アドバンスト CG
第 4 回レポート

学籍番号：201811411
所属：情報学群情報メディア創成学類
氏名：加藤虎之介

2021 年 5 月 20 日

1　実行環境

1.1　実行に用いた OS

macOS Big Sur ver11.3.1

1.2　プログラム起動時に表示される情報

OpenGL version: 2.1 ATI-4.4.17
GLSL version: 1.20
Vendor: ATI Technologies Inc.
Renderer: AMD Radeon Pro 5300M OpenGL Engine

2　課題 A

2.1　修正したソースコード

Code 1　LoopSubdivision.cpp

```
1 #include "LoopSubdivision.h"
2 #include <unordered_map>
3
4 using namespace std;
5 using namespace glm;
6
7 void LoopSubdivision::subdivide(PolygonMesh &mesh, int nSubdiv)
8 {
9         if (mesh.getVertices().empty() || mesh.getFaceIndices().empty())
```

```
10          {
11                  std::cerr << __FUNCTION__ << ": mesh not ready" << std::endl;
12                  return;
13          }
14
15          mesh.triangulate();
16
17          HalfEdge::Mesh _mesh;
18          _mesh.build(mesh);
19
20          for (int iter = 0; iter < nSubdiv; ++iter)
21                  _mesh = apply(_mesh);
22
23          _mesh.restore(mesh);
24          mesh.calcVertexNormals();
25  }
26
27  HalfEdge::Mesh LoopSubdivision::apply(HalfEdge::Mesh &mesh)
28  {
29          //return mesh; // TODO: delete this line
30
31          HalfEdge::Mesh newMesh;
32
33          const int nOldVertices = (int)mesh.vertices.size();
34          const int nOldFaces = (int)mesh.faces.size();
35          const int nOldHalfEdges = (int)mesh.halfEdges.size();
36
37          // Step 0: allocate memory for even (i.e., old) vertices
38
39          newMesh.vertices.reserve(nOldFaces + nOldHalfEdges);
40
41          for (int vi = 0; vi < nOldVertices; ++vi)
42                  newMesh.addVertex();
43
44          // Step 1: create odd (i.e., new) vertices by splitting half edges
45
46          unordered_map<long, HalfEdge::Vertex *> newEdgeMidpointDict;
47          unordered_map<long, pair<HalfEdge::HalfEdge *, HalfEdge::HalfEdge *>>
              newHalfEdgeDict;
48
49          for (int hi = 0; hi < nOldHalfEdges; ++hi)
50          {
51                  auto oldHE = mesh.halfEdges[hi];
52
53                  vec3 startVertexPosition = mesh.vertices[oldHE->pStartVertex->id]->
                      position;
```

```
54                  vec3 endVertexPosition = mesh.vertices[oldHE->getEndVertex()->id]->
                      position;
55
56                  HalfEdge::Vertex *edgeMidpoint = nullptr;
57
58                  if (oldHE->pPair == nullptr) // on boundary
59                  {
60                          edgeMidpoint = newMesh.addVertex();
61                          // TODO: calculate the position of edge midpoint
62                          edgeMidpoint->position = (startVertexPosition + endVertexPosition
                              ) / 2.f;
63                  }
64                  else
65                  {
66                          auto edgeMidpointIter = newEdgeMidpointDict.find(oldHE->pPair->
                              id); // check if pair has been already registered
67
68                          if (edgeMidpointIter == newEdgeMidpointDict.end()) // not found
69                          {
70                                  edgeMidpoint = newMesh.addVertex();
71                                  // TODO: calculate the position of edge midpoint
72                                  vec3 topVertexPosition = mesh.vertices[oldHE->pPrev->
                                      getStartVertex()->id]->position;
73                                  vec3 bottomVertexPosition = mesh.vertices[oldHE->pPair->
                                      pNext->getEndVertex()->id]->position;
74                                  edgeMidpoint->position = 3.f / 8.f * (
                                      startVertexPosition + endVertexPosition) + 1.f / 8.f *
                                       (topVertexPosition + bottomVertexPosition);
75                          }
76                          else // founded
77                          {
78                                  // edgeMidpointInter->
                                      second で Vertex のポインタにアクセス可能
79                                  edgeMidpoint = edgeMidpointIter->second;
80                          }
81                  }
82
83                  newEdgeMidpointDict[oldHE->id] = edgeMidpoint; // used in Step 3
84
85                  auto formerHE = newMesh.addHalfEdge();
86                  auto latterHE = newMesh.addHalfEdge();
87
88                  auto evenStartVertex = newMesh.vertices[oldHE->pStartVertex->id];
89                  auto evenEndVertex = newMesh.vertices[oldHE->getEndVertex()->id];
90
91                  formerHE->pStartVertex = evenStartVertex;
```

```
92                  if (evenStartVertex->pHalfEdge == nullptr)
93                          evenStartVertex->pHalfEdge = formerHE;
94
95                  latterHE->pStartVertex = edgeMidpoint;
96                  if (edgeMidpoint->pHalfEdge == nullptr)
97                          edgeMidpoint->pHalfEdge = latterHE;
98
99                  newHalfEdgeDict[hi] = make_pair(formerHE, latterHE);
100
101                 // register pairs
102
103                 if (oldHE->pPair != nullptr)
104                 {
105                         auto iter = newHalfEdgeDict.find(oldHE->pPair->id);
106
107                         if (iter != newHalfEdgeDict.end())
108                         {
109                                 HalfEdge::HalfEdge *pairFormerHE = iter->second.first;
110                                 HalfEdge::HalfEdge *pairLatterHE = iter->second.second;
111
112                                 HalfEdge::Helper::SetPair(pairFormerHE, latterHE);
113                                 HalfEdge::Helper::SetPair(pairLatterHE, formerHE);
114                         }
115                 }
116         }
117
118         // Step 2: update even (i.e., old) vertices
119
120         for (int vi = 0; vi < nOldVertices; ++vi)
121         {
122                 auto newVertex = newMesh.vertices[vi];
123                 const auto oldVertex = mesh.vertices[vi];
124                 const auto oldVertexPosition = oldVertex->position;
125
126                 // TODO: calculate the new vertex position
127                 // c.f., HalfEdge::Vertex::countValence() in HalfEdgeDataStructure.cpp
128                 int valence = 0; // 価数
129                 auto he = oldVertex->pHalfEdge;
130                 bool onBoundary = false; // 境界線が含まれるかを格納
131
132                 std::cout << oldVertex->onBoundary() << endl;
133
134                 // 注目点の周辺の点の座標を取得する。
135                 vector<vec3> peripheral_vertices_pos; // 周囲の点の座標を格納
136                 do
137                 {
```

```
138                     ++valence;
139
140                     // pair が存在しない→境界線
141                     if (he->pPair == nullptr)
142                     {
143                             onBoundary = true;
144                             break;
145                     }
146
147                     const auto next_he = he->pPair->pNext;
148                     peripheral_vertices_pos.push_back(next_he->getEndVertex()->
                          position);
149                     he = next_he;
150             } while (he != oldVertex->pHalfEdge);
151
152             // 頂点周りに境界線がある場合
153             if (onBoundary)
154             {
155                     he = oldVertex->pHalfEdge->pPrev;
156
157                     do
158                     {
159                             ++valence;
160
161                             if (he->pPair == nullptr)
162                                     break;
163
164                             const auto prev_he = he = he->pPair->pPrev;
165                             peripheral_vertices_pos.push_back(prev_he->getStartVertex
                                  ()->position);
166                             he = prev_he;
167                     } while (he != oldVertex->pHalfEdge);
168             }
169
170             // 既存頂点の座標を更新
171             // regular の場合（Loop
                  Subdivision は三角形メッシュであるため、価数 6 は regular）
172             if (valence == 6)
173             {
174                     // 境界線や折り目の場合
175                     if (onBoundary)
176                     {
177                             newVertex->position = 3.f / 4.f * oldVertexPosition +
                                  1.f / 8.f * oldVertex->pHalfEdge->getEndVertex()->
                                  position + 1.f / 8.f * oldVertex->pHalfEdge->pPrev->
                                  pStartVertex->position;
```

```
178                             std::cout << "boundary" << endl;
179                     }
180                     // 境界線や折り目以外の場合
181                     else
182                     {
183                             // 中心点
184                             newVertex->position = 10.f / 16.f * oldVertexPosition;
185                             // 周辺点
186                             for (auto itr = peripheral_vertices_pos.cbegin(); itr !=
                                    peripheral_vertices_pos.cend(); ++itr)
187                             {
188                                     newVertex->position += 1.f / 16.f * (*itr);
189                             }
190                     }
191             }
192             // extraordinary の場合
193             else
194             {
195                     const float beta = (valence == 3) ? 3.f / 16.f : 3.f / (8.f *
                            valence);
196                     // 中心点
197                     newVertex->position = (1 - valence * beta) * oldVertexPosition;
198                     // 周辺点
199                     for (auto itr = peripheral_vertices_pos.cbegin(); itr !=
                          peripheral_vertices_pos.cend(); ++itr)
200                     {
201                             newVertex->position += beta * (*itr);
202                     }
203             }
204     }
205
206     // Step 3: create new faces
207
208     for (int fi = 0; fi < nOldFaces; ++fi)
209     {
210             auto oldFace = mesh.faces[fi];
211
212             // TODO: update the half-edge data structure within each old face
213             // HINT: the number of new faces within each old face is always 4 in the
                    case of Loop subdivision,
214             // so you can write down all the steps without using a "for" or "while"
                    loop
215
216             // 更新前の同一面内のHalfEdge を取得
217             vector<HalfEdge::HalfEdge *> pOldHalfEdges(3);
218             pOldHalfEdges[0] = oldFace->pHalfEdge;
```

```
219             pOldHalfEdges[1] = oldFace->pHalfEdge->pNext;
220             pOldHalfEdges[2] = oldFace->pHalfEdge->pPrev;
221
222             // 面内のHalfEdgeを取得
223             vector<HalfEdge::HalfEdge *> pNewHalfEdges(12);
224             // 更新前の面の辺上にあるHalfEndgeを取得
225             for (size_t i = 0; i < 3; i++)
226             {
227                     auto newPair = newHalfEdgeDict.find(pOldHalfEdges[i]->id)->
                          second;
228                     pNewHalfEdges[2 * i] = newPair.first;
229                     pNewHalfEdges[2 * i + 1] = newPair.second;
230             }
231
232             // 新しいHalfEndgeの領域を確保
233             pNewHalfEdges[6] = newMesh.addHalfEdge();
234             pNewHalfEdges[7] = newMesh.addHalfEdge();
235             pNewHalfEdges[8] = newMesh.addHalfEdge();
236             pNewHalfEdges[9] = newMesh.addHalfEdge();
237             pNewHalfEdges[10] = newMesh.addHalfEdge();
238             pNewHalfEdges[11] = newMesh.addHalfEdge();
239
240             // 新しいFaceの領域を確保
241             vector<HalfEdge::Face *> pNewFaces(4);
242             for (auto itr = pNewFaces.begin(); itr != pNewFaces.end(); ++itr)
243             {
244                     *itr = newMesh.addFace();
245             }
246
247             // HalfEdge Classのメンバ変数のセット
248             // 所属するFaceの set
249             pNewHalfEdges[0]->pFace = pNewHalfEdges[6]->pFace = pNewHalfEdges[5]->
                  pFace = pNewFaces[0];
250             pNewHalfEdges[9]->pFace = pNewHalfEdges[10]->pFace = pNewHalfEdges
                  [11]->pFace = pNewFaces[1];
251             pNewHalfEdges[1]->pFace = pNewHalfEdges[2]->pFace = pNewHalfEdges[7]->
                  pFace = pNewFaces[2];
252             pNewHalfEdges[3]->pFace = pNewHalfEdges[4]->pFace = pNewHalfEdges[8]->
                  pFace = pNewFaces[3];
253
254             // 同じ面内で自身より手前のHalfEdge
255             pNewHalfEdges[0]->pPrev = pNewHalfEdges[5];
256             pNewHalfEdges[1]->pPrev = pNewHalfEdges[7];
257             pNewHalfEdges[2]->pPrev = pNewHalfEdges[1];
258             pNewHalfEdges[3]->pPrev = pNewHalfEdges[8];
259             pNewHalfEdges[4]->pPrev = pNewHalfEdges[3];
```

```
260                 pNewHalfEdges[5]->pPrev = pNewHalfEdges[6];
261                 pNewHalfEdges[6]->pPrev = pNewHalfEdges[0];
262                 pNewHalfEdges[7]->pPrev = pNewHalfEdges[2];
263                 pNewHalfEdges[8]->pPrev = pNewHalfEdges[4];
264                 pNewHalfEdges[9]->pPrev = pNewHalfEdges[11];
265                 pNewHalfEdges[10]->pPrev = pNewHalfEdges[9];
266                 pNewHalfEdges[11]->pPrev = pNewHalfEdges[10];
267
268                 // 同じ面内で自身より先にあるHalfEdge
269                 pNewHalfEdges[0]->pNext = pNewHalfEdges[6];
270                 pNewHalfEdges[1]->pNext = pNewHalfEdges[2];
271                 pNewHalfEdges[2]->pNext = pNewHalfEdges[7];
272                 pNewHalfEdges[3]->pNext = pNewHalfEdges[4];
273                 pNewHalfEdges[4]->pNext = pNewHalfEdges[8];
274                 pNewHalfEdges[5]->pNext = pNewHalfEdges[0];
275                 pNewHalfEdges[6]->pNext = pNewHalfEdges[5];
276                 pNewHalfEdges[7]->pNext = pNewHalfEdges[1];
277                 pNewHalfEdges[8]->pNext = pNewHalfEdges[3];
278                 pNewHalfEdges[9]->pNext = pNewHalfEdges[10];
279                 pNewHalfEdges[10]->pNext = pNewHalfEdges[11];
280                 pNewHalfEdges[11]->pNext = pNewHalfEdges[9];
281
282                 // 始点となるVertex（新 newHalfEdge のみ）
283                 pNewHalfEdges[6]->pStartVertex = newEdgeMidpointDict.find(pOldHalfEdges
                      [0]->id)->second;
284                 pNewHalfEdges[7]->pStartVertex = newEdgeMidpointDict.find(pOldHalfEdges
                      [1]->id)->second;
285                 pNewHalfEdges[8]->pStartVertex = newEdgeMidpointDict.find(pOldHalfEdges
                      [2]->id)->second;
286                 pNewHalfEdges[9]->pStartVertex = newEdgeMidpointDict.find(pOldHalfEdges
                      [2]->id)->second;
287                 pNewHalfEdges[10]->pStartVertex = newEdgeMidpointDict.find(pOldHalfEdges
                      [0]->id)->second;
288                 pNewHalfEdges[11]->pStartVertex = newEdgeMidpointDict.find(pOldHalfEdges
                      [1]->id)->second;
289
290                 // 自身と向きが逆で対になるHalfEdge（新 newHalfEdge のみ）
291                 pNewHalfEdges[6]->pPair = pNewHalfEdges[9];
292                 pNewHalfEdges[7]->pPair = pNewHalfEdges[10];
293                 pNewHalfEdges[8]->pPair = pNewHalfEdges[11];
294                 pNewHalfEdges[9]->pPair = pNewHalfEdges[6];
295                 pNewHalfEdges[10]->pPair = pNewHalfEdges[7];
296                 pNewHalfEdges[11]->pPair = pNewHalfEdges[8];
297
298                 // Face Class のメンバ変数の設定
299                 pNewFaces[0]->pHalfEdge = pNewHalfEdges[0];
```

```
300                 pNewFaces[1]->pHalfEdge = pNewHalfEdges[9];
301                 pNewFaces[2]->pHalfEdge = pNewHalfEdges[1];
302                 pNewFaces[3]->pHalfEdge = pNewHalfEdges[3];
303         }
304
305         cerr << __FUNCTION__ << ": check data consistency" << endl;
306         newMesh.checkDataConsistency();
307
308         return move(newMesh);
309 }
```

Code 2 CatmullClarkSubdivision.cpp

```
1  #include "CatmullClarkSubdivision.h"
2  #include <unordered_map>
3
4  using namespace std;
5  using namespace glm;
6
7  void CatmullClarkSubdivision::subdivide(PolygonMesh &mesh, int nSubdiv)
8  {
9          if (mesh.getVertices().empty() || mesh.getFaceIndices().empty())
10         {
11                 std::cerr << __FUNCTION__ << ": mesh not ready" << std::endl;
12                 return;
13         }
14
15         HalfEdge::Mesh _mesh;
16         _mesh.build(mesh);
17
18         for (int iter = 0; iter < nSubdiv; ++iter)
19                 _mesh = apply(_mesh);
20
21         _mesh.restore(mesh);
22         mesh.calcVertexNormals();
23 }
24
25 HalfEdge::Mesh CatmullClarkSubdivision::apply(HalfEdge::Mesh &mesh)
26 {
27         // return mesh; // TODO: delete this line
28
29         HalfEdge::Mesh newMesh;
30
31         const int nOldVertices = (int)mesh.vertices.size();
32         const int nOldFaces = (int)mesh.faces.size();
33         const int nOldHalfEdges = (int)mesh.halfEdges.size();
34
```

```
35          // Step 0: allocate memory for even (i.e., old) vertices
36
37          newMesh.vertices.reserve(nOldFaces + nOldVertices + nOldHalfEdges);
38
39          for (int vi = 0; vi < nOldVertices; ++vi)
40                  newMesh.addVertex();
41
42          // Step 1: generate face centroids
43          unordered_map<long, HalfEdge::Vertex *> newCentroidDict;
44          for (int fi = 0; fi < nOldFaces; ++fi)
45          {
46                  auto oldFace = mesh.faces[fi];
47                  auto newFaceCentroid = newMesh.addVertex();
48                  // TODO: calculate the positions of face centroids
49                  newFaceCentroid->position = oldFace->calcCentroidPosition();
50                  newCentroidDict[oldFace->id] = newFaceCentroid;
51          }
52
53          // Step 2: create odd (i.e., new) vertices by splitting half edges
54
55          unordered_map<long, HalfEdge::Vertex *> newEdgeMidpointDict;
56          unordered_map<long, pair<HalfEdge::HalfEdge *, HalfEdge::HalfEdge *>>
              newHalfEdgeDict;
57
58          for (int hi = 0; hi < nOldHalfEdges; ++hi)
59          {
60                  auto oldHE = mesh.halfEdges[hi];
61
62                  vec3 startVertexPosition = oldHE->getStartVertex()->position;
63                  vec3 endVertexPosition = oldHE->getEndVertex()->position;
64
65                  HalfEdge::Vertex *edgeMidpoint = nullptr;
66
67                  if (oldHE->pPair == nullptr) // on boundary
68                  {
69                          edgeMidpoint = newMesh.addVertex();
70                          // TODO: calculate the position of edge midpoint
71                          edgeMidpoint->position = 1.f / 2.f * (startVertexPosition +
                            endVertexPosition);
72                  }
73                  else
74                  {
75                          auto edgeMidpointIter = newEdgeMidpointDict.find(oldHE->pPair->
                            id); // check if pair has been already registered
76
77                          if (edgeMidpointIter == newEdgeMidpointDict.end())
```

```
78                      {
79                              edgeMidpoint = newMesh.addVertex();
80                              // TODO: calculate the position of edge midpoint
81                              edgeMidpoint->position = 1.f / 4.f * (
                                  startVertexPosition + endVertexPosition + oldHE->pFace
                                  ->calcCentroidPosition() + oldHE->pPair->pFace->
                                  calcCentroidPosition());
82                      }
83                      else
84                      {
85                              edgeMidpoint = edgeMidpointIter->second;
86                      }
87              }
88
89              newEdgeMidpointDict[oldHE->id] = edgeMidpoint;
90
91              auto formerHE = newMesh.addHalfEdge();
92              auto latterHE = newMesh.addHalfEdge();
93
94              auto evenStartVertex = newMesh.vertices[oldHE->pStartVertex->id];
95              auto evenEndVertex = newMesh.vertices[oldHE->pNext->pStartVertex->id];
96
97              formerHE->pStartVertex = evenStartVertex;
98              if (evenStartVertex->pHalfEdge == nullptr)
99                      evenStartVertex->pHalfEdge = formerHE;
100
101             latterHE->pStartVertex = edgeMidpoint;
102             if (edgeMidpoint->pHalfEdge == nullptr)
103                     edgeMidpoint->pHalfEdge = latterHE;
104
105             newHalfEdgeDict[hi] = make_pair(formerHE, latterHE);
106
107             // register pairs
108
109             if (oldHE->pPair != nullptr)
110             {
111                     auto iter = newHalfEdgeDict.find(oldHE->pPair->id);
112
113                     if (iter != newHalfEdgeDict.end())
114                     {
115                             HalfEdge::HalfEdge *pairFormerHE = iter->second.first;
116                             HalfEdge::HalfEdge *pairLatterHE = iter->second.second;
117
118                             HalfEdge::Helper::SetPair(pairFormerHE, latterHE);
119                             HalfEdge::Helper::SetPair(pairLatterHE, formerHE);
120                     }
```

```
121                 }
122         }
123
124         // Step 3: update even (i.e., old) vertex positions
125
126         for (int vi = 0; vi < nOldVertices; ++vi)
127         {
128                 auto newVertex = newMesh.vertices[vi];
129                 const auto oldVertex = mesh.vertices[vi];
130                 const auto oldVertexPosition = oldVertex->position;
131
132                 // TODO: calculate the new vertex position
133                 // c.f., HalfEdge::Vertex::countValence() in HalfEdgeDataStructure.cpp
134
135                 int valence = 0;
136                 bool onBoundary = false;
137                 auto he = oldVertex->pHalfEdge;
138
139                 vector<HalfEdge::Vertex *> midpoints; // 中点を格納
140                 vector<HalfEdge::Vertex *> centroids; // 重心を格納
141
142                 do
143                 {
144                         ++valence;
145
146                         // 中点を取得
147                         auto midpoint = newEdgeMidpointDict.find(he->id)->second;
148                         midpoints.push_back(midpoint);
149                         // 重心を取得
150                         auto centroid = newCentroidDict.find(he->pFace->id)->second;
151                         centroids.push_back(centroid);
152
153                         if (he->pPair == nullptr)
154                         {
155                                 onBoundary = true;
156                                 break;
157                         }
158
159                         he = he->pPair->pNext;
160                 } while (he != oldVertex->pHalfEdge);
161
162                 if (onBoundary)
163                 {
164                         he = oldVertex->pHalfEdge->pPrev;
165
166                         do
```

```
167                         {
168                                 ++valence;
169
170                                 auto midpoint = newEdgeMidpointDict.find(he->id)->second
                                      ;
171                                 midpoints.push_back(midpoint);
172
173                                 auto centroid = newCentroidDict.find(he->pFace->id)->
                                      second;
174                                 centroids.push_back(centroid);
175
176                                 if (he->pPair == nullptr)
177                                 {
178                                         break;
179                                 }
180
181                                 he = he->pPair->pPrev;
182                         } while (he != oldVertex->pHalfEdge);
183                 }
184
185                 // even vertex の座標を計算
186                 // 境界線上にある場合
187                 if (onBoundary)
188                 {
189                         newVertex->position = 3.f / 4.f * oldVertexPosition + 1.f / 8.f
                             * (oldVertex->pHalfEdge->getEndVertex()->position + oldVertex
                              ->pHalfEdge->pPrev->pStartVertex->position);
190                 }
191                 // 境界線以外にある場合
192                 else
193                 {
194                         vec3 R, S; // R:辺の中点の平均座標 ,S:面の重心の平均座標
195                         for (size_t i = 0; i < valence; i++)
196                         {
197                                 R += midpoints[i]->position;
198                                 S += centroids[i]->position;
199                         }
200                         R /= (float)valence;
201                         S /= (float)valence;
202
203                         newVertex->position = (valence - 3.f) / valence *
                              oldVertexPosition + 4.f / valence * R - 1.f / valence * S;
204                 }
205         }
206
207         // Step 4: set up new faces
```

```
208
209         for (int fi = 0; fi < nOldFaces; ++fi)
210         {
211                 auto oldFace = mesh.faces[fi];
212                 auto centroidVertex = newMesh.vertices[oldFace->id + nOldVertices];
213
214                 // TODO: update the half-edge data structure within each old face
215                 // HINT: use the following std::vector to store temporal data and process
                      step by step
216                 vector<HalfEdge::HalfEdge *> tmpToCentroidHalfEdges;
217                 vector<HalfEdge::Face *> tmpNewFaces;
218
219                 // face の設定
220                 auto he = oldFace->pHalfEdge;
221                 int k = 0;
222                 do
223                 {
224                         auto newFace = newMesh.addFace();
225
226                         tmpNewFaces.push_back(newFace);
227
228                         he = he->pNext;
229                         k++;
230                 } while (he != oldFace->pHalfEdge);
231
232                 // HalfEdge の設定(同一面内)(ここが問題)
233                 he = oldFace->pHalfEdge;
234                 int i = 0;
235                 do
236                 {
237                         // 中点→重心に向かうhalf edge の追加
238                         auto toCentroidHalfEdge = newMesh.addHalfEdge();
239                         // 重心→中点に向かうhalf edge の追加
240                         auto toMidpointHalfEdge = newMesh.addHalfEdge();
241
242                         // 始点となるvertex を設定
243                         // 中心→重心
244                         // auto midpoint = newEdgeMidpointDict.find(he->pPrev->id)->second
                              ;
245                         auto midpoint = newEdgeMidpointDict.find(he->id)->second;
246                         toCentroidHalfEdge->pStartVertex = midpoint;
247                         // 重心→中心
248                         toMidpointHalfEdge->pStartVertex = centroidVertex;
249
250                         // 自身が所属するFace
251                         auto pNewFace = tmpNewFaces[i];
```

```
252                     toCentroidHalfEdge->pFace = pNewFace;
253                     toMidpointHalfEdge->pFace = pNewFace;
254
255                     auto he1 = newHalfEdgeDict.find(he->id)->second.first;
256                     auto he2 = newHalfEdgeDict.find(he->pPrev->id)->second.second;
257                     he1->pFace = pNewFace;
258                     he2->pFace = pNewFace;
259
260                     // 中点・重心の頂点に関してその頂点を始点とするいずれかのHEを登録
261                     midpoint->pHalfEdge = toCentroidHalfEdge;
262                     centroidVertex->pHalfEdge = toMidpointHalfEdge;
263
264                     // pNext
265                     toCentroidHalfEdge->pNext = toMidpointHalfEdge;
266                     toMidpointHalfEdge->pNext = he2;
267                     he2->pNext = he1;
268                     he1->pNext = toCentroidHalfEdge;
269
270                     // pPrev
271                     toCentroidHalfEdge->pPrev = he1;
272                     toMidpointHalfEdge->pPrev = toCentroidHalfEdge;
273                     he2->pPrev = toMidpointHalfEdge;
274                     he1->pPrev = he2;
275
276                     // 一時保存配列への追加
277                     tmpToCentroidHalfEdges.push_back(toCentroidHalfEdge);
278
279                     // faceに含まれるいずれかのHalfEdgeの一本を登録
280                     pNewFace->pHalfEdge = toMidpointHalfEdge;
281
282                     he = he->pNext;
283                     i++;
284             } while (he != oldFace->pHalfEdge);
285
286             // pairの設定
287             auto olfHE = oldFace->pHalfEdge;
288             for (int i = 0; i < k; i++)
289             {
290                     auto toMidpointHE = tmpNewFaces[i]->pHalfEdge;
291
292                     int j = (i == 0) ? (k - 1) : (i - 1);
293                     auto toCentroidHE = tmpToCentroidHalfEdges[j];
294
295                     toMidpointHE->pPair = toCentroidHE;
296                     toCentroidHE->pPair = toMidpointHE;
297             }
```

```
298         }
299
300         std::cerr << __FUNCTION__ << ": check data consistency" << endl;
301         newMesh.checkDataConsistency();
302
303         return move(newMesh);
304 }
```

2.2 実行結果

2.2.1 Loop 細分割

Loop 細分割を cube.obj に対して実行した結果は以下のようになった。

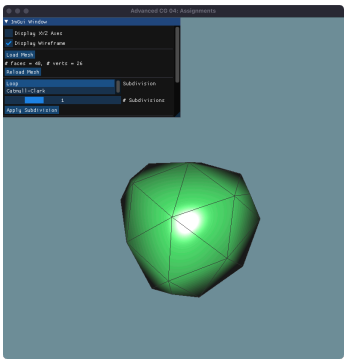

図 1　適用回数：1 回

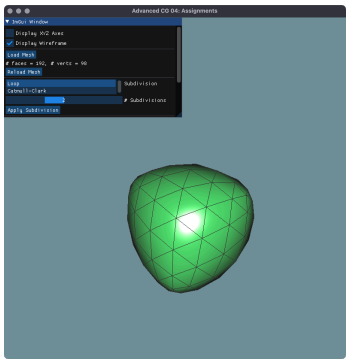

図 2　適用回数：2 回

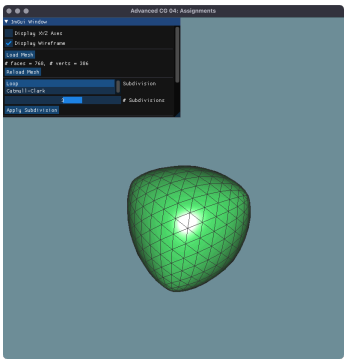

図 3　適用回数：3 回

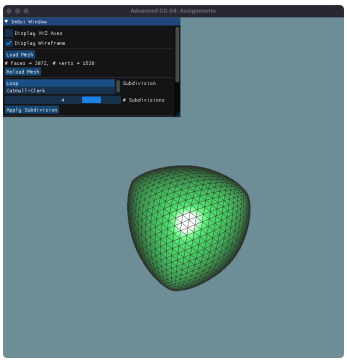

図 4　適用回数：4 回

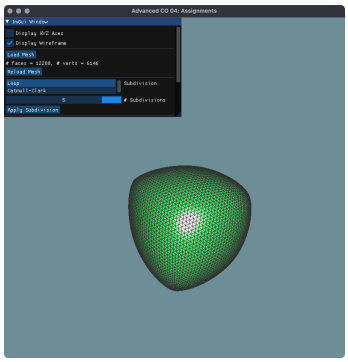

図 5　適用回数：5 回

2.2.2 Catmull-Clark 細分割

Catmull-Clark 細分割を cube.obj に対して実行した結果は以下のようになった。

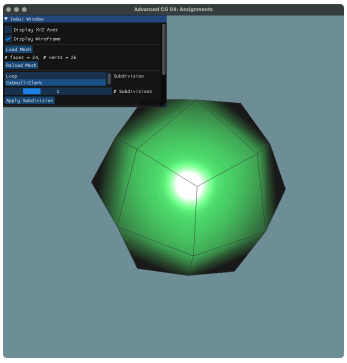

図 6　適用回数：1 回

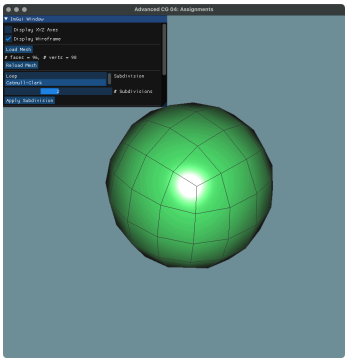

図 7　適用回数：2 回

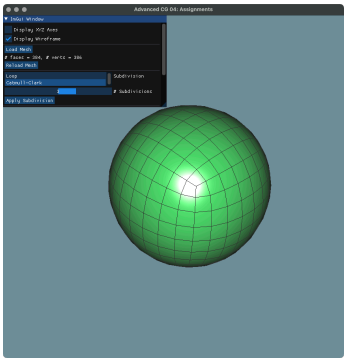

図 8　適用回数：3 回

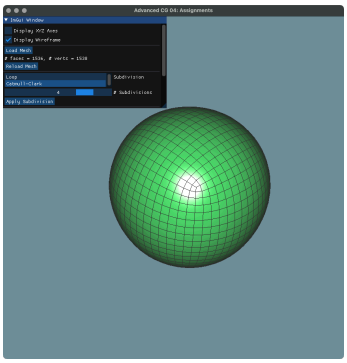

図 9　適用回数：4 回

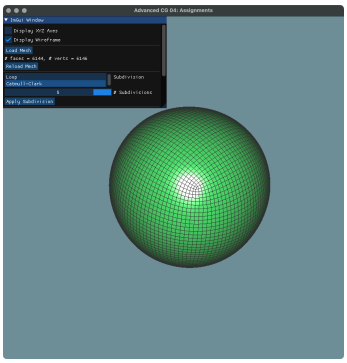

図 10　適用回数：5 回