

```

1  #include <iostream>
2  #include <CGAL/Polyhedron_3.h>
3  #include <CGAL/Polyhedral_mesh_domain_with_features_3.h>
4  #include <CGAL/Linear_cell_complex_for_combinatorial_map.h>
5  #include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
6  #include <CGAL/draw_linear_cell_complex.h>
7  #include <CGAL/Linear_cell_complex_operations.h>
8
9  typedef CGAL::Exact_predicates_inexact_constructions_kernel K;
10 typedef CGAL::Linear_cell_complex_for_combinatorial_map<3> LCC_3;
11 typedef LCC_3::Dart_handle Dart_handle;
12 typedef LCC_3::Point Point;
13 typedef LCC_3::Traits Traits;
14 typedef LCC_3::FT FT;
15 typedef LCC_3::Traits::Vector Vector;
16
17 typedef CGAL::Mesh_polyhedron_3<K>::type Polyhedron;
18 typedef CGAL::Polyhedral_mesh_domain_with_features_3<K> Mesh_Domain;
19 typedef CGAL::Side_of_triangle_mesh<Polyhedron, K> Point_inside;
20
21 typedef boost::optional< boost::variant< Point, CGAL::Segment_3<K> > > IntersectionType;
22
23 const int NUMBER_OF_DARTS_PER_CUBE = 24;
24
25 LCC_3 create_initial_mesh(const Polyhedron &polyhedron, const Mesh_Domain &domain);
26
27
28 int main(int argc, char* argv[]) {
29
30     const char *fname = (argc > 1) ? argv[1] : "sphere.off";
31
32     //Create input polyhedron
33     Polyhedron polyhedron;
34     std::ifstream input(fname);
35     input >> polyhedron;
36
37     if (input.fail()) {
38         std::cerr << "Error: Cannot read file. Maybe bad path or bad file " << fname << std::endl;
39         return EXIT_FAILURE;
40     }
41     input.close();
42
43     if (!CGAL::is_triangle_mesh(polyhedron)) {
44         std::cerr << "Input geometry is not triangulated." << std::endl;
45         return EXIT_FAILURE;
46     }
47
48     //Create domain
49     Mesh_Domain domain(polyhedron);
50
51     //create initial mesh
52     LCC_3 lcc = create_initial_mesh(polyhedron, domain);
53
54     // per each lcc vertexes
55     for(LCC_3::One_dart_per_cell_range<0>::iterator it = lcc.one_dart_per_cell<0>().begin(), itend = lcc.one_dart_per_cell<0>().end(); it != itend; ++it)
56     {
57         //vertex is a cube extreme if it is incident to a number of cubes < 8
58         if( lcc.one_dart_per_incident_cell<3,0>(it).size() < 8)
59         {
60             Vector v = compute_normal_of_cell_0(lcc, it);
61
62             Point p1 = Point(v.x(), v.y(), v.z());
63             CGAL::Ray_3 ray = CGAL::Ray_3<K>(p1, v);
64
65             bool b = domain.aabb_tree().do_intersect(ray);
66             if(b)
67             {
68                 // find intersection point and set it in the hexahedron
69                 auto a = domain.aabb_tree().first_intersection(ray);
70                 Point first = boost::get<Point>(a->first);
71                 lcc.point(it) = first ;
72             }
73         }
74     }
75
76     CGAL::draw(lcc);
77     return EXIT_SUCCESS;
78 }
79
80
81
82
83
84

```