Weierstraß-Institut für Angewandte Analysis und Stochastik

TU Braunschweig

Hang Si

Mesh Generation Techniques

Leibniz Gemeinschaft

Mohrenstrasse 39 10117 Berlin si@wias-berlin.de www.wias-berlin.de March 03, 2006

Content

I. Introduction

II. Triangulation and Point Sets

III. Mesh Generation Methods

IV. Software Implementation Issues

References

- Books
  - “Mesh Generation - Application to Finite Elements”,
    Pascal J. Frey and Paul L. Goerge,
    Hermes Science Publishing, Paris, Oxford, 814 p. 2000.
  - “Geometry and Topology for Mesh Generation”,
    Herbet Edelsbrunner,
    Cambridge Univ. Press, England, 2001.
- Links
  - http://www-users.informatik.rwth-aachen.de/˜roberts/meshgeneration.html
  - http://www.andrew.cmu.edu/user/sowen/mesh.html
  - http://www.imr.sandia.gov
  - http://www-c.inria.fr/Eric.Saltel/gamma

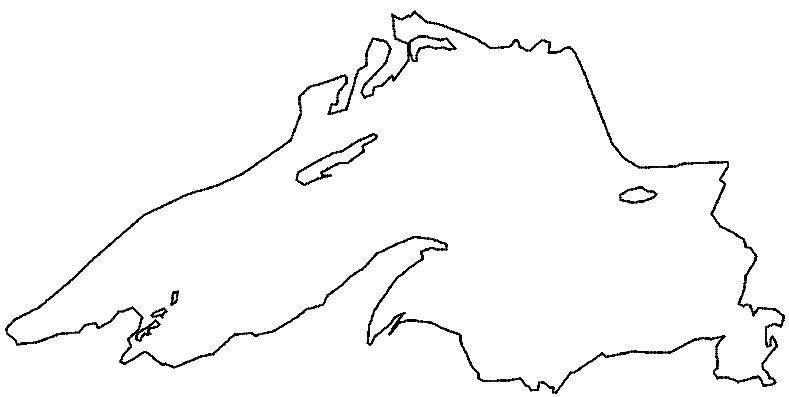
Part I. Introduction

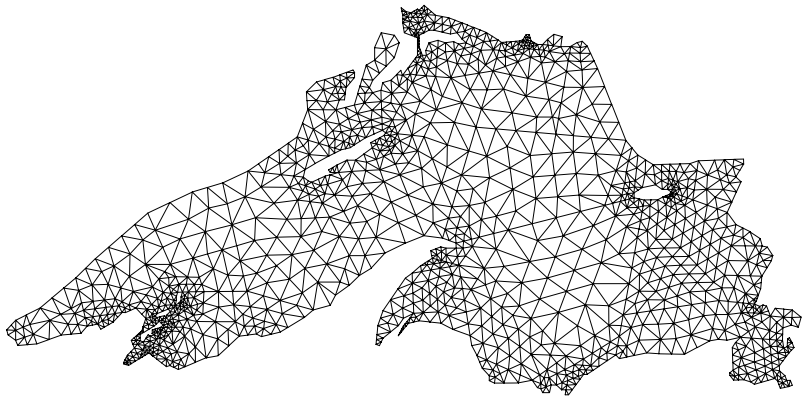
- ▷ What is “mesh generation”?
- ▷ The motivation of studying.
- ▷ Topics in mesh generation.
- ▷ Purpose and scope of this course.

What is “mesh generation”

- Mesh generation is a process of discretizing a geometrical object into simple elements, such as triangles, quads, tetrahedra, hexhedra, prisms, etc.

- Mesh generation techniques help us to reduce the complexity of the task and to make this process done automatically as much as possible.

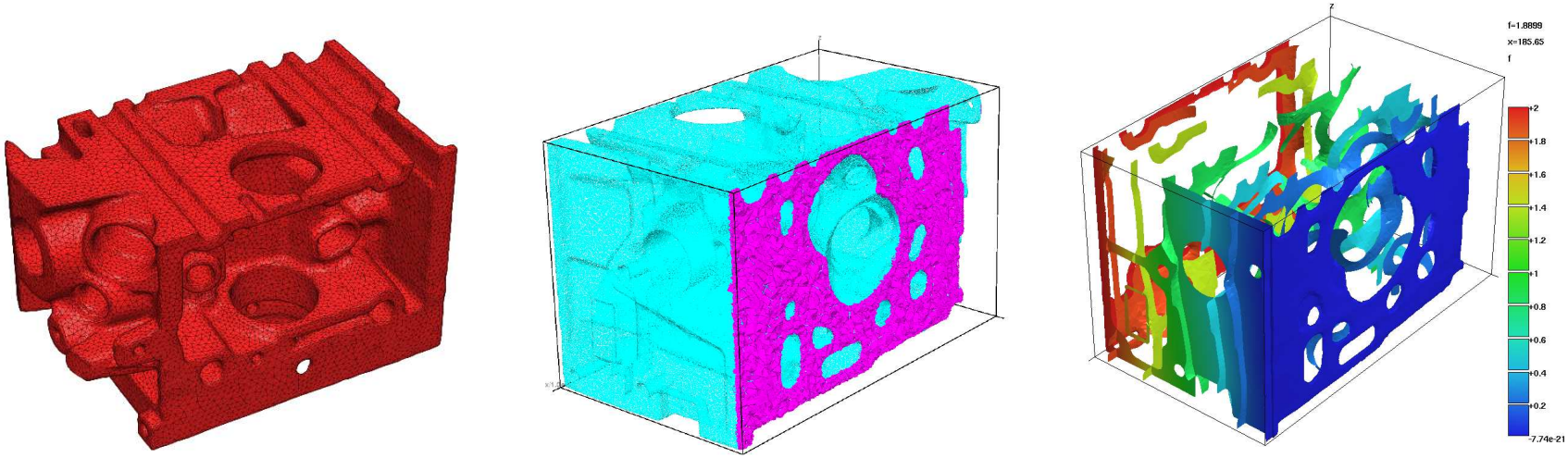




Motivation

- The finite element methods (FEM) and finite volume methods (FVM) are the most popular numerical methods. In these methods, a physical problem (such as heat transfer, electromagnetic radiation, fliud flow, etc) is replaced by a discrete problem that can be actually be simulated in computers.

- Mesh generation is an essiential pre-requisite for any numerical simulation based on FEM and FVM.



Motivation

- Compare to the consensus that FEM and FVM are important topics of study. The meshing technologies are frequently negected as a full topic of interest.
  - "... Let $\mathcal{T}_h$ be an admissible triangulation, ...".
  - "... Assume the the mesh size h tends to 0, then ...".

- It turns out in practice that the task of creating a suitable mesh for numerical simulation is absolutely non-trivial. There are few reasons for it:
  - The complexities of the geometrical shapes.
  - The limitations of computer's capabilities.

- Mesh construction is usually a complicated, time-cosuming process. It has become a bottleneck for the applying of numerical methods.

- Despit the importance, our understanding of this field is rather superficial. A systematic study is thus very necessary.

The Mesh Problem

- In the context of numerical simulation, the purpose of generating a mesh is not only to get a discretization of a domain, but also to get a “suitable” mesh for solving the related physical problem.

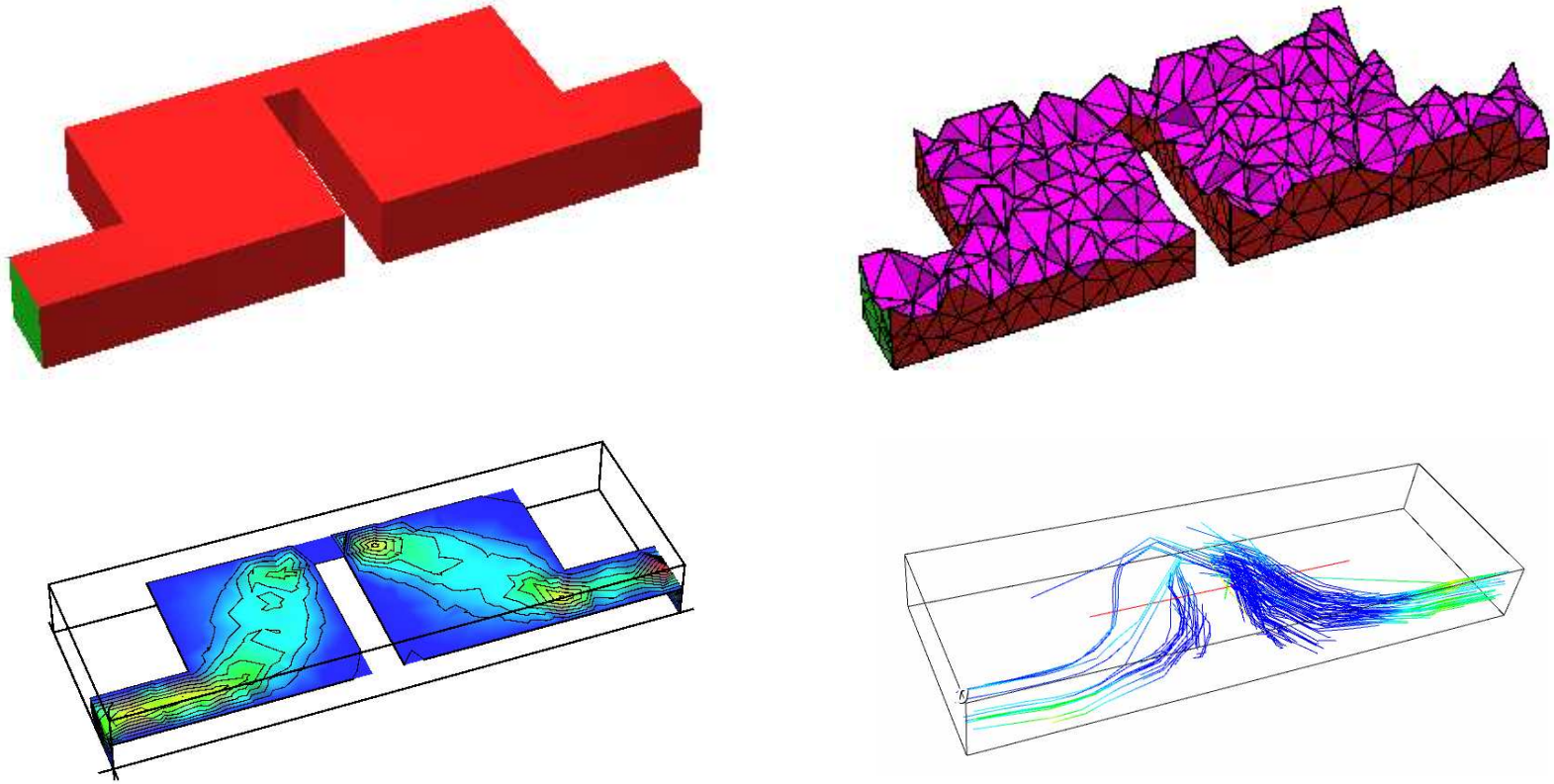
- Example 1.

- Example 2.

- Example 3.

Example 1

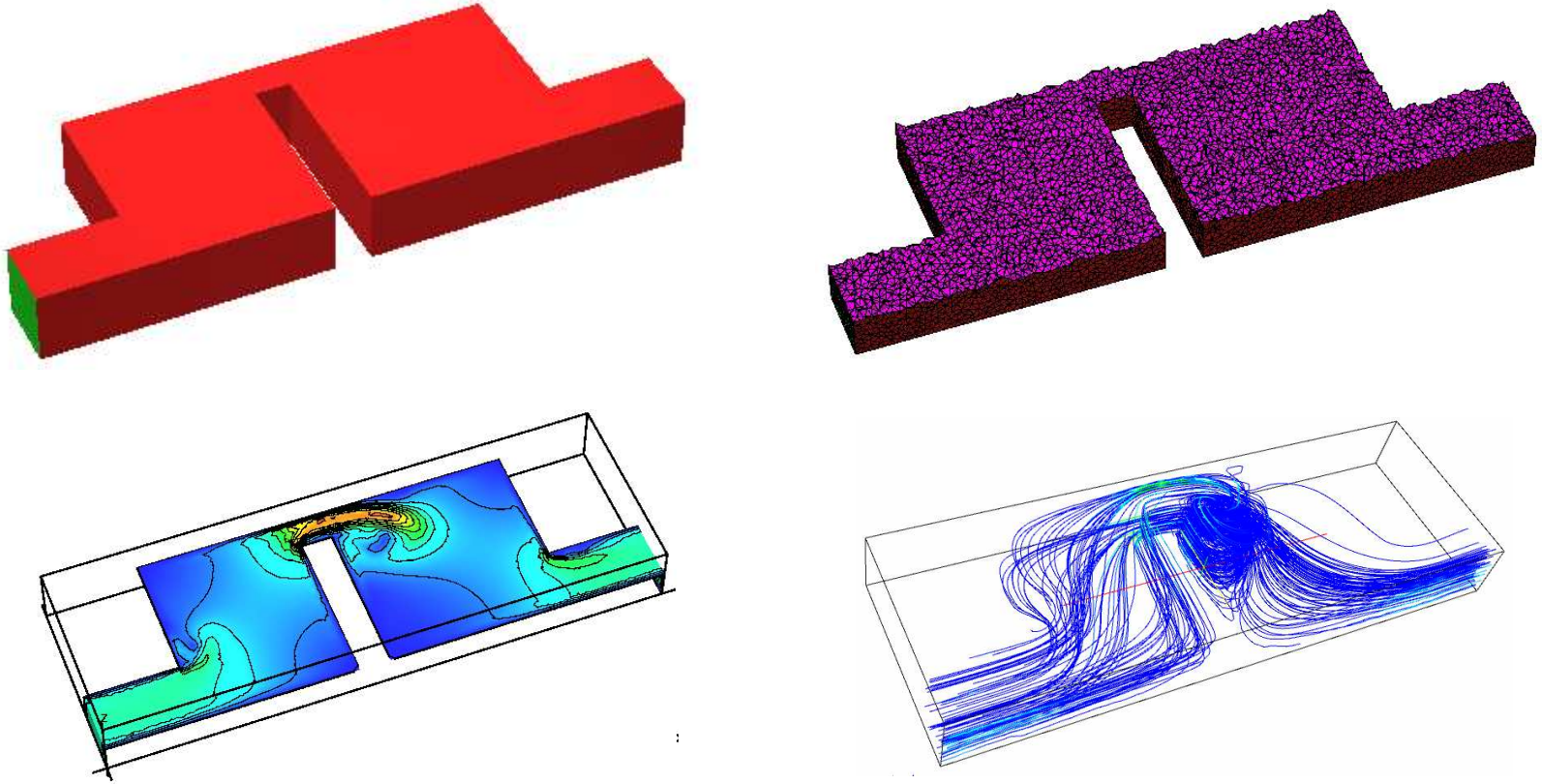
▷ Example 1: a solution of incompressible Navier-Stock eqations.



Top-left: the simulation domain (two chambers connected by a narrow passage). Top-right: the computational tetrahedral mesh (601 nodes, 2,046 tetrahedra). Bottom-left: the plot of density of the magnitude of the velocity on a cutplane. Bottom-right: the stream lines of velocities.

Example 2

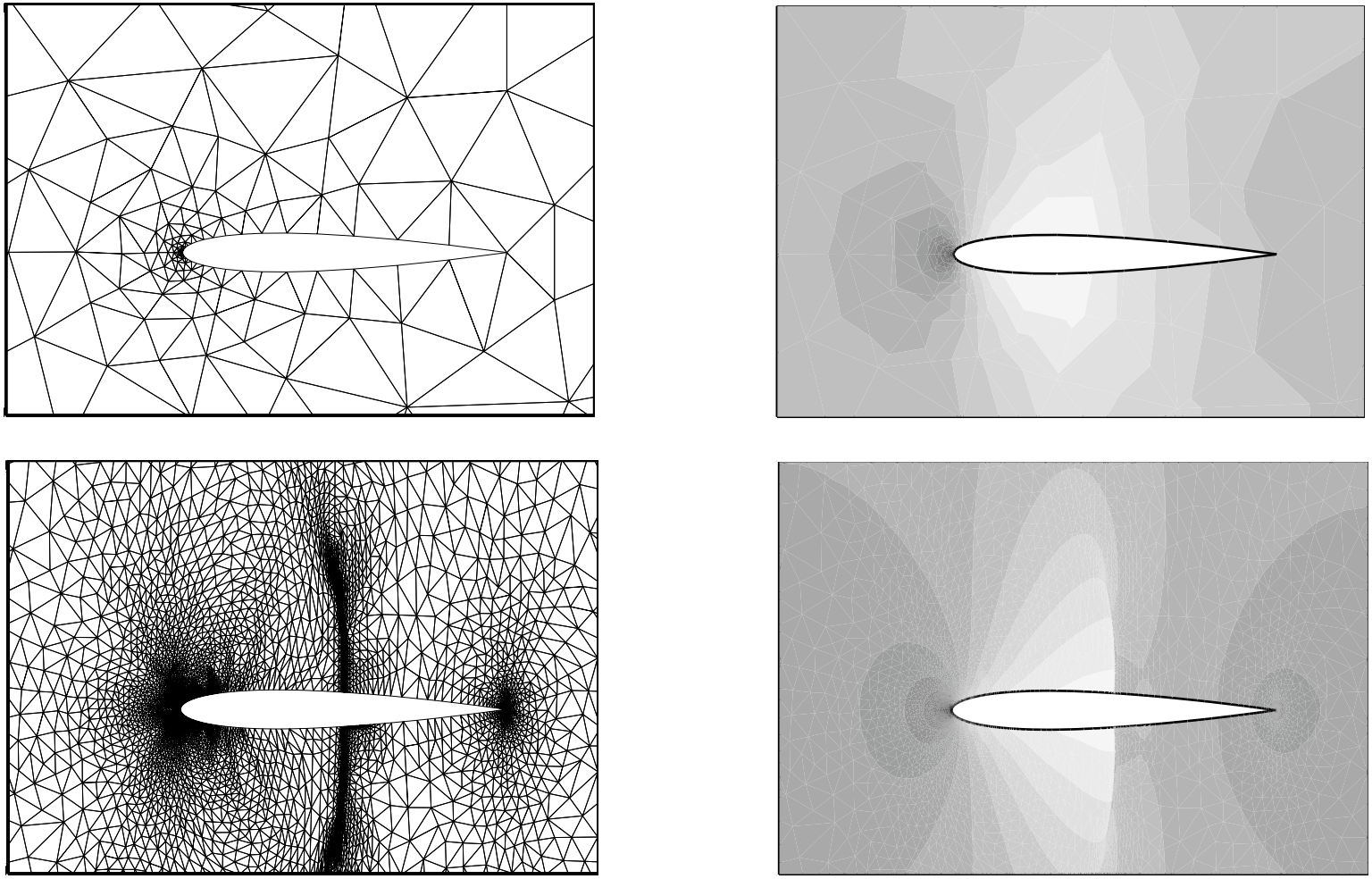
▷ Example 2: a solution of incompressible Navier-Stock eqations.



Top-left: the simulation domain (two chambers connected by a narrow passage). Top-right: the computational tetrahedral mesh (34,230 nodes, 189,485 tetrahedra). Bottom-left: the plot of density of the magnitude of the velocity on a cutplane. Bottom-right: the stream lines of velocities.

Example 3

▷ Example 3: the computation of Euler flow around a wing.



BAMG (http://www-rocq1.inria.fr/gamma/cdrom/www/bamg/eng.htm)

Purpose and Scope

- Mesh generation is an inter-disciplinary research. It involves several different fields of study:
  - Mathematics (geometry and topology);
  - Computer science (data structure and algorithm);
  - Engineering (numerical simulation).

- This is an introduction course of mesh generation. There is no requirement of knowing special knowledge.

- Topics in this course including basic conceptions in mesh generation, meshing technologies recently developed.

- The technologies discussed in this course are for simplical meshes, i.e., mesh elements are triangle or tetrahedra.

- The purpose is to help graduate students who are facing meshing problems to make the task of mesh construction more efficient.

Part II. Triangulation and Point Sets

- Basic definitions
  - Simplical complex
  - Triangulation
  - Delaunay triangulation
  - Constrained triangulation

- Triangulation algorithms
  - Gift-wrapping
  - Flipping
  - Incremental

Basic definitions

- Let S be a finite set of n points in $\mathbf{R}^d$. S is *convex* if for any two points in S, the line segment between them is also cotained in S. The *convex hull* of S, denoted as conv(S) is the intersection of all convex sets in $\mathbf{R}^d$ that contains S.

- A *k-simplex* is the convex hull of a collection of $k+1$ affinely independent points. For example, a 0-simplex is a point, 1-simplex is an edge, 2-simplex is a triangle, and 3-simplex is a tetrahedron. Let σ be a simplex, any subset of the vertices of σ is agian a simplex, we call it a *face* of σ.

- A *simplical complex* K in $\mathbf{R}^d$ is the collection of simplices such that (1) $\sigma \in K$ implices every face of σ is in K, (2) the intersection of any 2 simplices is a face of each.

Basic definitions

- A *triangulation* $\mathcal{T}$ of S is a simplical complex, and the set of simplices of $\mathcal{T}$ covers the conv(S).

- A triangulation $\mathcal{T}$ of S does not necessarily use all vertices of S. In the following, we only consider the case when S is the point set of $\mathcal{T}$.

- Given a point set S, an interesting question to ask:
  - How many triangulations are there?

Basic definitions

- Among the different possible types of triangulations, the Delaunay triangulation is of great interest.

- Given a point set S in $\mathbf{R}^d$. A simplex σ is *Delaunay* if there exists a ball that passes through the vertices of σ and contains no other vertices of S.

- The triangulation $\mathcal{D}$ is a *Delaunay triangulation* of S if every simplex of $\mathcal{D}$ is Delaunay.

- Properties of Delaunay triangulations: exist in any dimension, uniqueness, ...

Basic definitions

- For each $s \in S$ define $V(s)$ to be a collection of points such that for all $x \in V(s)$ and $y \in S - \{s\}$, dist$(x, s) \leq$ dist(x, y). The *Voronoi diagram* of S is the collection $\{V(s) | s \in S\}$.

- From the Delaunay triangulation of S, we can obtain the corresponding Voronoi diagram of S. They're geometrical dual. Voronoi diagram is an idea partition for finite volume methods.

- A d-dimensional Delaunay triangulation is the lower convex hull of a $(d+1)$-dimensional polytope after projecting back to dimension d. One can obtain Delaunay triangulation by constructing a higher dimensional convex hull.

Basic definitions

- Let S be a set of point in $\mathbf{R}^2$, together with a set of line segments, L, each connecting two points in S. We require that any two line segments are either disjoint or meet at a common endpoint.

- A *constrained triangulation* of S and L is a triangulation of S that contains all line segments of L as edges.

- Let $\mathcal{T}$ be a constrained triangulation of S and L. Points $x, y \in S$ are *visible* from each other if xy contains no points of S in its interior and it shares no interior point with a constraining line segment.

- A triangulation $\mathcal{T}$ is a *constrained Delaunay triangulation* of S and L if it is a constrained triangulation and every simplex of $\mathcal{T}$ is constrained Delaunay.

Triangulation algorithms

- Triangulations are widely studied in computational geometry. Many usable and efficient algorithms are available for constructing triangulations, Delaunay triangulations both in two and higher dimension.

- Three fundamental algorithms for constructing Delaunay triangulations are discussed in detail.
  - Gift-wrapping,
  - Edge-flipping,
  - Incremental flip.

Gift-wrapping

Let S be a set of $n \geq 3$ points in $\mathbf{R}^2$. The gift-wrapping algorithm works as follows. Let $\mathcal{T}$ be the current triangulation and Q is a queue of edges. At the beginning, both $\mathcal{T}$ and Q are empty;

Algorithm(Gift-wrapping in E^2)
initialization:
find $p_0, p_1, p_2 \in S$ which form a Delaunay triangle;
$\mathcal{T} := \mathcal{T} \cup \{(p_0, p_1, p_2)\}$;
$Q := Q \cup \{(p_i, p_j), (p_1, p_2), (p_2, p_0)\}$;
repeat:
remove an edge (p_i, p_j) from Q;
find $p_k \in S$ such that p_i, p_j, pk form a Delaunay triangle;
if p_k exists, **then**
$\mathcal{T} := \mathcal{T} \cup \{(p_i, p_j, p_k)\}$;
if (p_i, p_k) is new in $\mathcal{T}$ **then** $Q := Q \cup \{(p_i, p_k)\}$ **endif**
if (p_j, p_k) is new in $\mathcal{T}$ **then** $Q := Q \cup \{(p_j, p_k)\}$ **endif**
endif
until $Q = \emptyset$;

Flipping

- A *flip* is a local geometrical change between a small set of points. For example, let a, b, c, and d in $\mathbf{R}^2$ such that the quadrilateral $abcd$ is convex. Then there exists exactly two ways to triangulate the quadrilateral, each has two triangles. You can obtain the two triangulations by switching the diagonal of the quadrilatera We call this operation an "edge flip".

- Let S be a set of points in $\mathbf{R}^2$, Let $\mathcal{T}$ be a triangulation of S. An edge $ab \in \mathcal{T}$ is *locally Delaunay* if: (1) it belongs to only one triangle and therefore bounds the convex ull, or (2) it belongs to two triangles, abc and abd, and d lies outside the circumcircle of abc.

- A locally Delaunay edge is not necessarily an edge of Delaunay triangulation.

- If every edge of $\mathcal{T}$ is locally Delaunay, then $\mathcal{T}$ is a Delaunay triangulation of S.

Flipping

We can use edge flips as elementary operations to convert an arbitrary triangulation $\mathcal{T}$ into the Delaunay triangulation. The algorithm uses a queue Q and maintains the invariant that unless an edge is locally Delaunay, it resides in queue. Initially, put all edges in Q.

Algorithm(Edge flipping)
 initialization:
    add all edges of $\mathcal{T}$ in Q.
 repeat:
    pop edge ab from Q;
    if ab is not locally Delaunay **then**
      flip ab to cd;
      for $xy \in \{ac, cd, bd, da\}$ **do**
        if xy is not in Q **then**
          add xy in Q;
        endif
      endfor
    endif
 until $Q = \emptyset$;

Flipping

- For a given point set S, we can form a *flip graph* of all the triangulations of S. Each triagulation is a node, and there is a directed arc from node μ to node ν if there is a flip that changes the triangulation μ to ν. The direction of the arc corresponds to our requirement that flip substitutes a locally Delaunay edge for one that is locally Delaunay. The running time analysis implies that the flip graph is acyclic and that its undirected version is connected. If we allow flips in either direction, we can go from any triangulation to any other triangulation in less than n^2 flips.

- **Open problem** Is the flip graph of a finite set $S \subseteq E^3$ connected if no four points in S are coplanar?

Incrmental flip

We obtain a fast algorithm for constructing Delaunay triangulation if we interleave flipping edges with adding points. Denote the points in S in E^2 as $p_1, p_2, ..., p_n$ and assume general position. The algorithm adds the points in sequence. After adding a point, it uses edge flips to satisfy the Delaunay lemma before the next point is added.

Algorithm(Incremental-flip)
for $i = 1$ **to** n **do**
  find $\sigma_{i-1} \in \mathcal{D}_{i-1}$ containing p_i;
  add p_i by splitting σ_{i-1} into three;
  while exists ab not locally Delaunay **do**
    flip ab to another diagonal cd;
  endwhile
endfor

Other Algorithms

- The divide-and-conquer algortihm was the first worst-case optimal Delaunay/Voro alforithm in two dimensions. It splits the set of points into two halves of the same size with a vertical line, recursively computes the Delaunay triangulation of the two halves, and then merges the two triangulations. It has time complexity $\Omega(n \log n)$.

- The Boywer-Watson incremental algorithm is similar to the incremental flip algorithm. After inserting a point, instead of using flip, it recovers the Delaunay triangulation by forming a star shaped cavity then retriangulate it.

- A Delaunay triangulation can always be computed by lifting the set of points and using a convex hull algorithm.

Implementation Issues

- It is not trivial to obtain a reliable and complete implementation of any of the Delaunay triangulation algorithms. The main problems are:
  - The rounding error caused by computer's floating-point arithmetics.
  - The degenerate cases.

- A fundamental implementation decision is the choice of exact or floating point filter. Exact arithmetic is known to obtain a completely reliable implementation. However, it is costy and slow. Floating-point filter is not guaranteed to be completely reliable. However, it is the typical choice and may be reliable enough in some circumstances.

- The degeneracies can be handled by the techniques of perturbation, either geometrically or symbolically.

Part III. Mesh Generation Methods

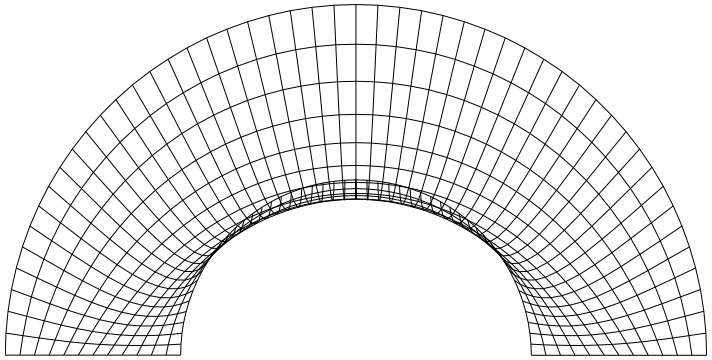
- ▷ Basic conceptions
- ▷ Overview of mesh generation methods
- ▷ Quality Delaunay mesh algorithms

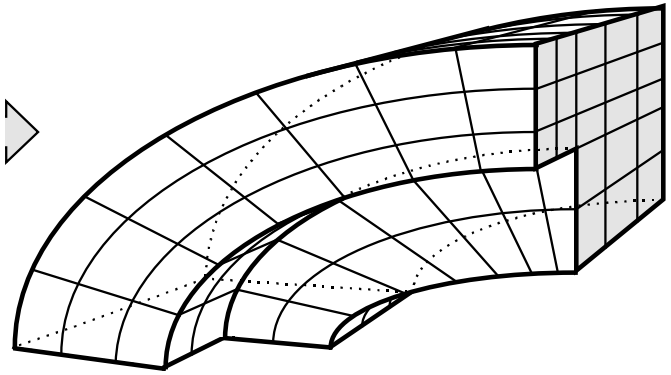
Structured Meshes

- A *structured mesh* is one in which all interior vertices are topologically alike. In two dimensions, it is often simply a square grid, and is usually a cubic grid in three dimensions.

- The advantages of structured mesh are:
  - easy to store (by defining the neighborhood connectivity implicitly);
  - efficient to access neighboring cells (by incrementing/decrementing indices).

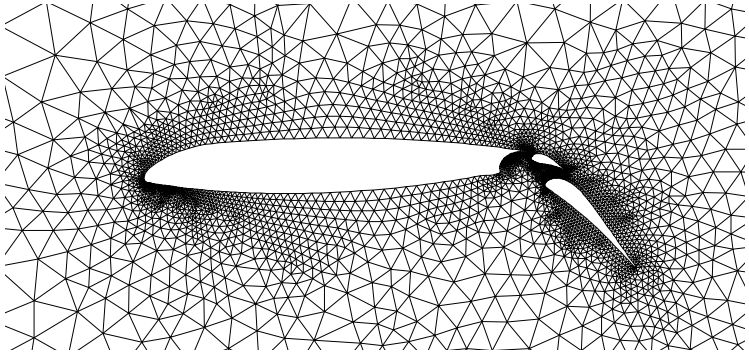
- The disadvantages of structured mesh are:
  - difficult or impossible to compute for complicated domains, not flexible.
  - may require much more elements than an unstructured mesh for the same problem.

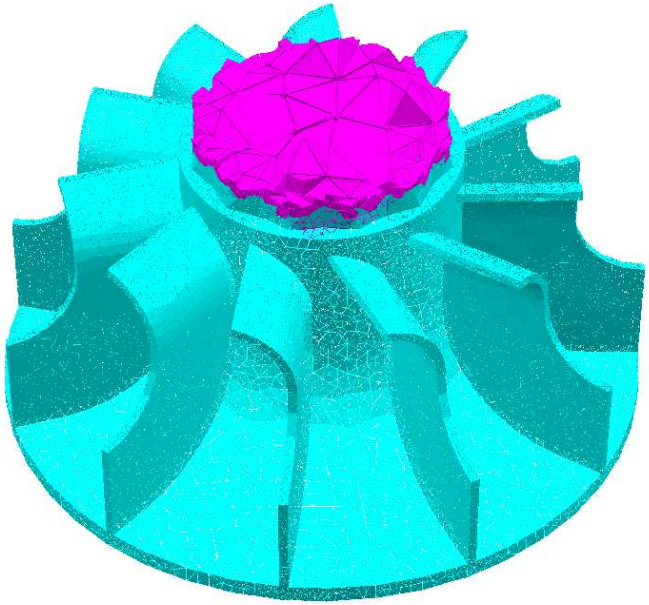




Unstructured Meshes

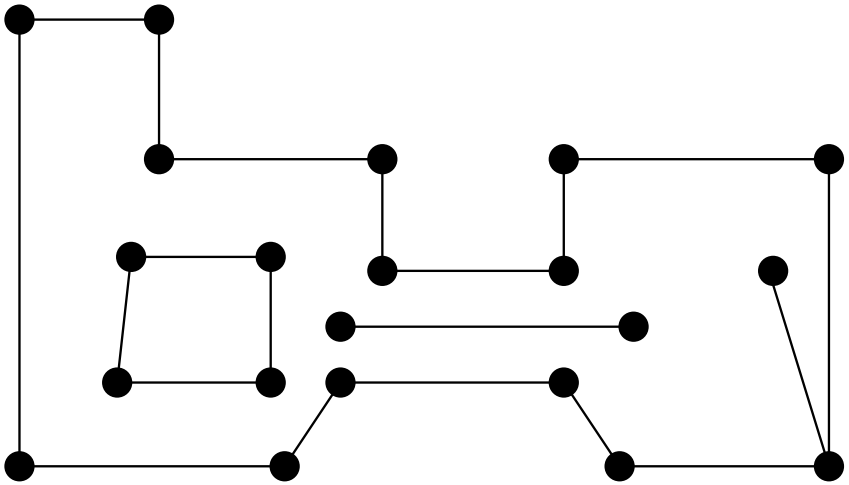
- An *unstructured mesh* is characterized as the local connections vary from points to points. Such a mesh is usually composed of triangular mesh in two dimensions and tetrahedra mesh in three dimensions.
- The *advantages* of unstructured meshes: flexibility in fitting complicated domains, rapid grading from small to large elements, and easy refinement and derefinemen
- Later on we will only discuss unstructured mesh generation methods.



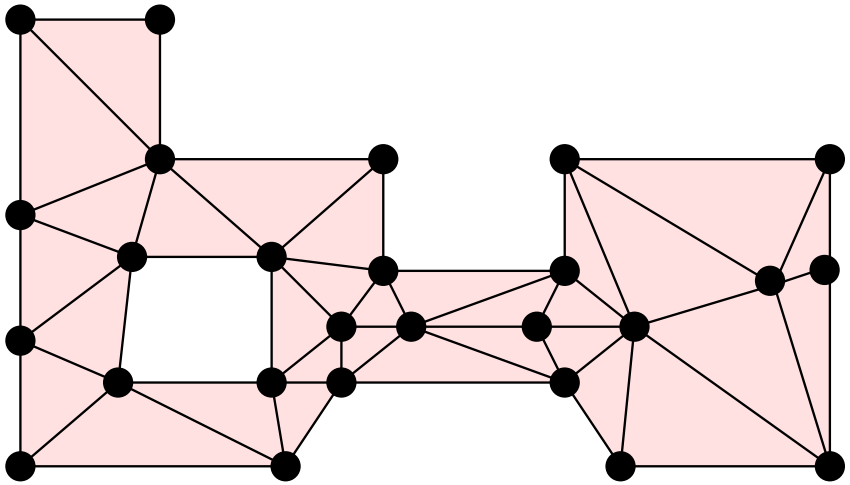


A *planar straight-line graph* (PSLG) [Chew'89] is a set of vertices in $\mathbf{R}^2$ along with a set of non-crossing segments, possibly with dangling segments, and isolated vertices.

A PSLG can be used to define a 2D mesh domain, possibly with holes.



A PSLG G



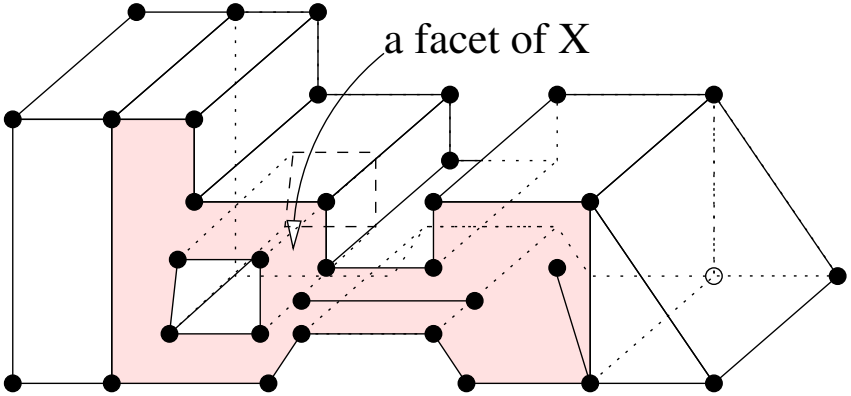
The triangular mesh of G

Three-dimensional Input Domains

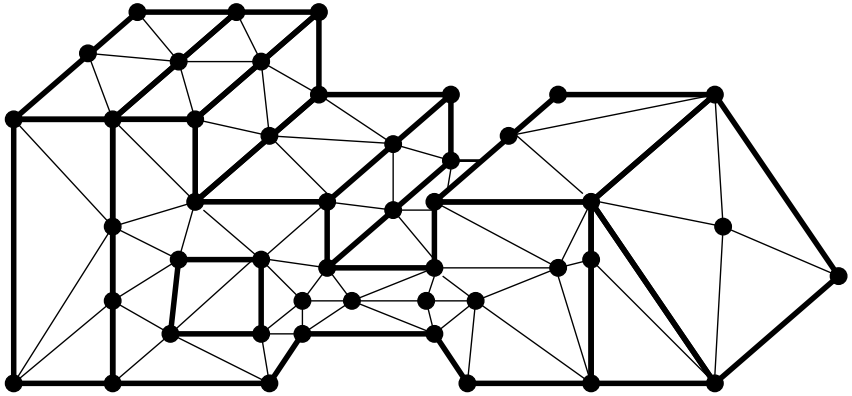
A *piecewise linear complex* (PLC) [Miller et., al.'96] is a set of vertices in $\mathbf{R}^3$, along with a set of segments and *facets*. Each facet is indeed a PSLG embedded in 3D.

The intersection of any two components of a PLC is either empty or a union of components of the PLC itself.

A PLC X can be used to define a 3D mesh domain Ω. The *boundary* of Ω consists of facets, segments, and vertices of X.



A PLC X



The tetrahedral mesh of X

Element shape

- The *aspect ratio* of an element is the ratio of it maximum to its minimum width, where width reders to the distnace between parallel supporting hypeplanes.

- In general, elements of large aspect ratio are bad. They leads to poorly conditione matrices, worsening the speed and accuracy of the linear solver. Moreover, large aspect ratio may given unacceptable interpolation error.

- However, elements of large aspect ratio are good when the solution to the PDEs is anisotropic. A good mesh for such problem requires properly aligned high aspect ratio elements.

Control Space

- The numerical problem imposes special conditions on the mesh. The conception control space is introduced for this purpose.

- Definition. (Δ, H) is a *control space* for the mesh of a given domain Ω if:
  - Δ covers Ω,
  - a function $H(P, \vec{d})$ is associated with every point $P \in \Delta$, where $\vec{d}$ is the direction.

- A control space includes two related ingredients: a covering triangulation Δ, and a function H. This pair allows for the specification of some properties or criteria t which the elements of the mesh should conform.

Unit Mesh

- The purpose of constructing a mesh is to obtain a good mesh, or each element of the mesh must be well-shaped.

- When a size map (control space) is provided, it is clear that a unit length in the metric associated with the control space is the targeted value.

- A *unit mesh* is a mesh with unit edge lengthes.

- In anisotropic case, a unit lenght in a metric map specifying a value h actually gives a length h in the usual sense and that the same is true in an anisotropic case.

- **Remark** A triangle with unit edge length must be good shaped. However, this is not the case of a tetrahedron.

Classes of Mesh Methods

- Manual or semi-automatical methods.
- Parameterization (mapping) methods.
- Domain decomposition methods.
- Point-insertion / element creation methods.
- Constructive methods

Unstructured Mesh Generation

The generation of unstructured meshes involves the placing of internal points and the preserving of the domain boundaries in the mesh. This usually achieved through different stages that can be summarized as follows:

1. Definition of the domain boundaries.

2. Specification of an element size distribution fucntion (optional).

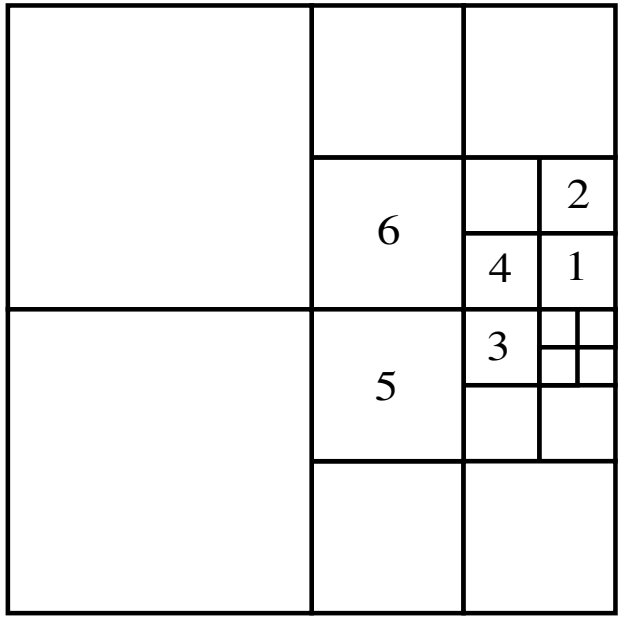
3. Generation of a mesh.

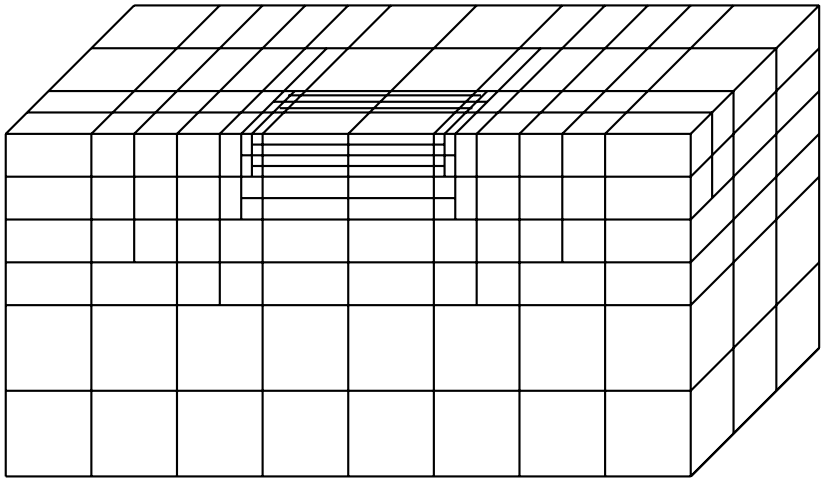
4. Optimization of the element shape (optional).

Step 3 is the main part of the process. Most developed techniques can be categorized into three classes: (1) quadtree/octree method; (2) advancing front method; and (3) Delaunay based method.

Quadtree-octree based methods - Introduction

Quadtee and octree are spatial data structures which are suitable for the task of spatial decompositions. In such approaches, the resulting hierarchical structure serves as a neighborhood space as well we a control space used to prescribe the desired element sizes.





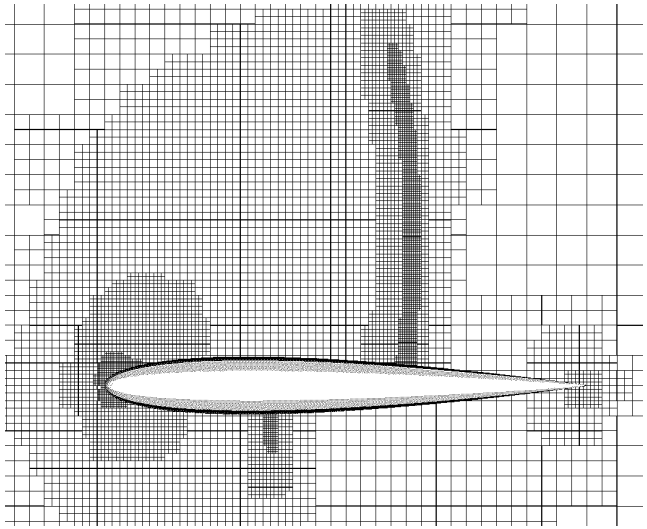
Quadtree-octree based methods - General Scheme

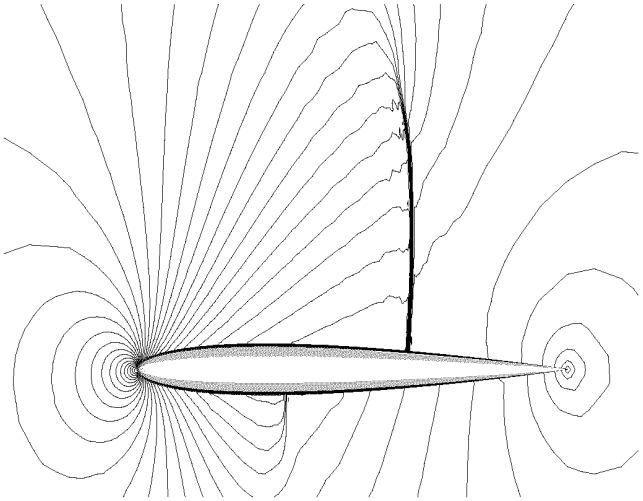
Schematically, a classical quadtrr-octree based technique involves the following steps:

1. Initializations.
   - boundary discretization.
   - definition of size function (optional).
2. Tree decomposition.
   - Initialization: the tree representation is derived from a box enclosing the domain.
   - Recursive subdivision of the box up to a satisfactory criterion.
3. Tree balancing: limit the difference between neighboring cells to only one level (the so-called 2:1 rule).
4. Cell meshing using predefined patterns (internal cells) and local connections (boundary cells)
5. Optimization: topological and geometrical modifications.

Quadtree-octree based methods - Features

- Features:
  - the process of tree creation is simple, efficient;
  - able to produce a set of cells that have a compatible size to the size distribution function;
  - no difficulty to create inetrnal points - field points are chosen at the quadrant (octant) corners;

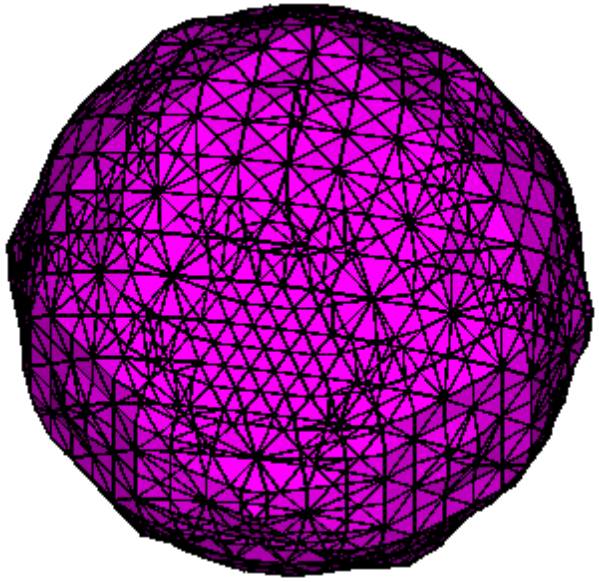


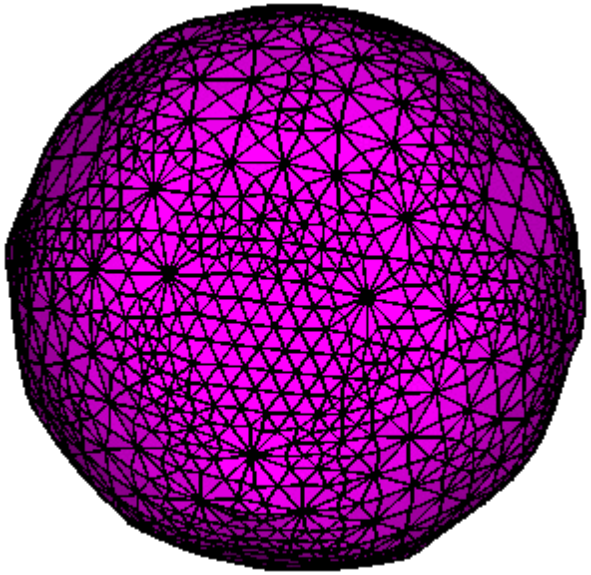


Adapted mesh and isolines of solution for unit span NACA 0012. (M. J. Aftosmis)

Quadtree-octree based methods - Difficulties

- Difficulties:
  - difficult to distinguish two rather closed boundary entities during the tree construction.
  - the intersection computations during the insertion of the entities.
  - the filtering of closely spaced points, which requires a robust algorithm in order to preserve the topology and the geometry of the domain.
- Most of the difficulties only happen in three dimensions.

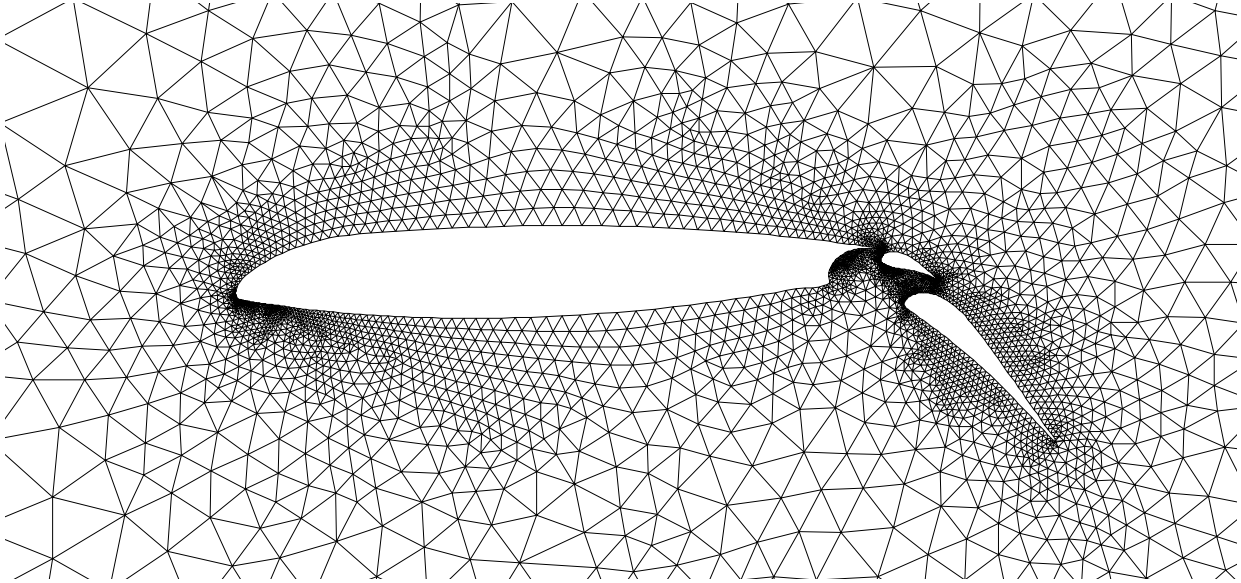




Examples created by software SimBio-Vgrid (http://www.ccrl-nece.de/vgrid/examples/ex.html)

Advancing Front based method

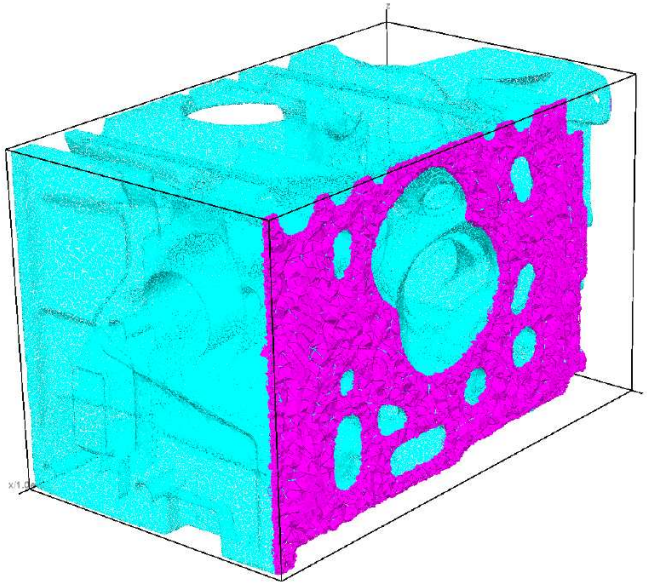
- The main idea of advancing front based method is to construct the mesh elements by element, starting from an initail *front*. The technique proceeds by creating new points and connecting them with some other points of current front so as to construct the mesh elements.
- The technique is iterative. At each element creation, a front is updated and then dynamically evolves. The iterative process terminates when the unfinished front list is empty.

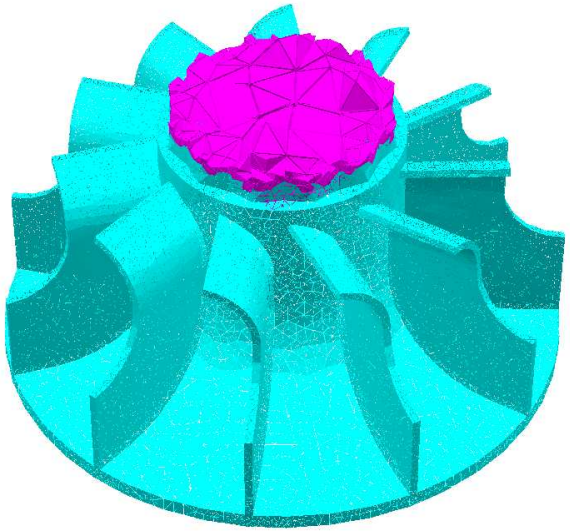


Delaunay Method

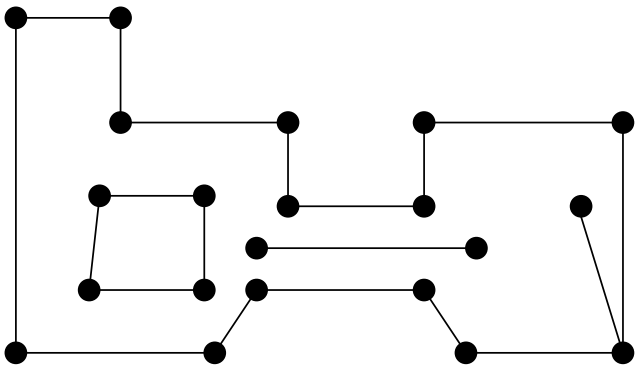
In this approach, a Delaunay triangulation of all boundary points is constructed first. The boundaries is then recovered from the initial triangulation. Then the exterior elements are removed. Finally, the resulting mesh is refined.

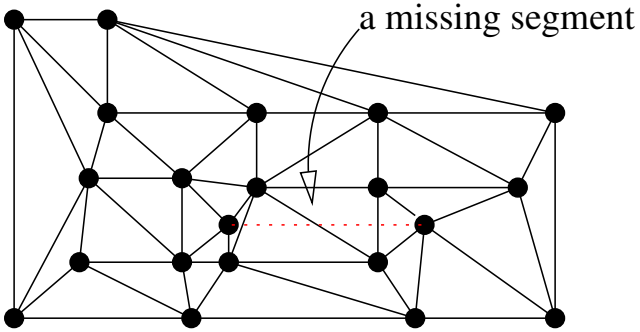
Features The Delaunay method results in a very efficient mesh generation technique. Most of the operations can be achieved in optimal time. In practice, this method appears to be one of the most efficient meshing techniques.





Why Delaunay triangulation is not enough?



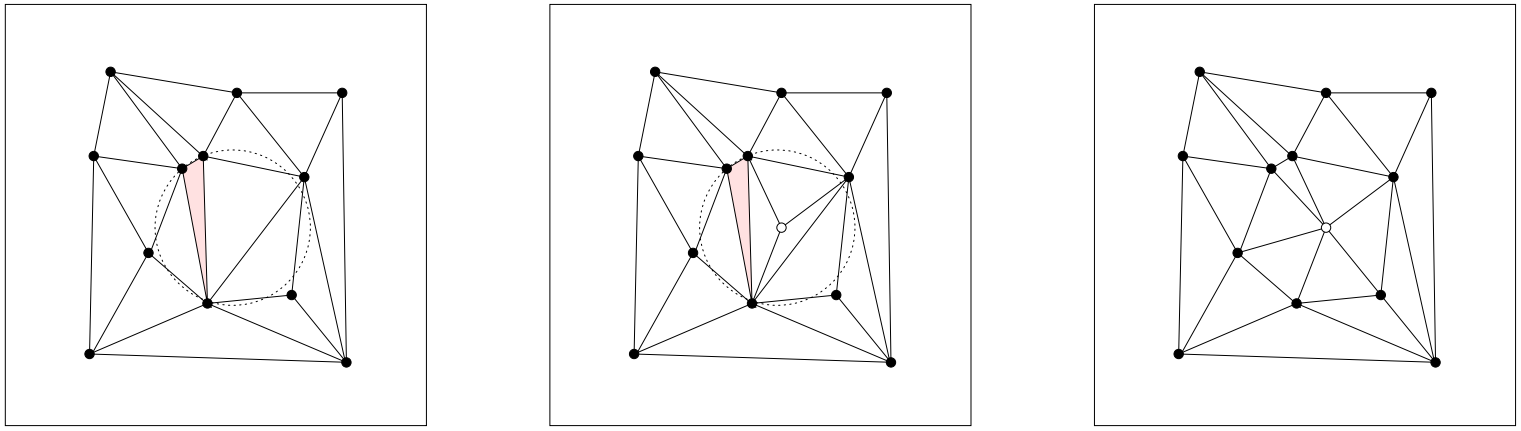


Quality Delaunay Mesh Algorithm

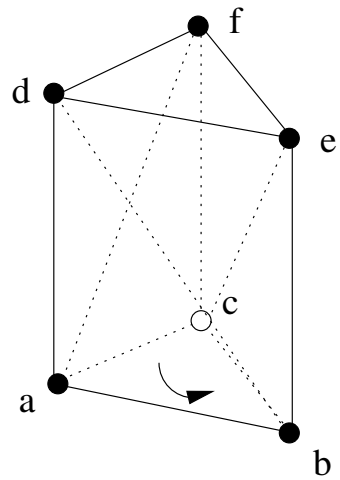
We introduce an algorithm that make the use of Delaunay triangulations in constructing triangular meshes in the plane. It provides “shape” and “size” guarantee

- ▷ All triangles have a good bounded aspect ratio.
- ▷ The number of triangles is within a constant factor of optimal.

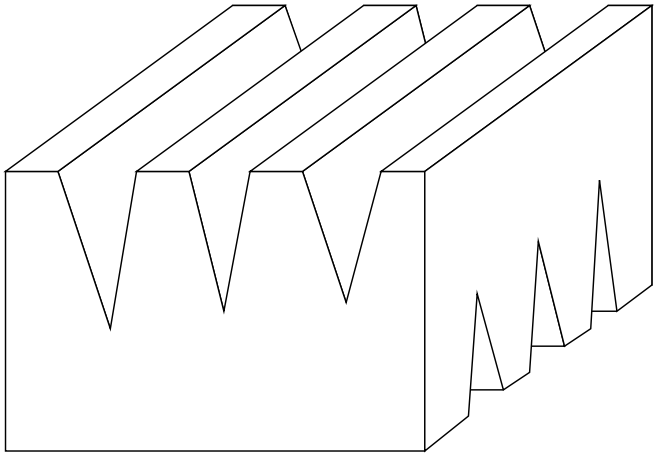
The idea is to add new vertices until the triangulation forms a satisfying mesh. Constraining segments are covered by Delaunay edges.



There are polyhedra which are not terahedralizable without inserting additional points (so-called *Steiner points*).



The Shönhardt polyhedron



A cube with cutting deep wedges

The problem of determining whether a polyhedra (or PLC) has a tetrahedralization without adding points is NP-hard [Ruppert and Seidel'92].

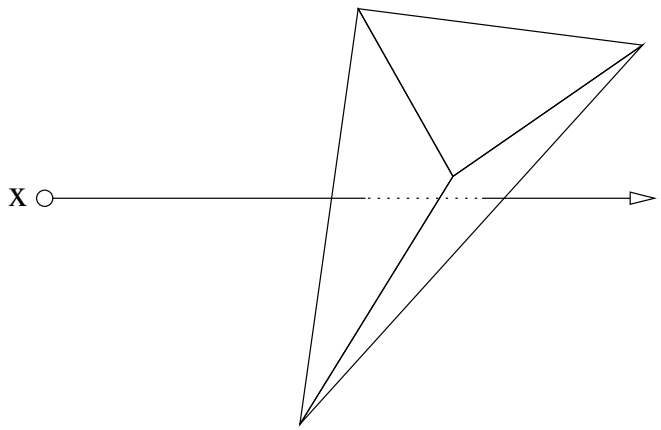
Algorithms for boundary recovery have to insert additional points.

The question: where to insert them?

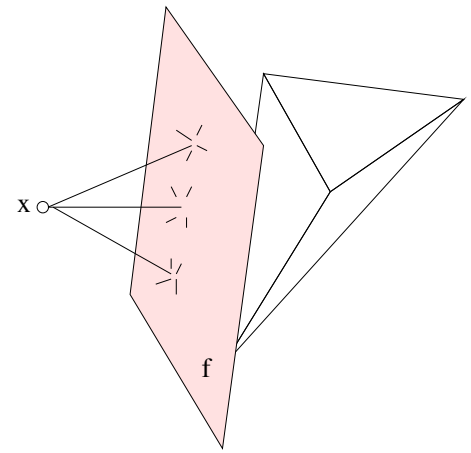
Visibility

Let $x \in \mathbf{R}^3$ be an arbitrary but fixed viewpoint. We say a tetrahedron τ is *visible* by x if there is a half-line starting at x that intersects the relative interior of τ.

We say τ is *invisible* by x if there is a facet F of X, and all the half-lines starting at x first intersect F.



The tetrahedron τ is visible by x

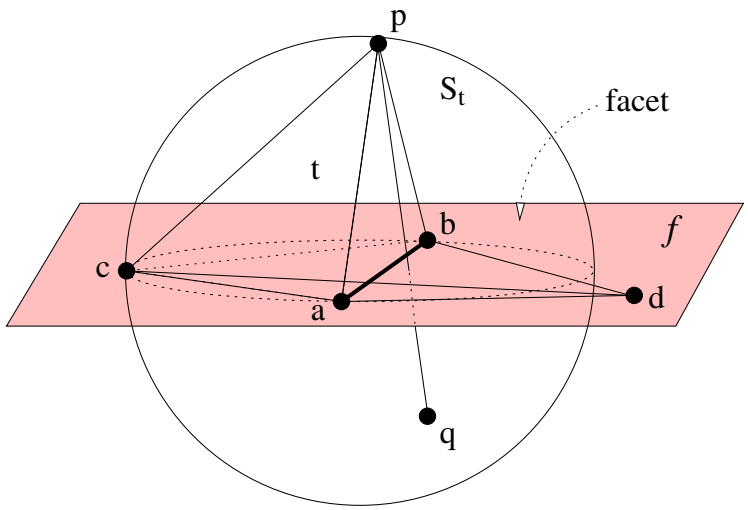


τ is invisible by x (the view is blocked by a facet f in front of τ).

Constrained Delaunay Tetrahedralization

A tetrahedron τ is *constrained Delaunay* if there exists a circumsphere of τ that encloses no vertex of V which is *visible* from its interior. The visibility is blocked by facets of X.

The *constrained Delaunay tetrahedralization* (CDT) $\mathcal{T}$ of X is a tetrahedralization of V, such that every tetrahedron of $\mathcal{T}$ is constrained Delaunay.



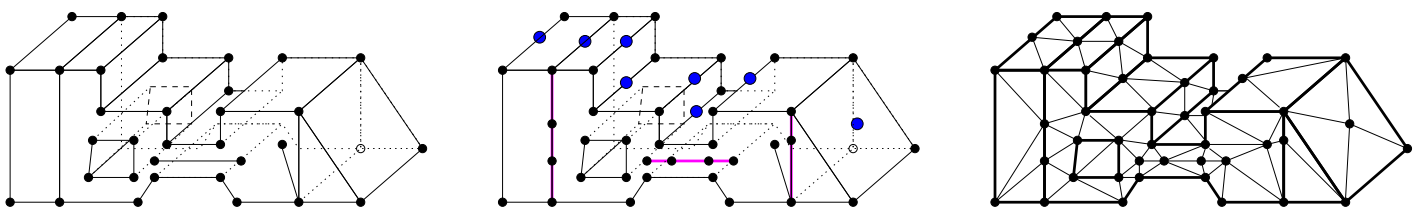
CDTs are suitable structures for boundary recovery problem. They preserve the boundaries and have properties as close as DTs. Moreover, they tend to need less additional points than other methods.

Highlights of the CDT Algorithm

Given a PLC $X(V, S, F)$, where V, the set of vertices, S, the set of segments, and F, the set of facets. The CDT algorithm constructs a new PLC $X'(V', S', F')$ such that:
(1) $V' \subseteq V$;
(2) for every $s \in S$, s is the union of segments in S';
(3) for every $f \in F$, f is the union of facets in F';

X' is created by inserting additional points on the segments of X. We proposed a segment recovery algorithm to recover the segments such that the DT of V' contains all segments of X'.

The facets of X' are recovered in the DT of V' by a facet recovery algorithm. The CDT of X' is constructed after all facets are recovered.



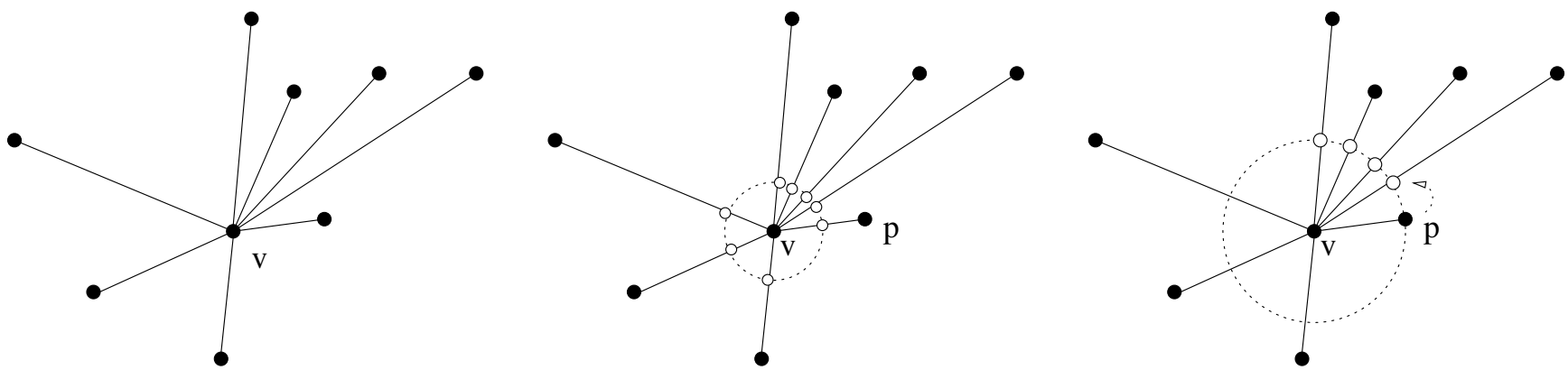
PLC X PLC X' CDT of X'

Segment Recovery - Protecting acute vertices

A vertex v is *acute* if at least two segments incident at it form an angle smaller than $90°$.

A general approach. Create a protecting ball B around the vertex v, the radius r of B is previously chosen to be small enough such that no segment inside B is missing.

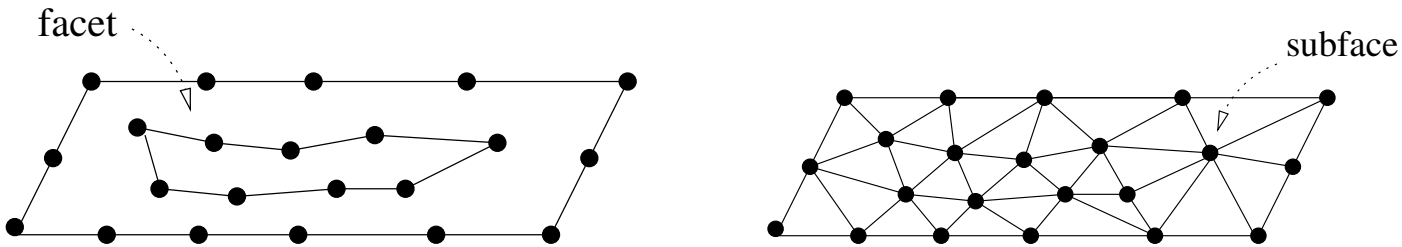
Our segment recovery algorithm will automatically create the protecting ball of v - There is no need to create protecting ball in advance. And the radius the protecting ball is at least lfs$(v)/C$, C is a constant depends on how many segments incident at v.



An acute vertex v The protecting ball B of v The automatically formed B'

Facet Recovery

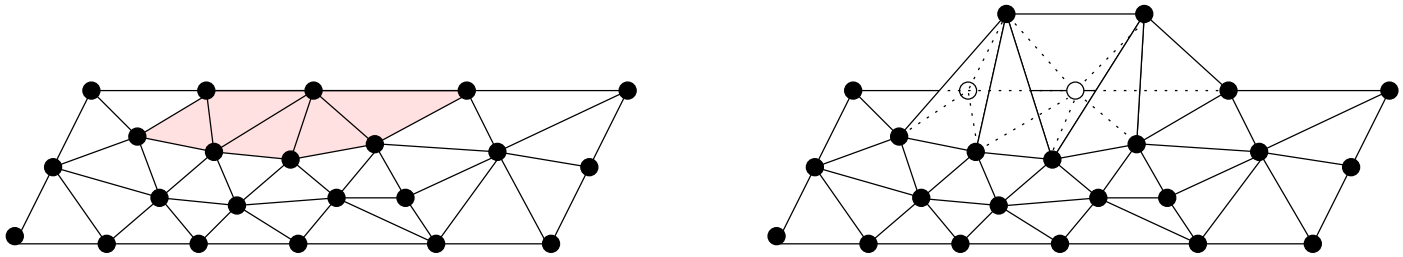
A facet is first traingulated into a set of *subfaces*.



Some subfaces are missing in DT - they are non-Delaunay and crossed by Delaunay edges.

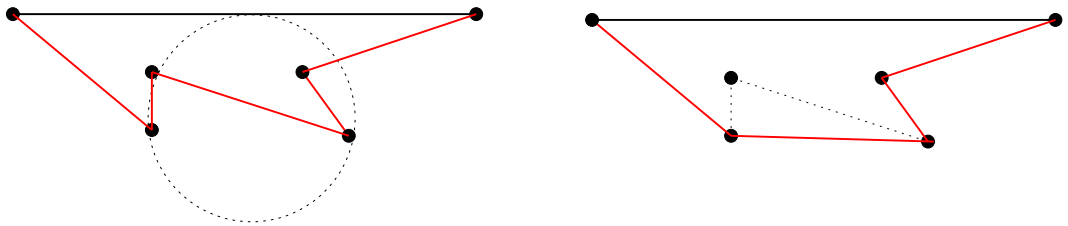
A *missing region* is a set of missing subfaces connect to each other.

From a missing region Ω one can form two cavities in DT, one at each side of Ω.

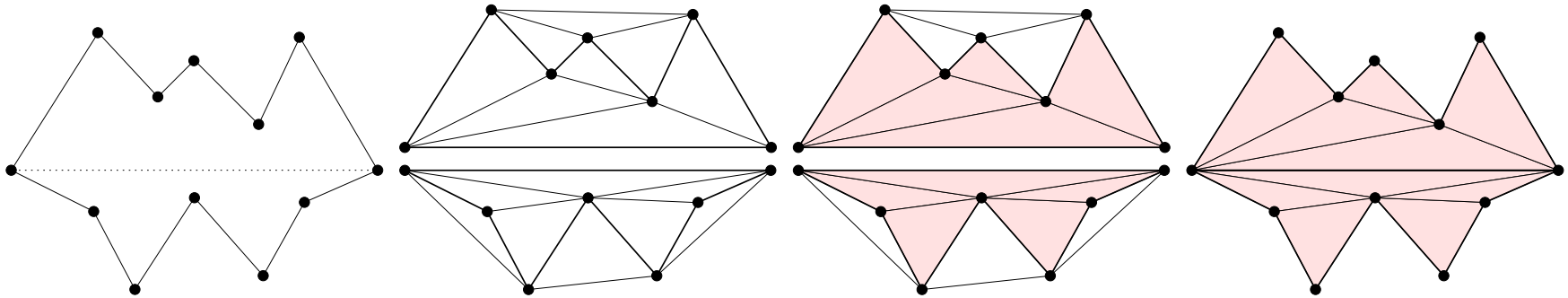


A caviuty C is filled with tetrahedra by the *cavity tetrahedralization procedure*.

1. Verify C. Enlarge C until all faces of C are strongly Delaunay.

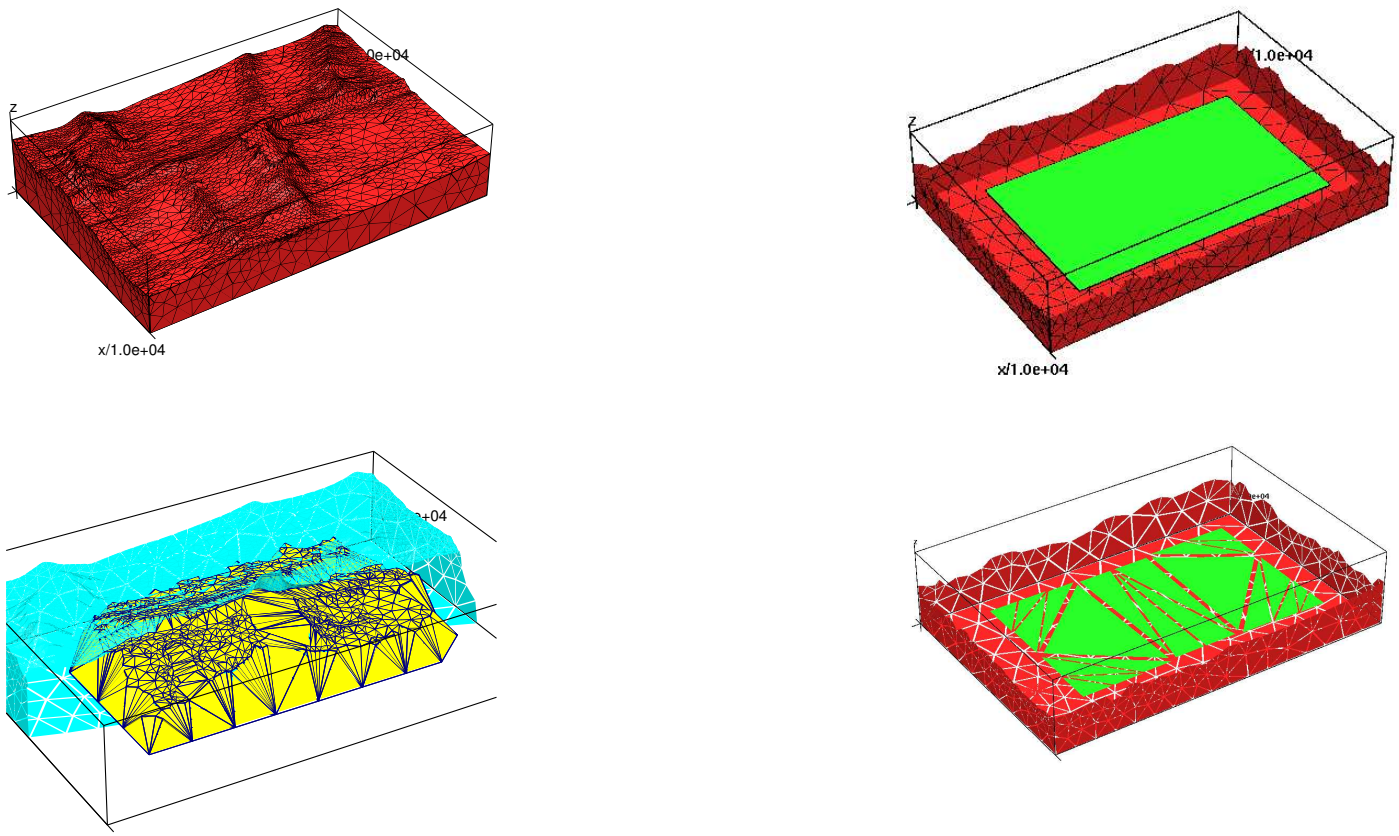


2. Tetrahedralize C.



Examples - Geological Model

Input: 6,116 nodes, 12,219 triangles.
The biggest cavity: 1,832 faces, 918 vertices.
Boundary recovery time: 1.49 second
Output: 8,308 nodes (2,191 points for segment protecting, 2 break points).



Model available from http://www-rocq1.inria.fr/gamma/

Part IV. Software Implementation Issues

- Mesh data structures.
- Sample code from Triangle (2D).
- Sample code from TetGen (3D).