Lecture 2
Unstructured Mesh Generation

MIT 16.930

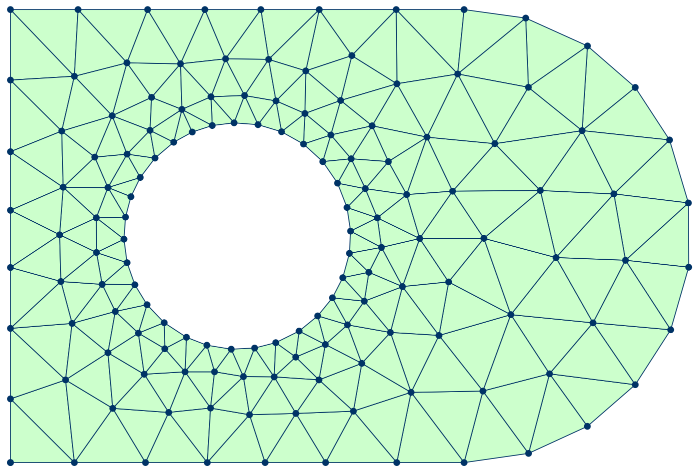
Advanced Topics in Numerical Methods for Partial Differential Equations

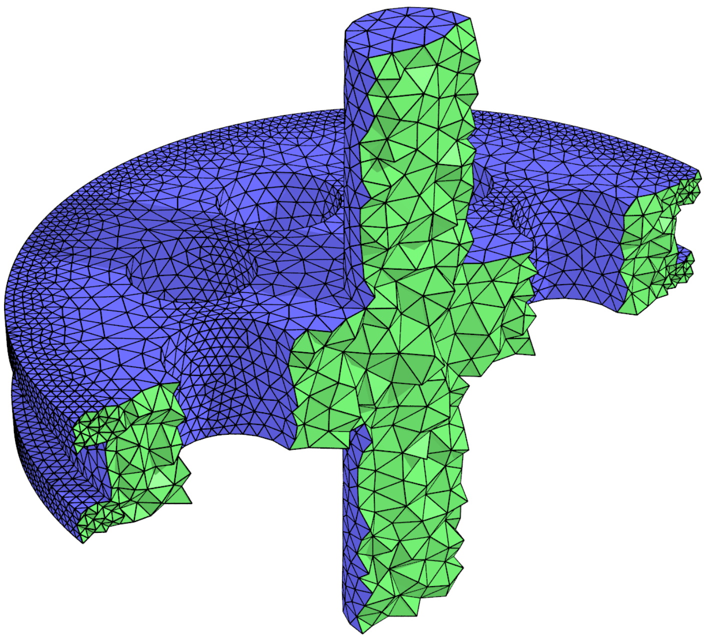
Per-Olof Persson (persson@mit.edu)

February 13, 2006

Mesh Generation

- Given a geometry, determine node points and element connectivity
- Resolve the geometry and high element qualities, but few elements
- Applications: Numerical solution of PDEs (FEM, FVM, DGM, BEM), interpolation, computer graphics, visualization

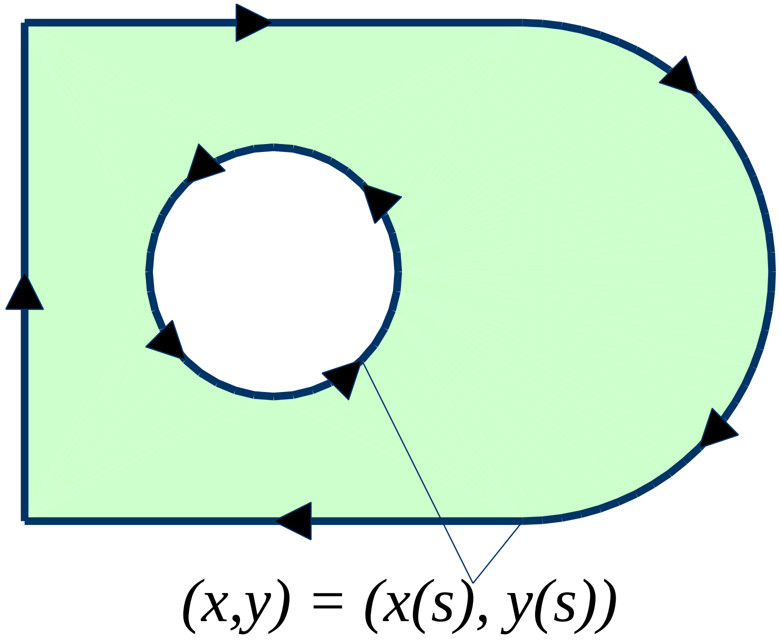




Geometry Representations

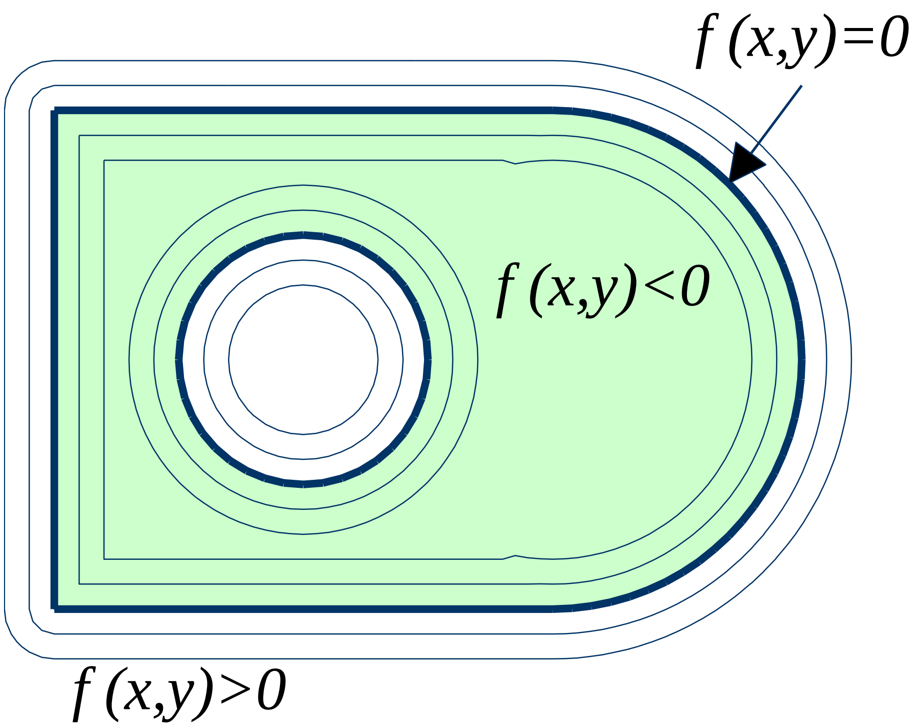
Explicit Geometry

- Parameterized boundaries



Implicit Geometry

- Boundaries given by zero level set



Meshing Algorithms

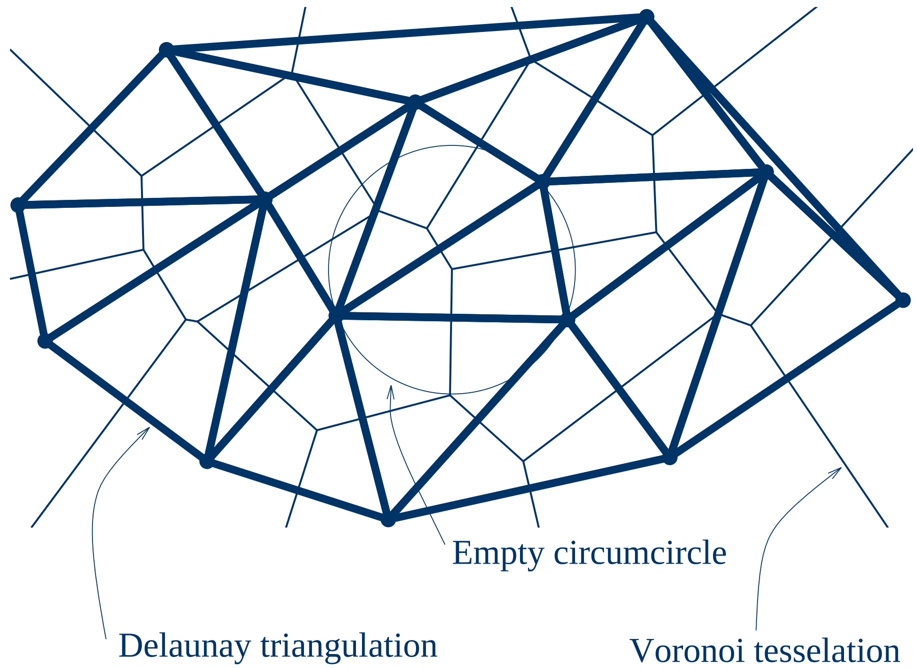
- ***Delaunay refinement***
  - Refine an initial triangulation by inserting points and updating connectivities
  - Efficient and robust
- ***Advancing front***
  - Propagate a layer of elements from boundaries into domain, stitch together at intersection
  - High quality meshes, good for boundary layers, but somewhat unreliable in 3-D

Meshing Algorithms

- ***Octree mesh***
  - Create an octree, refine until geometry well resolved, form elements between cell intersections
  - Guaranteed quality even in 3-D, however somewhat ugly meshes
- ***DistMesh***
  - Improve initial triangulation by node movements and connectivity updates
  - Easy to understand and use, handles implicit geometries, high element qualities, but non-robust and low performance

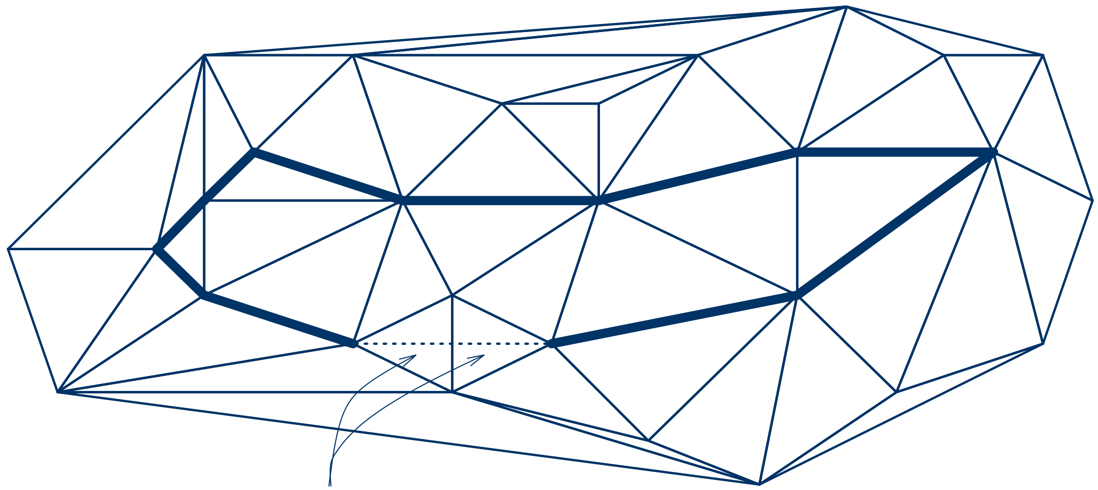
Delaunay Triangulation

- Find non-overlapping triangles that fill the convex hull of a set of points
- Properties:
  - Every edge is shared by at most two triangles
  - The circumcircle of a triangle contains no other input points
  - Maximizes the minimum angle of all the triangles

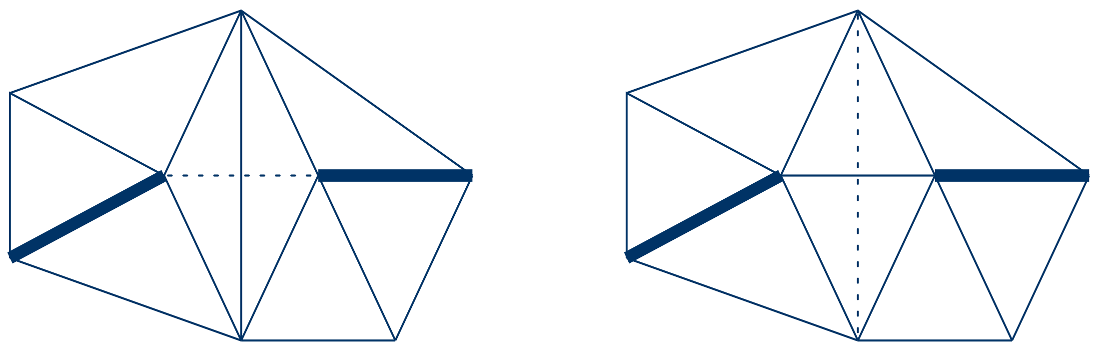


Constrained Delaunay Triangulation

- The Delaunay triangulation might not respect given input edges

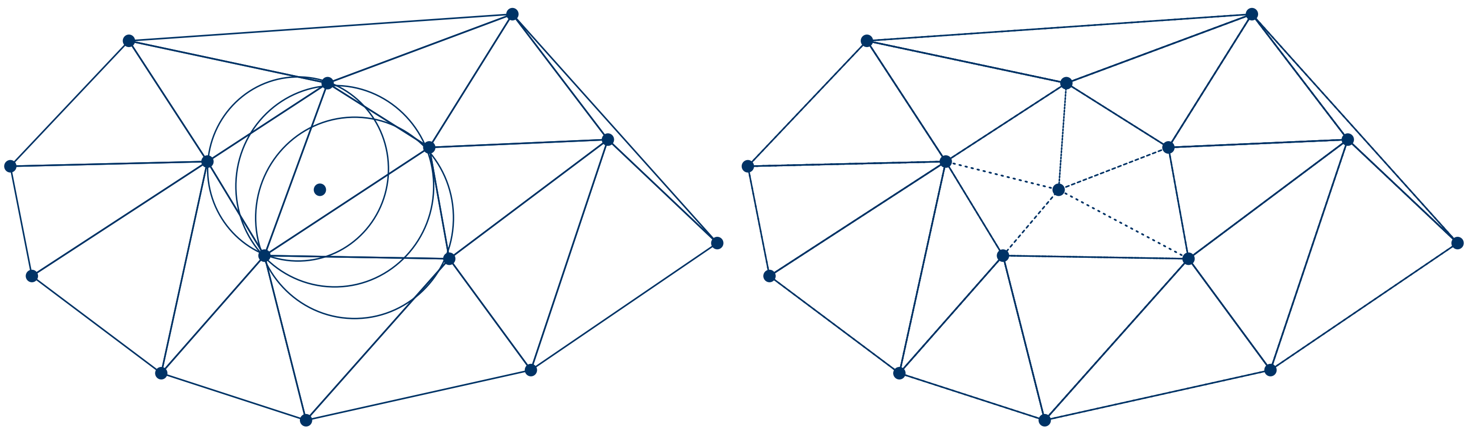


- Use local edge swaps to recover the input edges



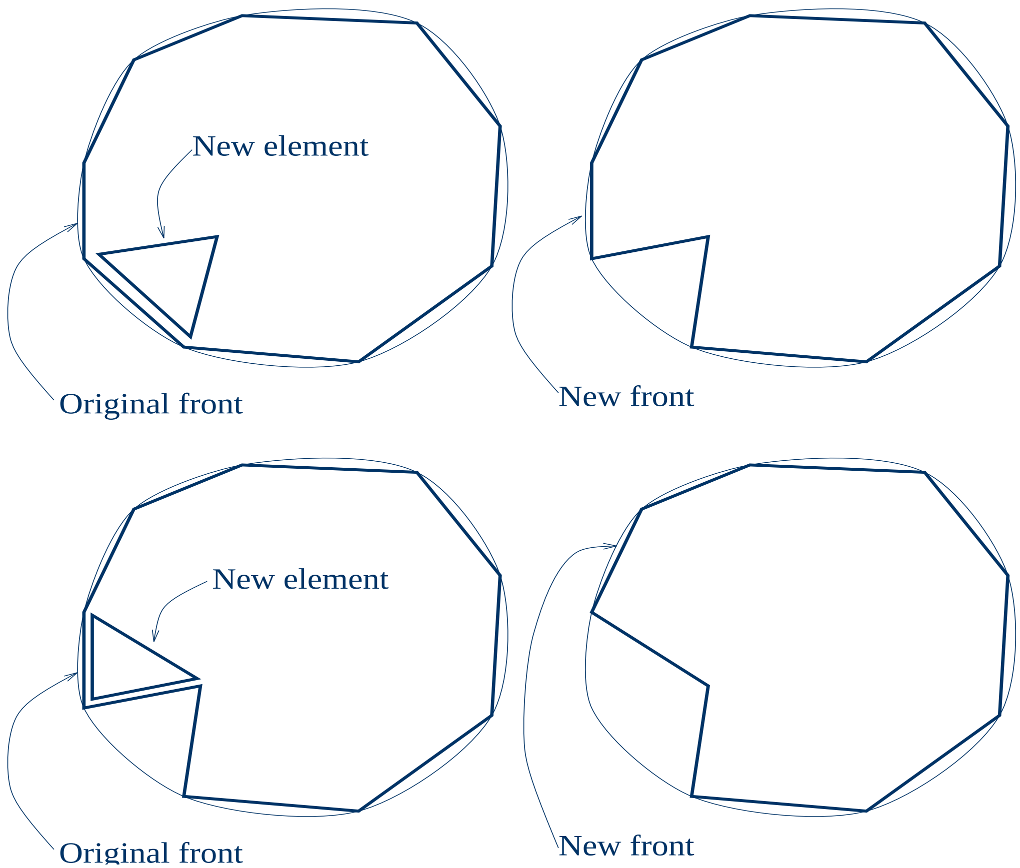
Delaunay Refinement Method

- Algorithm:
  - Form initial triangulation using boundary points and outer box
  - Replace an undesired element (bad or large) by inserting its circumcenter, retriangulate and repeat until mesh is good
- Will converge with high element qualities in 2-D
- Very fast – time almost linear in number of nodes



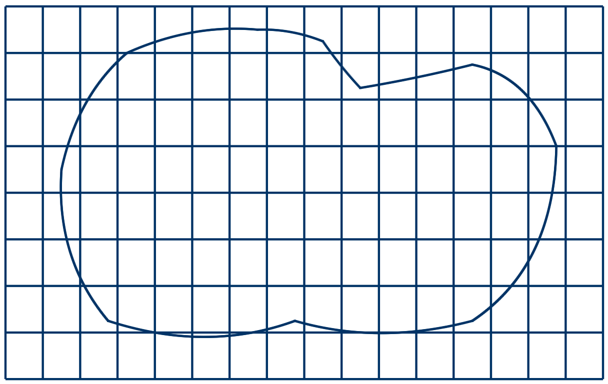
The Advancing Front Method

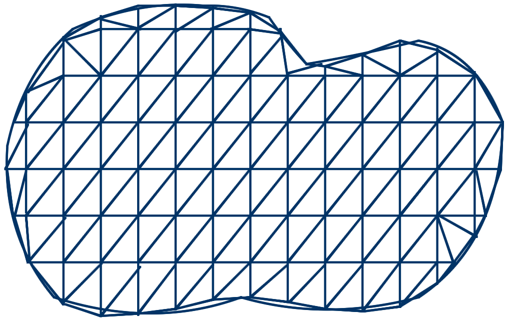
- Discretise the boundary as initial front
- Add elements into the domain and update the front
- When front is empty the mesh is complete



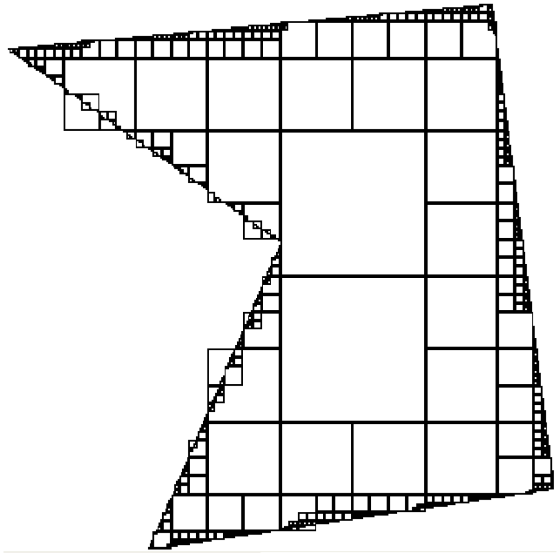
Grid Based and Octree Meshing

- Overlay domain with regular grid, crop and warp edge points to boundary





- Octree instead of regular grid gives graded mesh with fewer elements

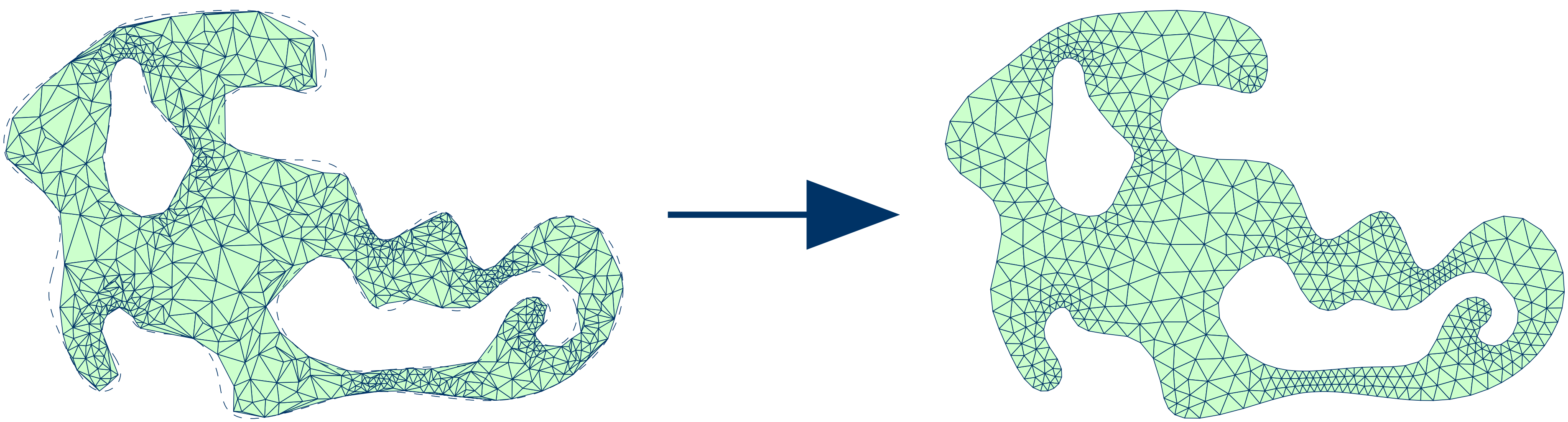


The DistMesh Mesh Generator

1. Start with *any* topologically correct initial mesh, for example random node distribution and Delaunay triangulation

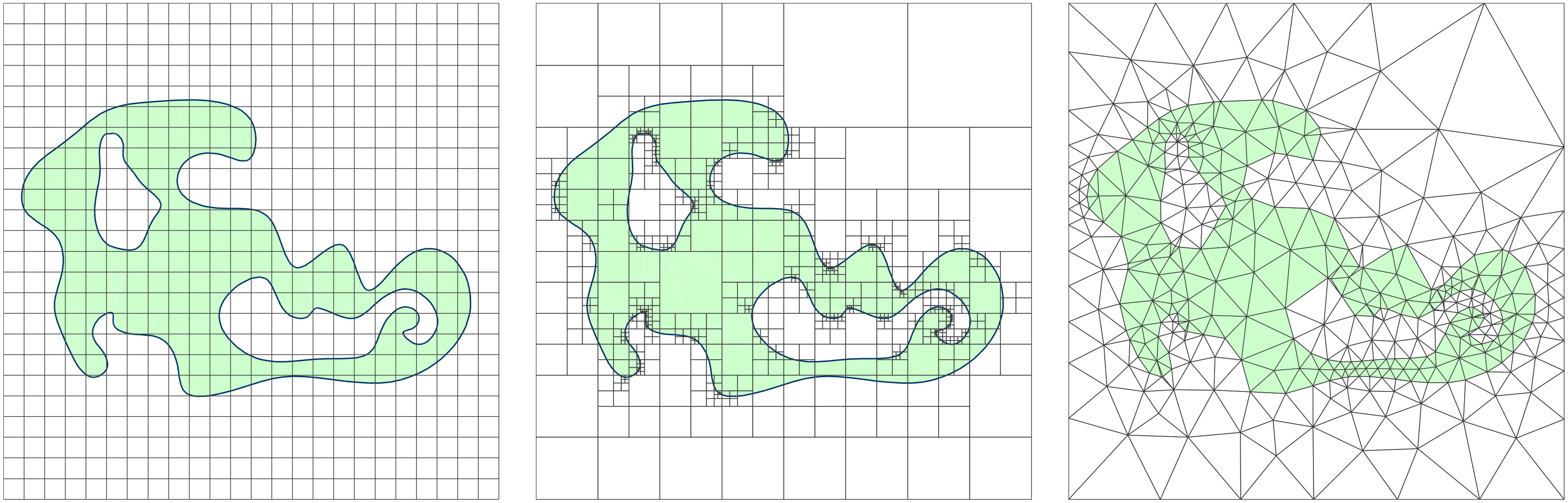
2. Move nodes to find force equilibrium in edges

- Project boundary nodes using implicit function ϕ
- Update element connectivities



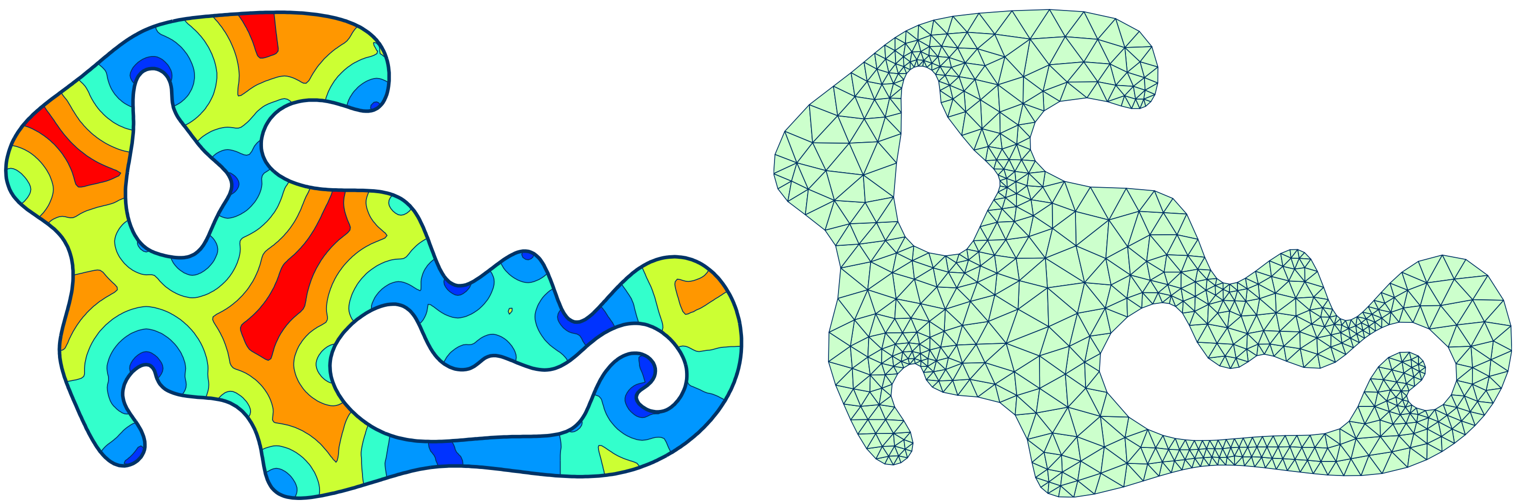
Mesh Size Functions

- Function $h(\boldsymbol{x})$ specifying desired mesh element size
- Many mesh generators need a priori mesh size functions
  - Physically-based methods such as DistMesh
  - Advancing front and Paving methods
- Discretize mesh size function $h(\boldsymbol{x})$ on a coarse background grid



Mesh Size Functions

- Based on several factors:
  - Curvature of geometry boundary
  - Local feature size of geometry
  - Numerical error estimates (adaptive solvers)
  - Any user-specified size constraints
- Also: $|\nabla h(\boldsymbol{x})| \leq g$ to limit ratio $G = g + 1$ of neighboring element sizes



Explicit Mesh Size Functions

- A *point-source*

$$h(\boldsymbol{x}) = h_{\text{pnt}} + g|\boldsymbol{x} - \boldsymbol{x}_0|$$

- Any shape, with distance function $\phi(\boldsymbol{x})$

$$h(\boldsymbol{x}) = h_{\text{shape}} + g\phi(\boldsymbol{x})$$

- Combine mesh size functions by $\min$ operator:

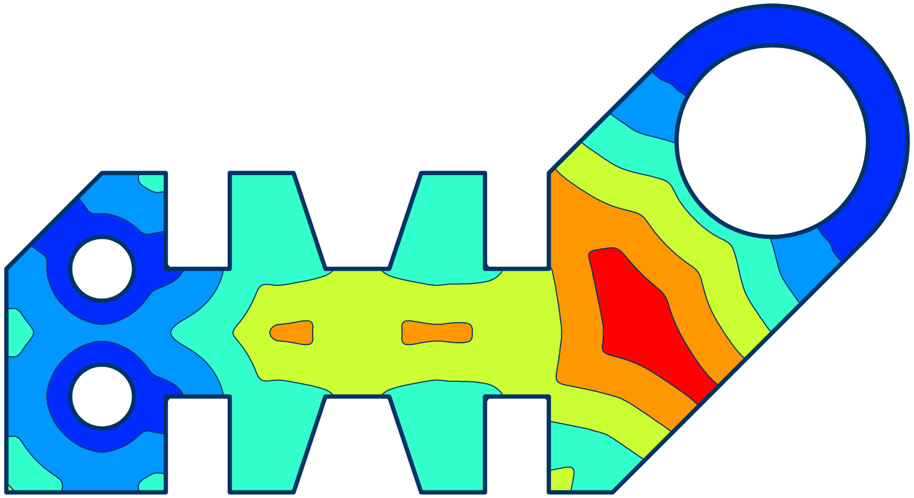
$$h(\boldsymbol{x}) = \min_i h_i(\boldsymbol{x})$$

- For more general $h(\boldsymbol{x})$, solve the *gradient limiting equation* [Persson]

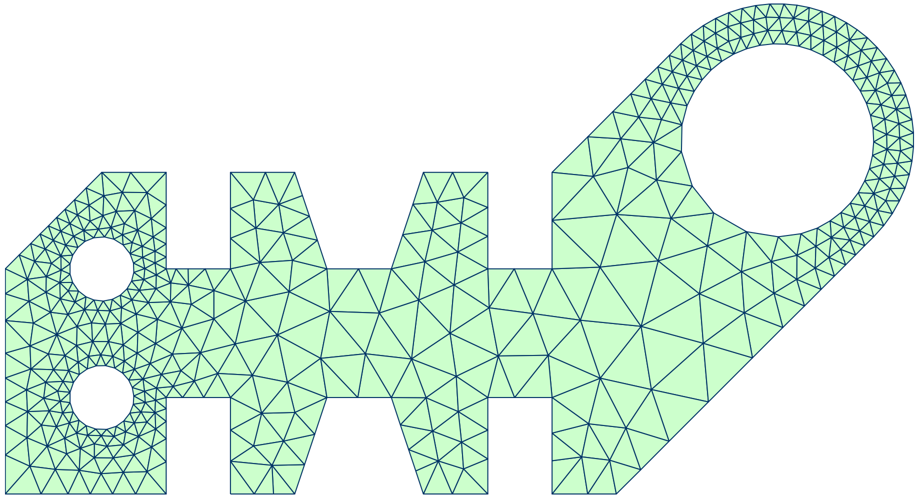
$$\frac{\partial h}{\partial t} + |\nabla h| = \min(|\nabla h|, g),$$
$$h(t = 0, \boldsymbol{x}) = h_0(\boldsymbol{x}).$$

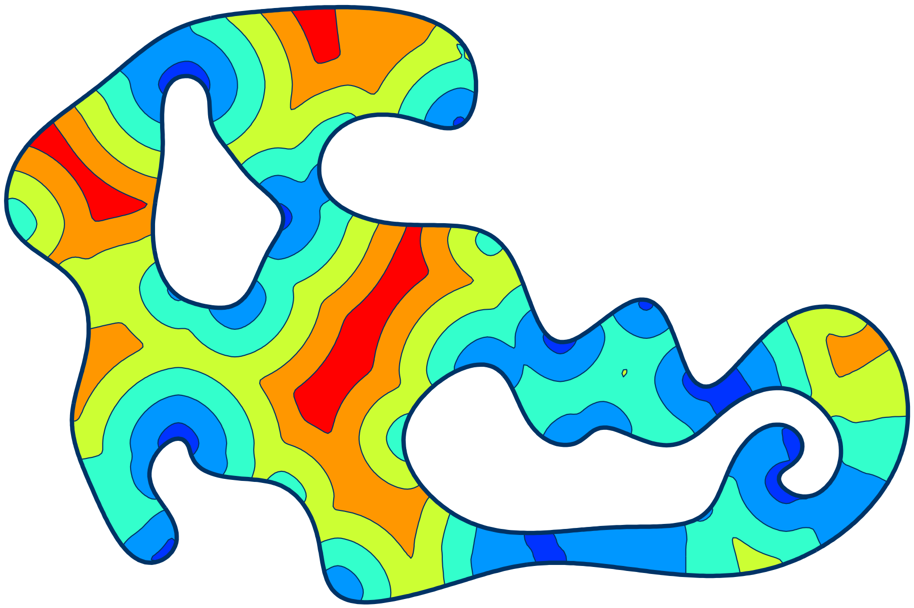
Mesh Size Functions – 2-D Examples

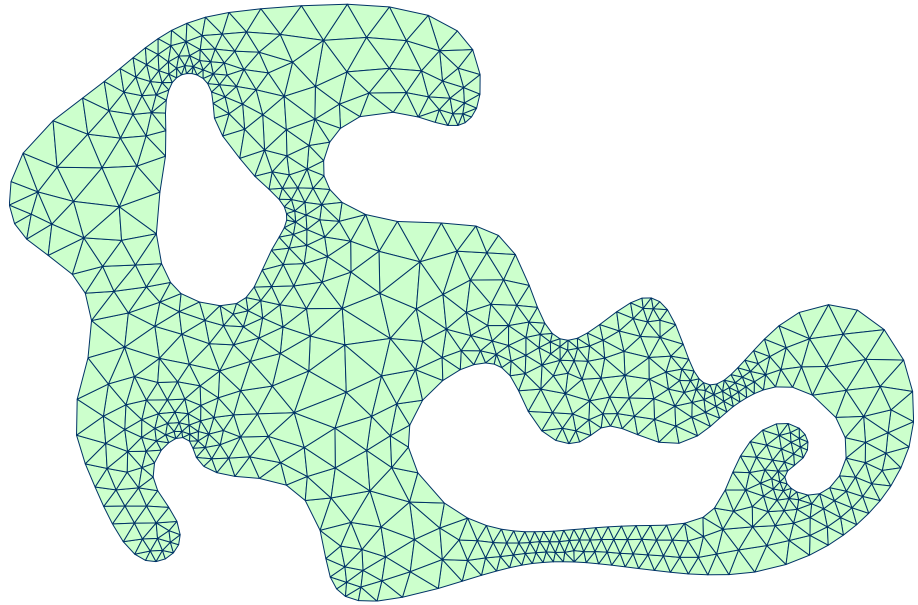
Mesh Size Function $h(x)$



Mesh Based on $h(x)$





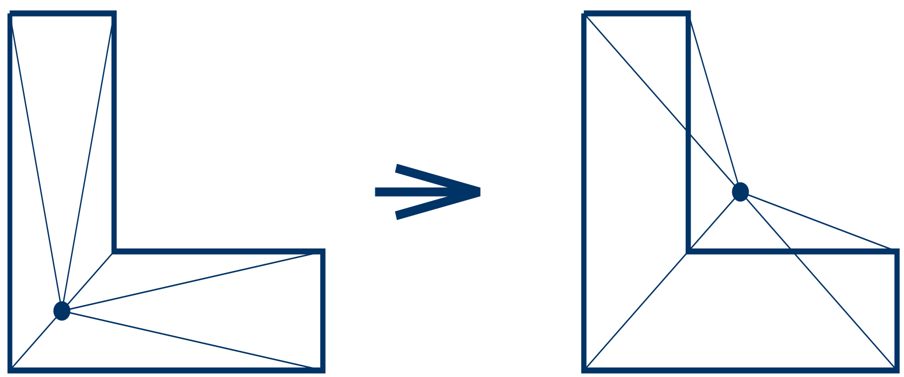


Laplacian Smoothing

- Improve node locations by iteratively moving nodes to average of neighbors:

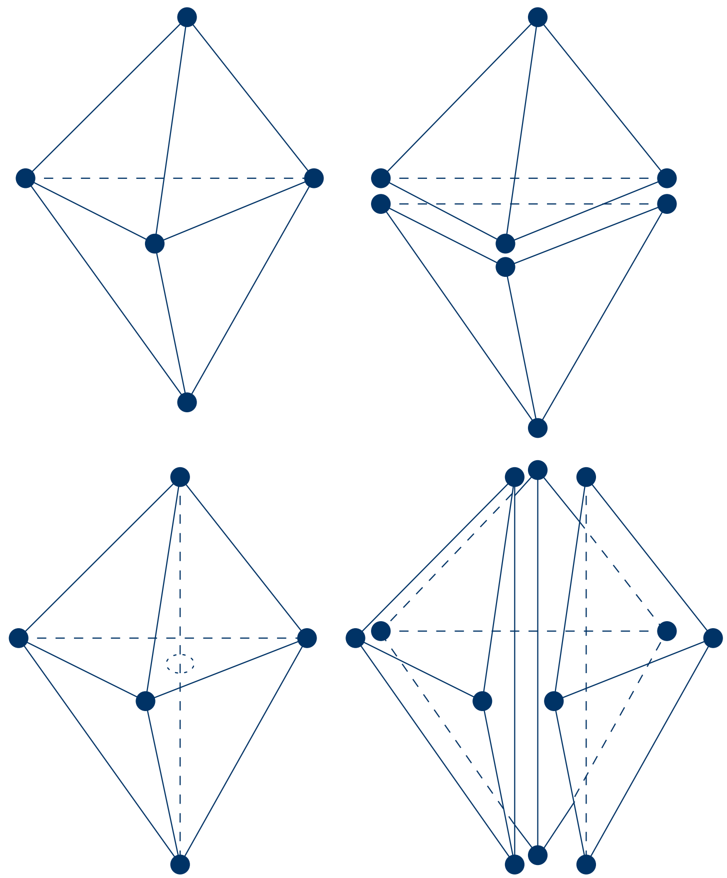
$$\boldsymbol{x}_i \leftarrow \frac{1}{n_i} \sum_{j=1}^{n_i} \boldsymbol{x}_j$$

- Usually a good postprocessing step for Delaunay refinement
- However, element quality can get worse and elements might even invert:



Face and Edge Swapping

- In 3-D there are several swappings between neighboring elements
- Face and edge swapping important postprocessing of Delaunay meshes



Boundary Layer Meshes

- Unstructured mesh for *offset curve* $\psi(\boldsymbol{x}) - \delta$
- The structured grid is easily created with the distance function