New *front edges* (*faces*) added and deleted from *front* as triangles (tetrahedral) are formed**
  - **Continue until *front edges* (*faces*) remain on *front***



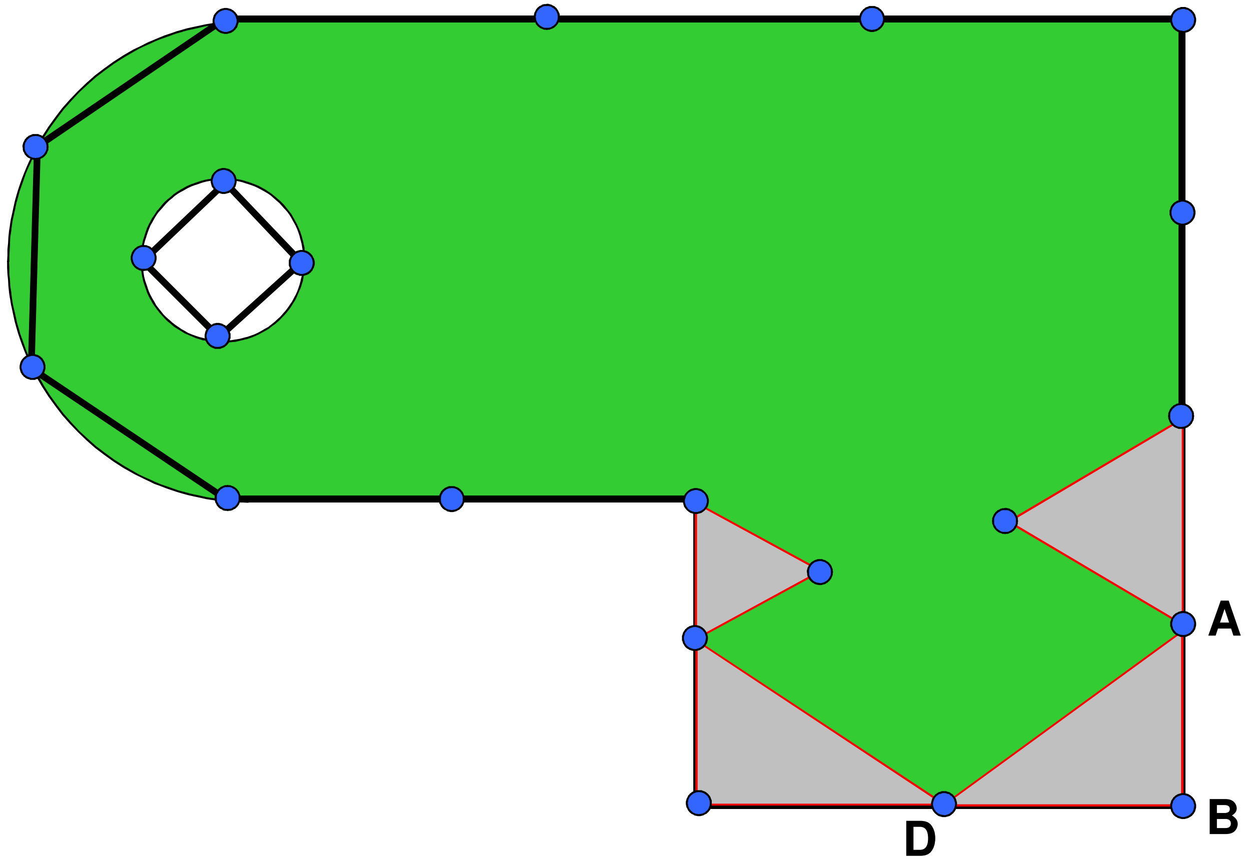
Unstructured mesh – advancing-front technique

- **Advancing front algorithm**
  - **New *front edges* added and deleted from *front* as triangles are formed**
  - **Continue until *front edges* remain on *front***



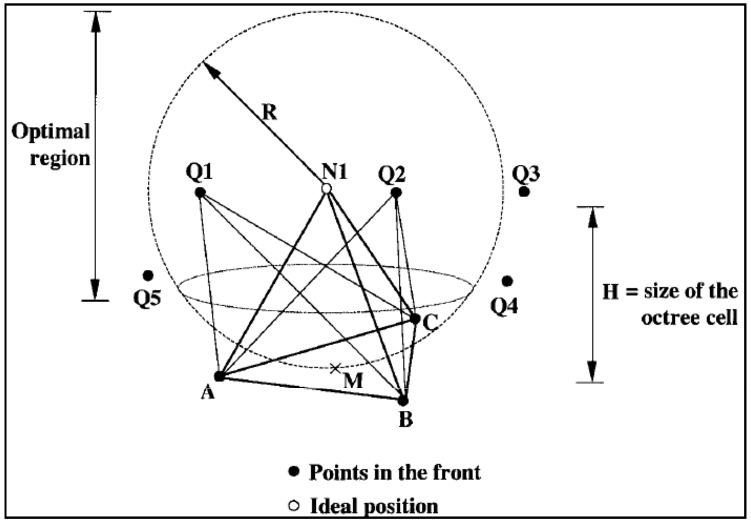
Unstructured mesh – advancing-front technique

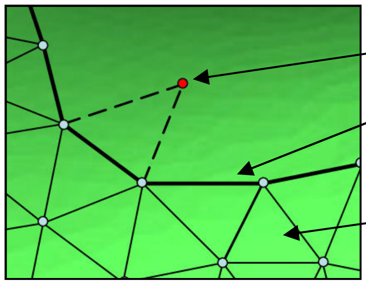
- **Advancing front algorithm**
  - **New *front edges* added and deleted from *front* as triangles are formed**
  - **Continue until *front edges* remain on *front***

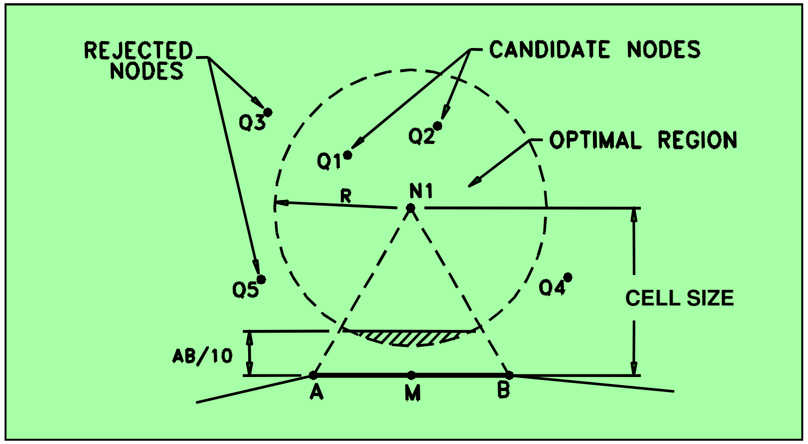


Unstructured mesh – advancing-front technique

- **Geometry-based element generation**
  - **Boundary contraction list**
    - **List of active edges**
    - **List of rejected edges**
  - **Generation of optimal elements**
    - **Size of element**
    - **Optimal location N1**
    - **Ratio = 0.85 * size**
    - **Upper bound and lower bond**
    - **Range Tree Search**

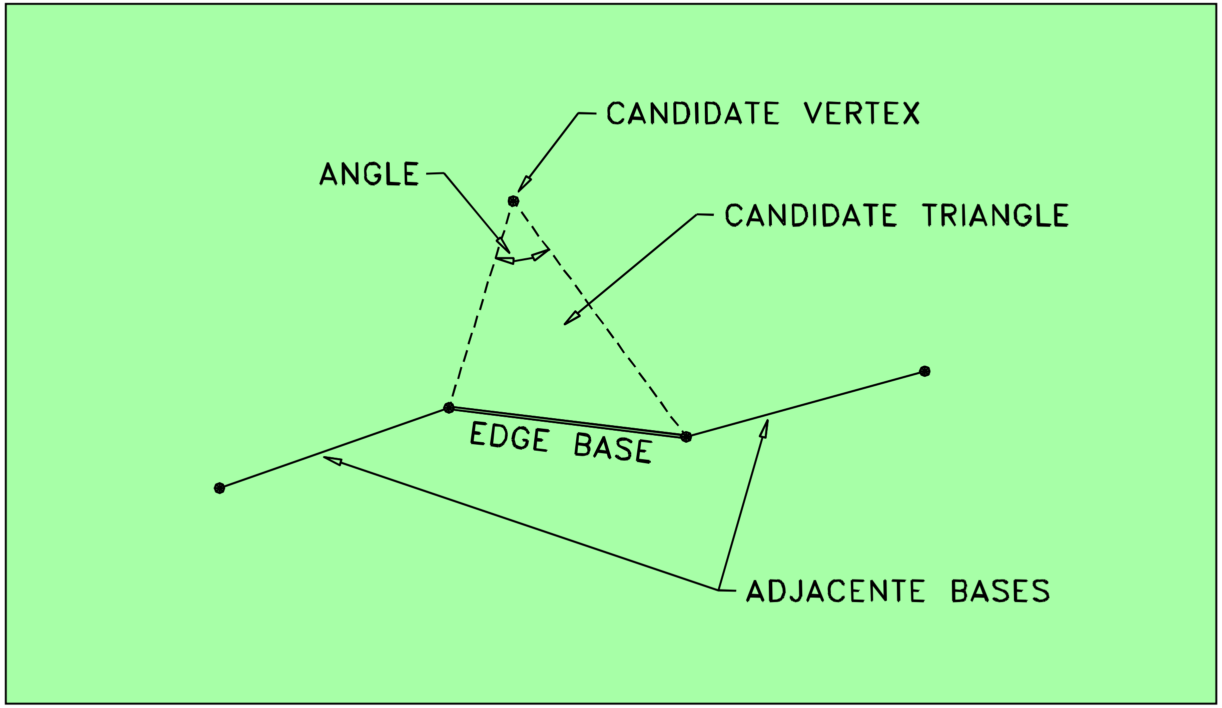






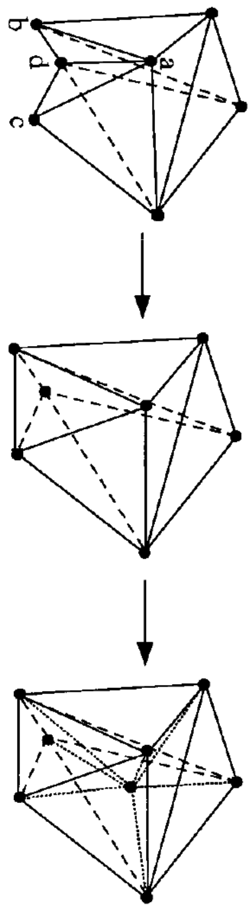
Unstructured mesh – advancing-front technique

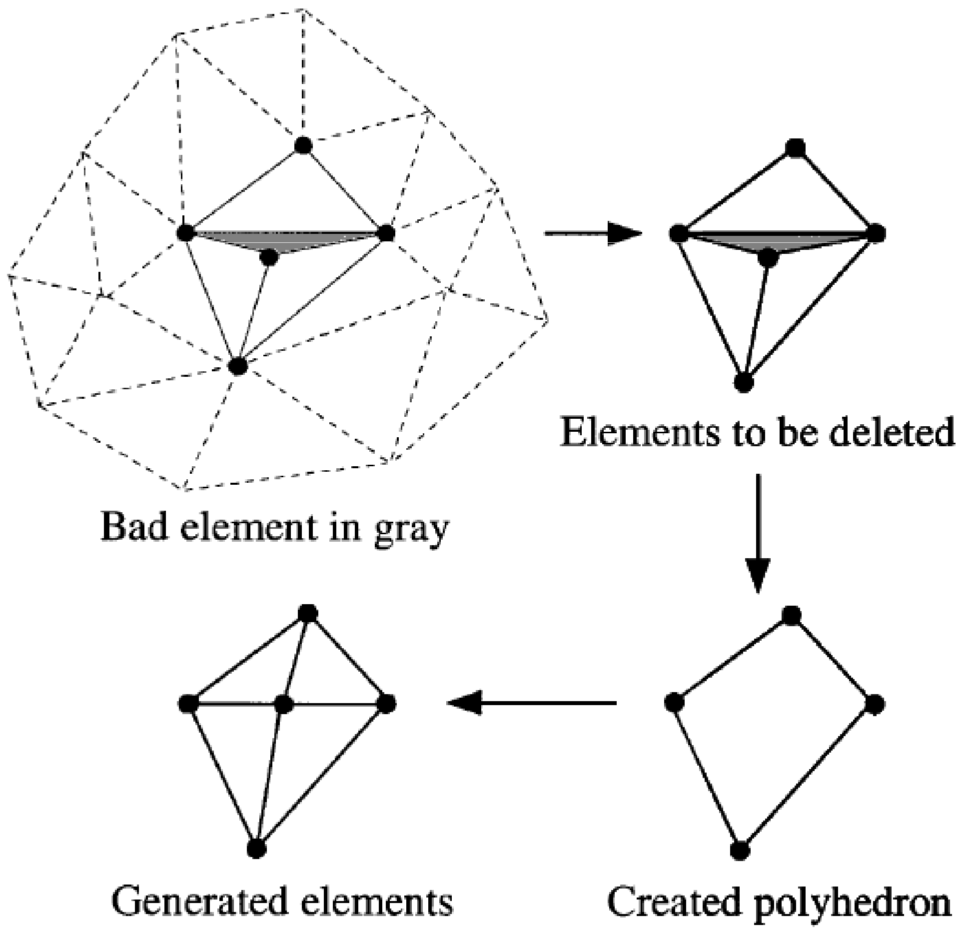
- **Topology-based element generation**
  - **List of rejected edges becomes active edges**
  - **Generation of elements by any node close to the base edge (best angle)**
  - **Generate a valid mesh, although not optimal**



Unstructured mesh – advancing-front technique

- **Back-Tracking**
  - **Locally modify the advancing front, deleting already generated adjacent tetrahedra until a 'near' convex non-meshed polyhedron is formed**

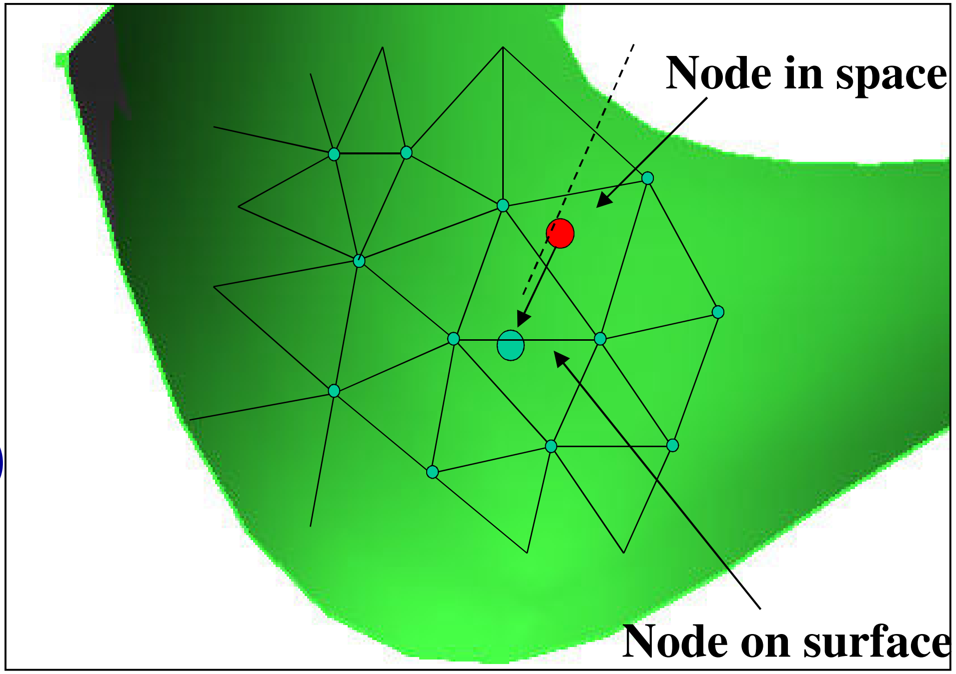




Unstructured mesh – local mesh improvement

- **Laplacian smoothing**
  - **Uses Laplacian equation and the closest point function for surface**

$$X_0^{n+1} = X_0^n + \phi \frac{\sum_{i=1}^{m} w_{i0}(X_i^n - X_0^n)}{\sum_{i=1}^{m} w_{i0}}$$



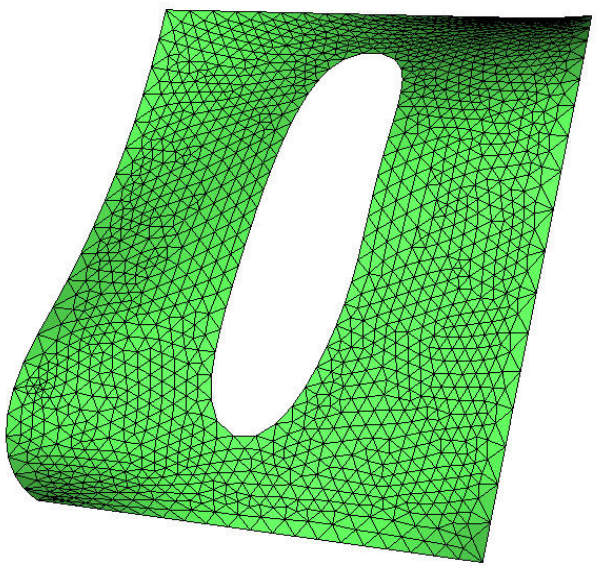
  - **ϕ = 1.0 and w_{i0}=1.0**

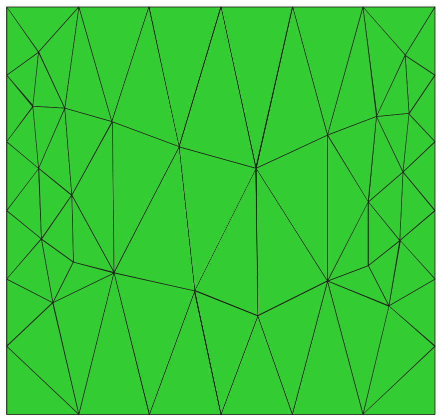
- **Taubin smoothing (surfaces)**
  - **Uses twice Laplacian equation**
    - **ϕ = 1.0 and w_{i0} = 0.63**
    - **ϕ = 1.0 and w_{i0} = - 0.67**
  - **Filters high frequencies**
  - **Preserves the low frequencies**
  - **Good results with geological and microstructure surfaces**

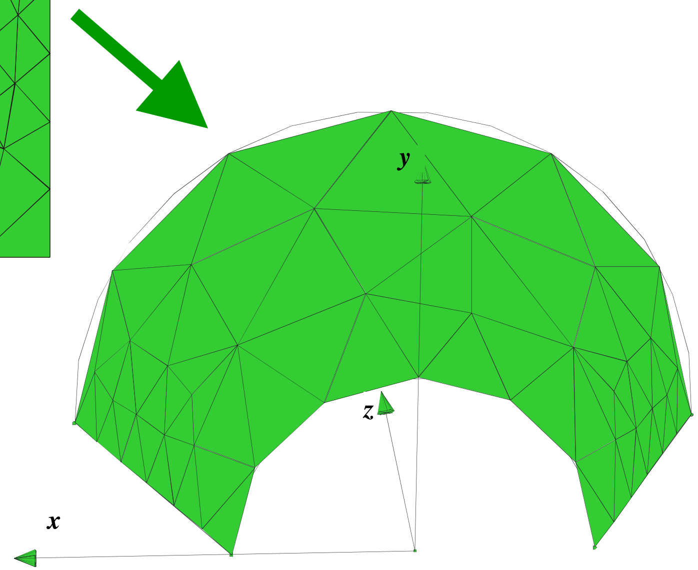
Unstructured mesh – Surface Meshing

- **Direct 3D Meshing**
  - **Elements formed in 3D using actual x-y-z representation of surface**

- **Parametric Space Meshing**
  - **Elements formed in 2D using parametric representation of surface**
  - **Nodes locations later mapped to 3D space**

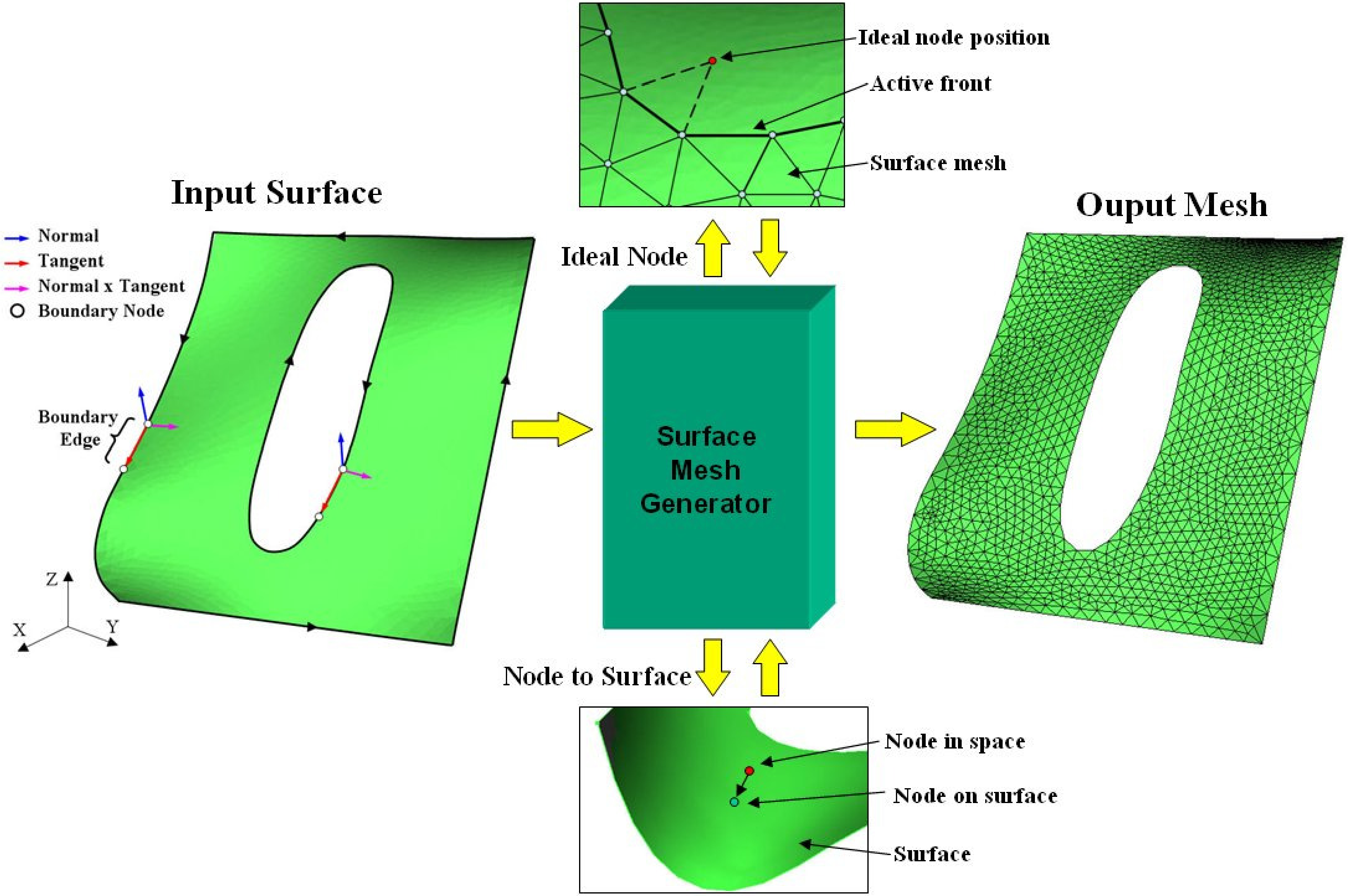






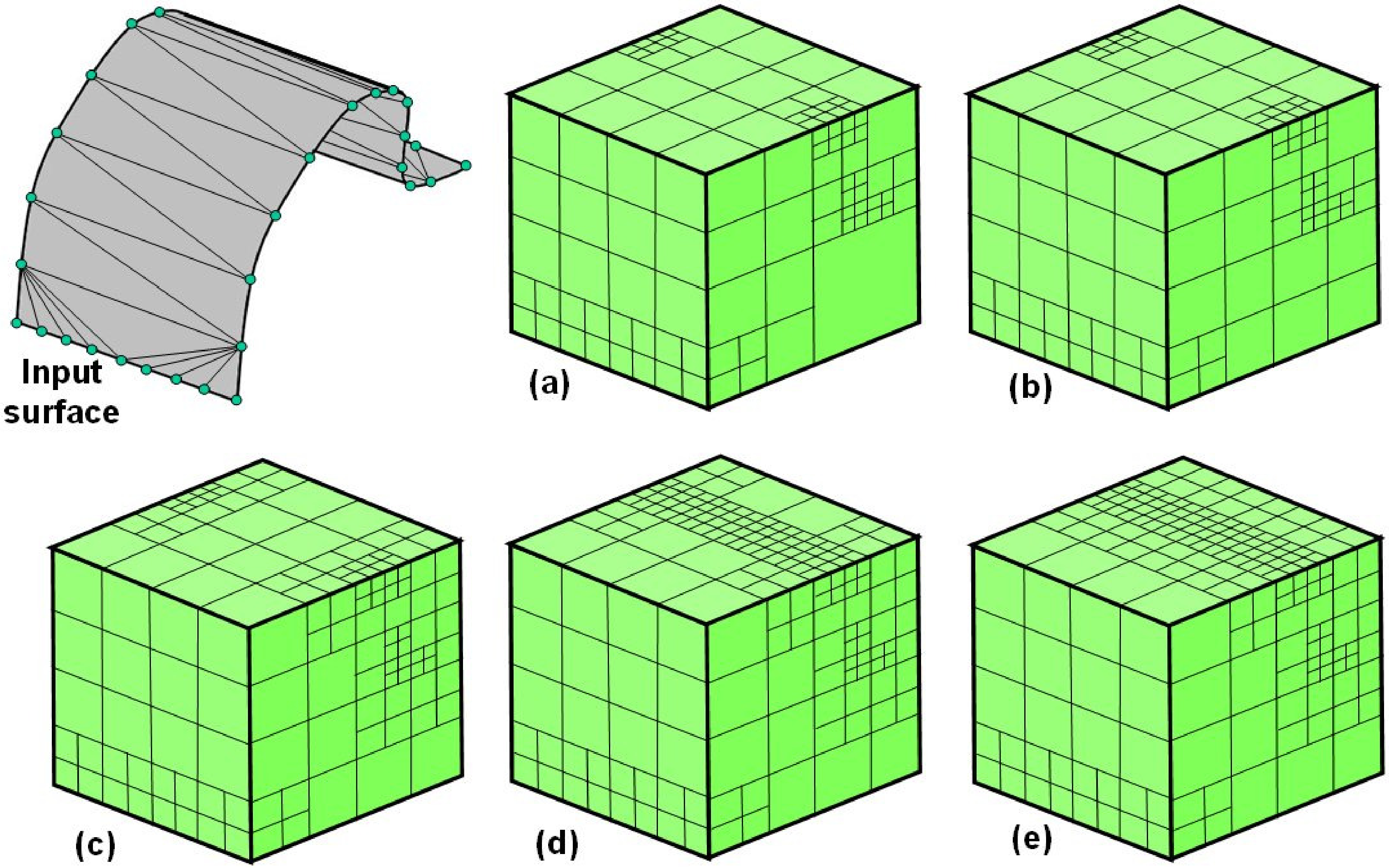
Unstructured mesh – Surface Meshing

- **Direct 3D Meshing**

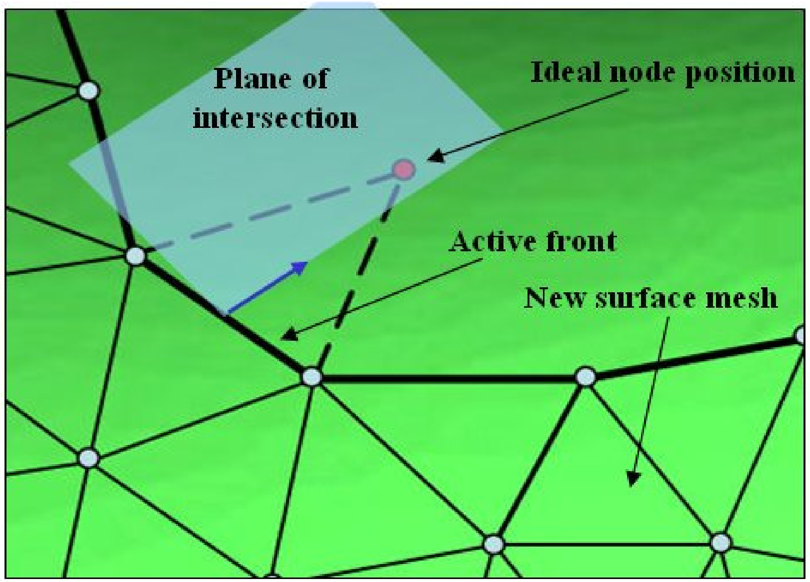


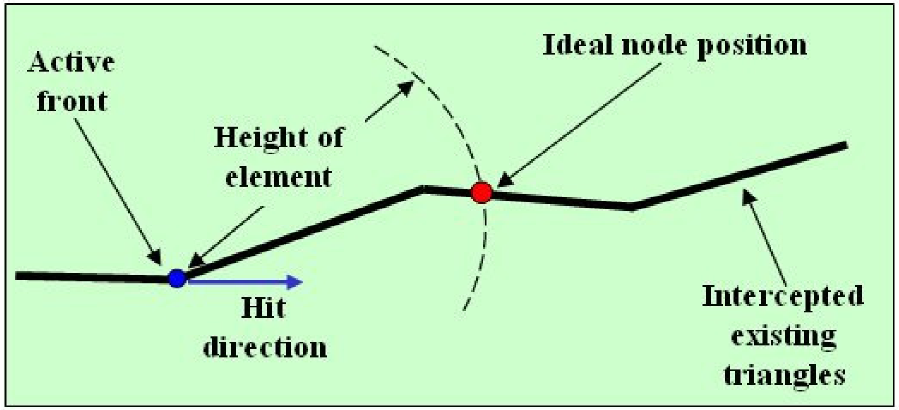
Unstructured mesh – Surface Meshing

- **Direct 3D Meshing – refinement of octree**



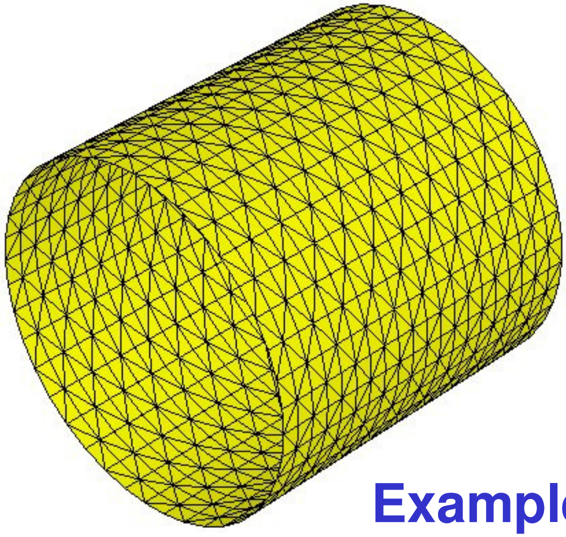
- **Direct 3D Meshing – node location**



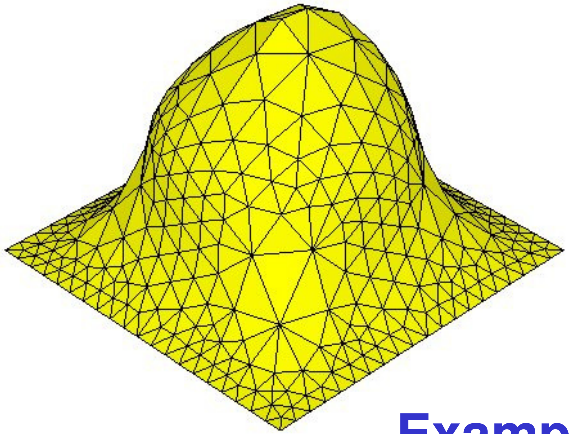


Unstructured mesh – Surface Meshing

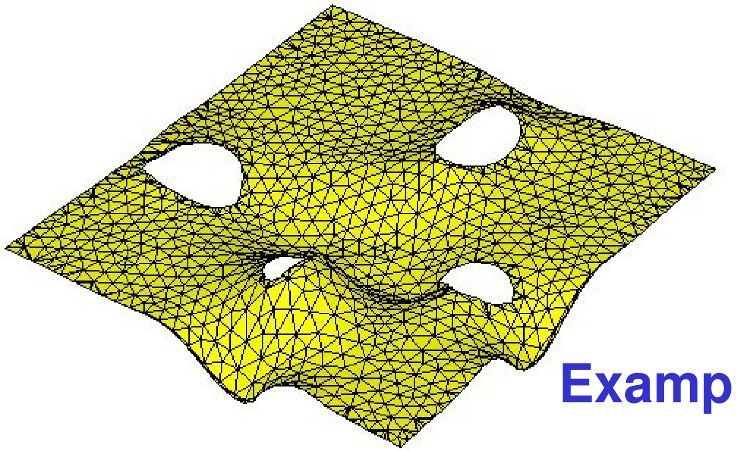
- **Direct 3D Meshing – Examples**



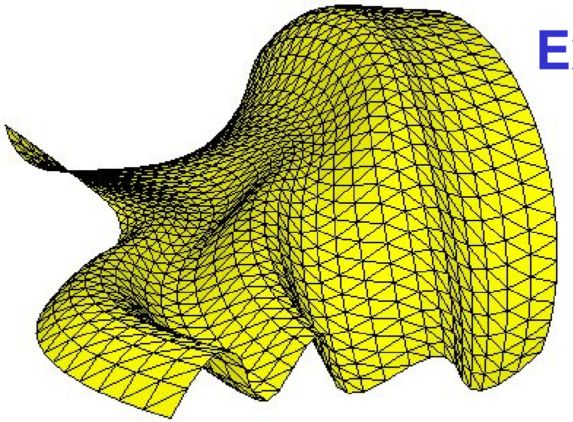
Example 1



Example 2



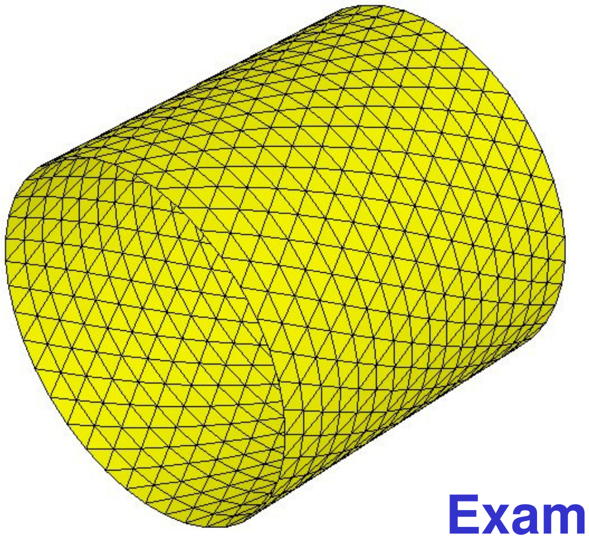
Example 3



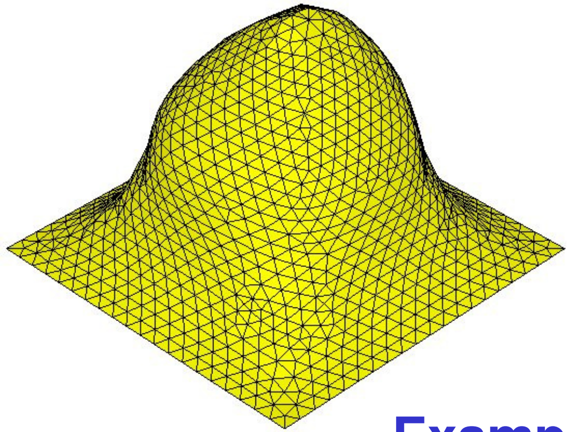
Example 4

Unstructured mesh – Surface Meshing

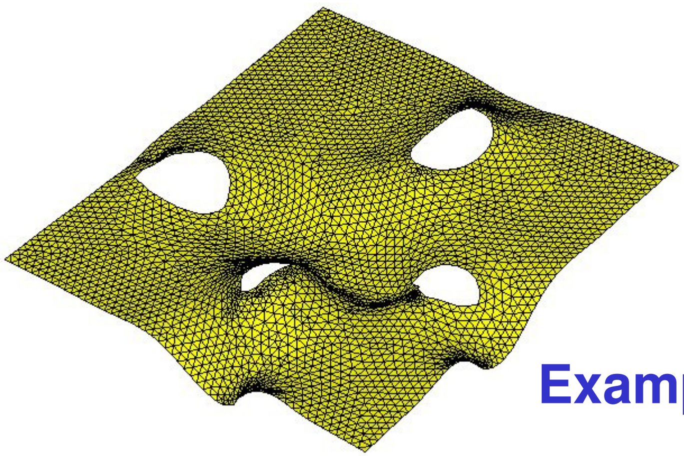
- **Direct 3D Meshing – Examples**



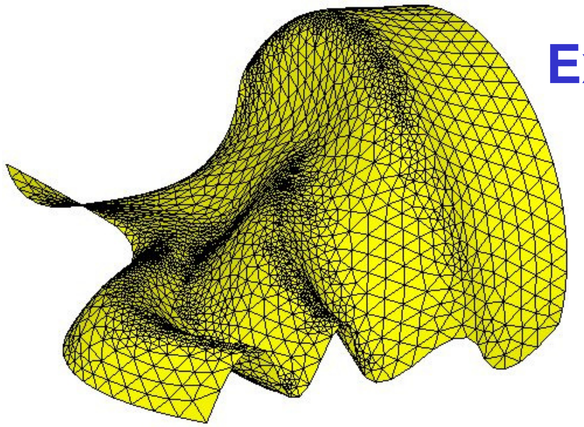
Example 1



Example 2

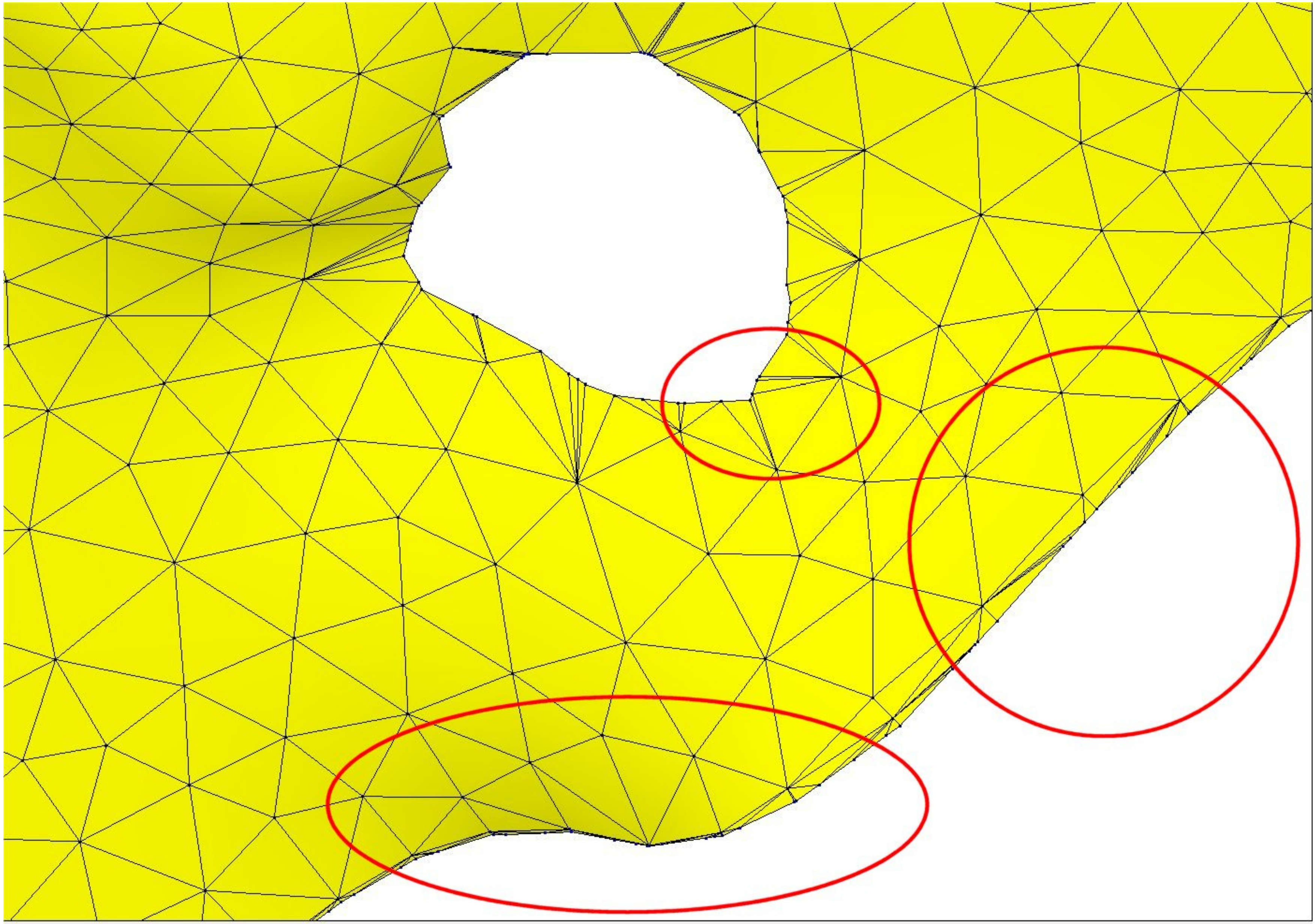


Example 3

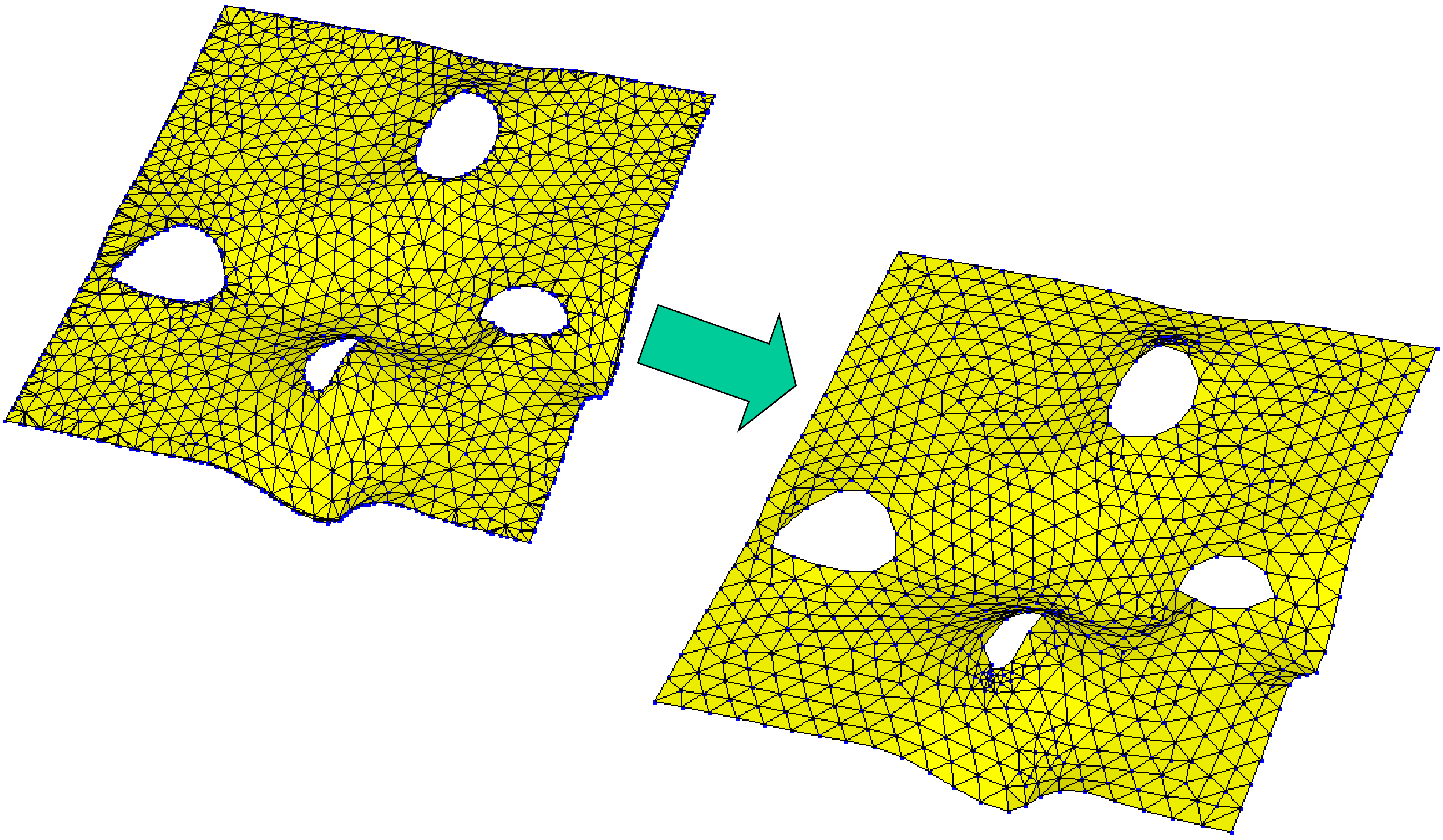


Example 4

Imported triangulation with poorly-shaped elements

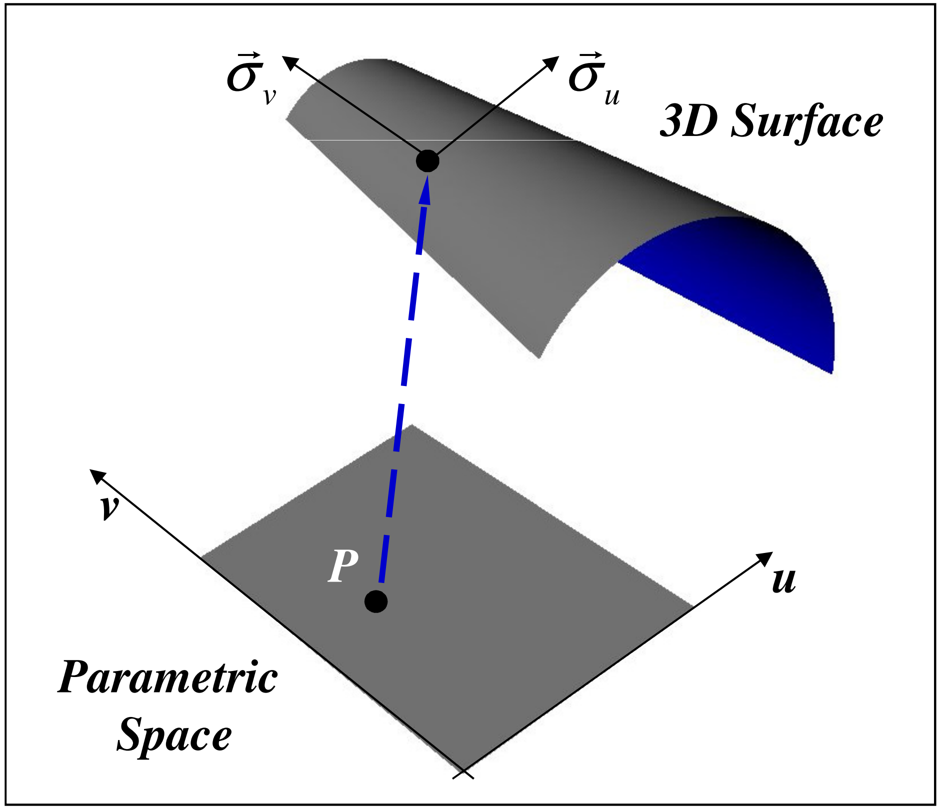


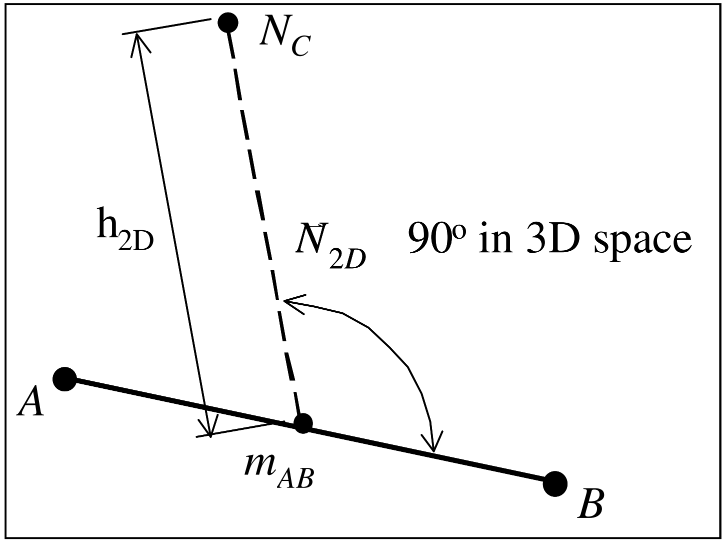
Example of surface re-triangulation



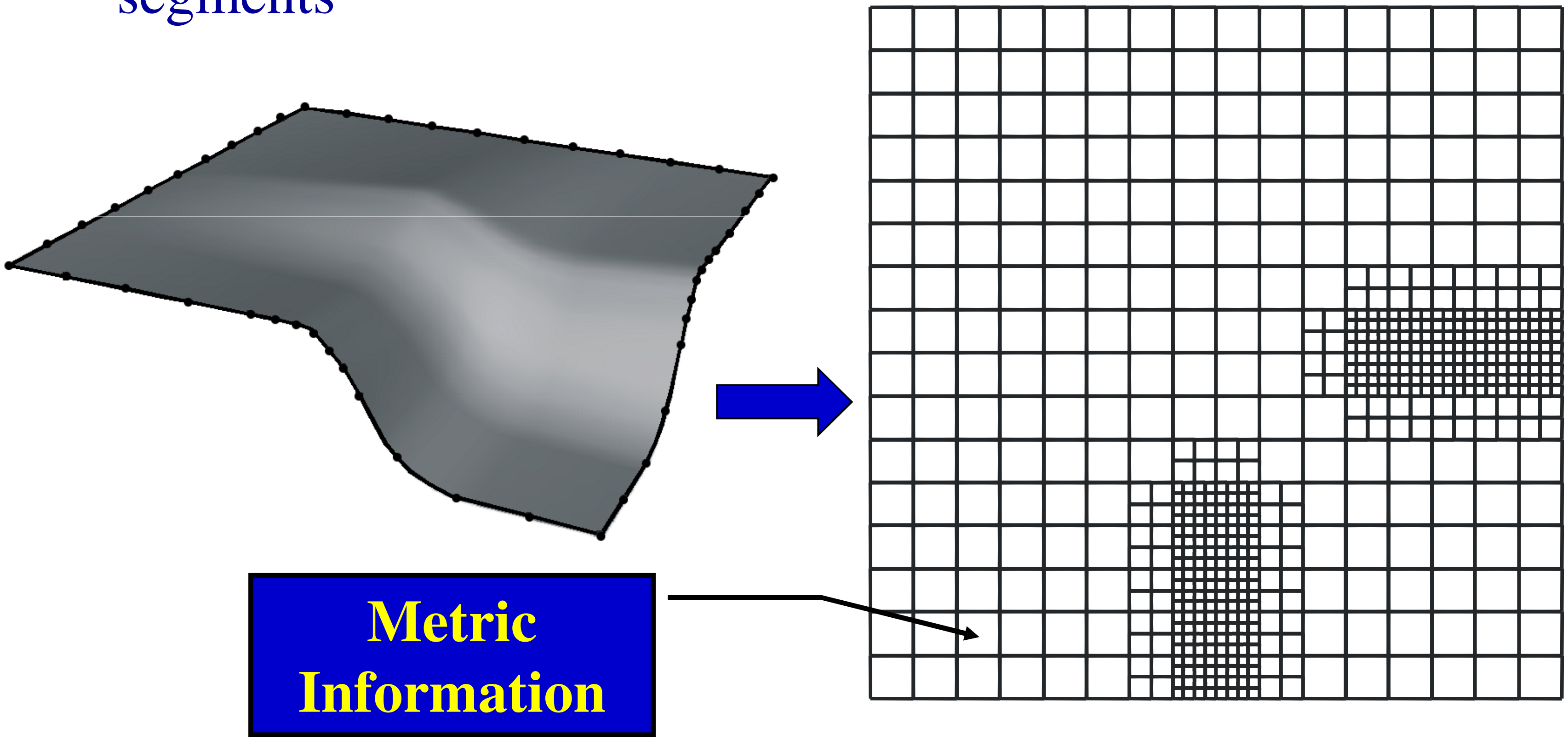
Unstructured mesh – Surface Meshing

- **Parametric Space Meshing**
  - **Elements formed in 2D using parametric representation of surface**
  - **Distance and angles are distorted in parametric space**
  - **Nodes locations later mapped to 3D space**

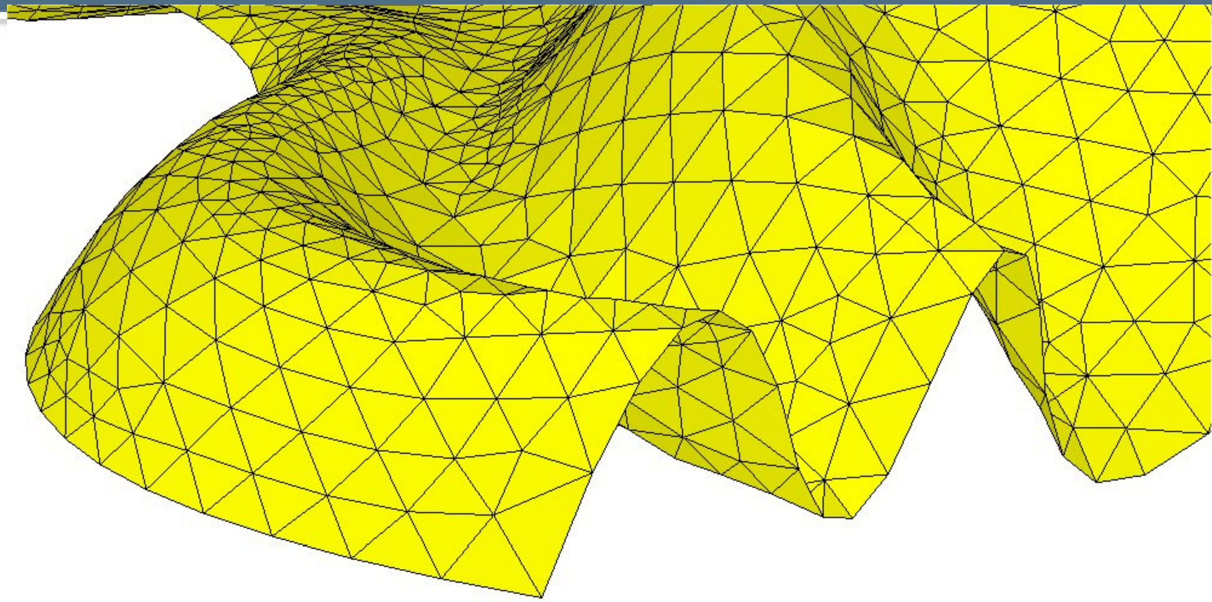




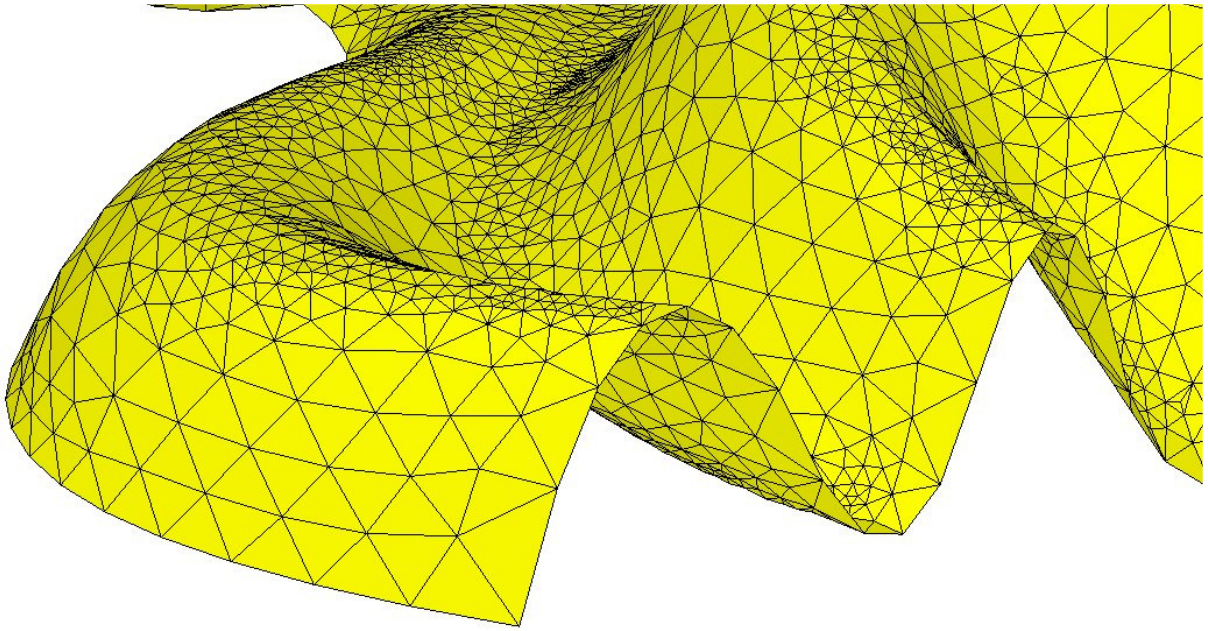
- **Parametric Space Meshing**
  - Given an analytical surface description and boundary segments
  - **Background quadtree**



Importance of considering the curvature



No consideration of curvature



Consideration of curvature