CGAL
Meshing
#include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
#include <CGAL/Delaunay_triangulation_2.h>

typedef CGAL::Exact_predicates_inexact_constructions_kernel Kernel;
typedef Kernel::Point_2 Point;

typedef CGAL::Delaunay_triangulation_2<Kernel> Delaunay;
typedef Delaunay::Vertex_handle Vertex_handle;

int main()
{
    Delaunay dt;

    dt.insert( std::istream_iterator<Point>(std::cin),
               std::istream_iterator<Point>() );

    Vertex_handle v = dt.nearest_vertex(Point(0.0,0.0));

    std::cout << "Nearest vertex to origin: " << v->point() << std::endl;
    return 0;
}
Adding Constraints
Constrained Delaunay Triangulation
Pseudo-dual: Bounded Voronoi Diagram

constrained

Bounded Voronoi diagram

“blind” triangles
Constrained Delaunay

Kilimandjaro
elevation contour
lines (38K segments)
```cpp
#include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
#include <CGAL/Constrained_Delaunay_triangulation_2.h>

typedef CGAL::Exact_predicates_inexact_constructions_kernel Kernel;
typedef Kernel::Point_2 Point;

typedef CGAL::Constrained_Delaunay_triangulation_2<Kernel> CDT;
typedef CDT::Vertex_handle Vertex_handle;

int main()
{
    CDT cdt;

    // from points
    cdt.insert_constraint(Point(0.0,0.0), Point(1.0,0.0));

    // from vertices
    Vertex_handle v1 = cdt.insert(Point(2.0,3.0));
    Vertex_handle v2 = cdt.insert(Point(4.0,5.0));
    cdt.insert_constraint(v1,v2);

    return 0;
}
```
#include <CGAL/Triangulation_conformer_2.h>

// constrained Delaunay triangulation
CDT cdt;
... // insert points & constraints

CGAL::make_conforming_Delaunay_2(cdt);
Delaunay Meshing
```cpp
#include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
#include <CGAL/Constrained_Delaunay_triangulation_2.h>
#include <CGAL/Delaunay_mesher_2.h>

typedef CGAL::Exact_predicates_inexact_constructions_kernel Kernel;
typedef CGAL::Constrained_Delaunay_triangulation_2<Kernel> CDT;

int main()
{
    CDT cdt;
    ... // insert points and constraints
    CGAL::refine_Delaunay_mesh_2(cdt);
    return 0;
}
```
Background
Delaunay Edge

An edge is said to be a Delaunay edge, if it is inscribed in an empty circle.
An edge is said to be a **Gabriel edge**, if its diametral circle is empty.
A constrained Delaunay triangulation is a **conforming Delaunay triangulation**, if every constrained edge is a Delaunay edge.

**Non conforming**

**Conforming**
A constrained Delaunay triangulation is a conforming Gabriel triangulation, if every constrained edge is a Gabriel edge.

non conforming  conforming  Gabriel
Steiner Vertices

Any constrained Delaunay triangulation can be refined into a conforming Delaunay or Gabriel triangulation by adding Steiner vertices.
Delaunay Refinement

Rule #1: break bad elements by inserting circumcenters (Voronoi vertices)

- “bad” in terms of size or shape (too big or skinny)

Picture taken from [Shewchuk]
Rule #2: Midpoint vertex insertion
A constrained segment is said to be encroached, if there is a vertex inside its diametral circle.
Delaunay Refinement

Encroached subsegments have priority over skinny triangles
API
Parameters for Mesh Generation

- **Shape**
  - Lower bound on triangle angles

*Input PLSG*

<table>
<thead>
<tr>
<th></th>
<th>5 deg</th>
<th>20.7 deg</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input PLSG" /></td>
<td><img src="image2" alt="Mesh 5 deg" /></td>
<td><img src="image3" alt="Mesh 20.7 deg" /></td>
</tr>
</tbody>
</table>
Parameters for Mesh Generation

- **Shape**
  - Lower bound on triangle angles

- **Size**
  - No constraint
  - Uniform sizing
  - Sizing function
Example Code

```c++
#include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
#include <CGAL/Constrained_Delaunay_triangulation_2.h>
#include <CGAL/Delaunay_mesher_2.h>
#include <CGAL/Delaunay_mesh_size_criteria_2.h>

typedef CGAL::Exact_predicates_inexact_constructions_kernel Kernel;
typedef CGAL::Constrained_Delaunay_triangulation_2<Kernel> CDT;
typedef CGAL::Delaunay_mesh_size_criteria_2<CDT> Criteria;
typedef CGAL::Delaunay_mesher_2<CDT, Criteria> Meshing_engine;

int main()
{
    CDT cdt;
    Meshing_engine engine(cdt);
    engine.refine_mesh();
    engine.set_criteria(Criteria(0.125, 0.5)); // min 20.6 deg
    // 0.5 for sizing
    engine.refine_mesh(); // refine once more, etc.
    return 0;
}
```
Parameters for Mesh Generation

- **Shape**
  - Lower bound on triangle angles

- **Size**
  - No constraint
  - Uniform sizing
  - Sizing function

- **Seeds**
  - Exclude/include components
Volume Mesh Generation in CGAL
Volume Mesh Generation Algorithm

repeat
{
    pick bad simplex
    if (Steiner point encroaches a facet)
        refine facet
    else
        refine simplex
        update Delaunay triangulation restricted to domain
}
until all simplices are good
Exude slivers
Apply the following rules with priority order

**Rule 1:** While there is a facet $f$ in $\text{Del}_{\text{bd}O}(\mathcal{P})$ with vertices $\notin \text{bd}O$
\hspace{1cm} \textit{refine\_facet}(f)

**Rule 2:** While there is a bad facet $f$ in $\text{Del}_{\text{bd}O}(\mathcal{P})$
\hspace{1cm} \textit{refine\_facet}(f)

**Rule 3:** While there is a bad tetrahedron $t$ in $\text{Del}_{O}(\mathcal{P})$
\hspace{1cm} \textit{refine\_tetrahedron\_or\_facet}(t)
Delaunay Filtering

- Domain boundary
- 3D complex embedded in a 3D triangulation
- Restricted Delaunay triangulation
Example
Mesh from Implicit Function

- 38k vertices
- 168k tetrahedra
- Mesh generation: 8s
Multi-Domain Volume Mesh
Polyhedral Multi-Domain
Added Value: Shortened Pipeline

Standard mesh generation pipeline

3D image

Marching cubes → Simplification → Remeshing → Mesh 1 → Mesh 2 → ... → Mesh N → Merging

Mesh Generation

CGAL-mesh generation pipeline
while (is_bad(simplex))
    refine(simplex);

Overall Design

User Data
- Domain
- Criteria

Answer

Query “oracle”
- Predicate
  - do_intersect_surface(Segment s)
  - is_in_domain(Point p)
- Construction
  - get_intersection_point(Segment s)

Mesh
Output
- Guarantees
  - Mesh matches criteria

CGAL Mesh generation Engine

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel
Overall Design

CGAL Mesh generation Engine
Overall Design

CGAL Mesh generation Engine

CGAL 3D (weighted) Delaunay triangulation
Overall Design

CGAL Mesh generation Engine

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel
CGAL Mesh generation Engine

while ( is_bad (simplex) )
    refine(simplex);

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel
Overall Design

User Data

Domain

CGAL Mesh generation Engine

while ( is_bad (simplex) )
refine(simplex);

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel
CGAL Mesh generation Engine

while ( is_bad (simplex) )
refine(simplex);

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel

Overall Design

User Data

Domain

Predicate

- do_intersect_surface(Segment s)
- is_in_domain(Point p)

Construction

- get_intersection_point(Segment s)

Query “oracle”
Overall Design

User Data

Domain

Query “oracle”

Predicate
• do_intersect_surface(Segment s)
• is_in_domain(Point p)

Construction
• get_intersection_point(Segment s)

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel

CGAL Mesh generation Engine

while ( is_bad (simplex) )
    refine(simplex);
Overall Design

User Data

Domain

Criteria

Answer

Query "oracle"

Predicate
- do_intersect_surface(Segment s)
- is_in_domain(Point p)

Construction
- get_intersection_point(Segment s)

CGAL Mesh generation Engine

while (is_bad(simplex))
refine(simplex);

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel
CGAL Mesh generation Engine

while (is_bad(simplex))
refine(simplex);

User Data

Domain
Criteria

Query “oracle”

Predicate
- do_intersect_surface(Segment s)
- is_in_domain(Point p)

Construction
- get_intersection_point(Segment s)

Answer

CGAL Mesh generation Engine

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel

Mesh

Guarantees
- Mesh matches criteria

Overall Design

User Data

Domain
Criteria

Query “oracle”

Predicate
- do_intersect_surface(Segment s)
- is_in_domain(Point p)

Construction
- get_intersection_point(Segment s)

Answer

CGAL Mesh generation Engine

CGAL 3D (weighted) Delaunay triangulation

CGAL Kernel

Mesh

Guarantees
- Mesh matches criteria
Main function

template <class C3T3, class MeshDomain_3, class MeshCriteria>

C3T3 make_mesh_3 (MeshDomain_3 domain, MeshCriteria criteria)

C3T3: 3D complex embedded in a 3D triangulation
Simple Code Example

```cpp
FT sphere_function(const Point& p) {
    return CGAL::squared_distance(p, Point(CGAL::ORIGIN)) - 1;
}

Mesh_domain domain(sphere_function,
    K::Sphere_3(CGAL::ORIGIN, 2.));

Mesh_criteria criteria(facet_angle=30, facet_size=0.1,
    facet_distance=0.025,
    cell_radius_edge=2, cell_size=0.1);

C3t3 c3t3 = CGAL::make_mesh_3<C3t3>(domain, criteria);
```