COSC422 Advanced Computer Graphics
Programming Exercise 12
Mesh Subdivision (Charles-Loop Algorithm)

This programming exercise demonstrates the working of the Charles-Loop algorithm in iteratively subdividing a mesh to produce a smooth approximation of a coarse mesh.

MeshSubdivin.cpp:

The program `MeshSubdivn.cpp` loads and displays the mesh model "Hand.off". The low polygonal model contains 58 vertices and 112 triangles (Fig. 1). The model can be rotated about the *x* and *y* axes using the arrow keys.

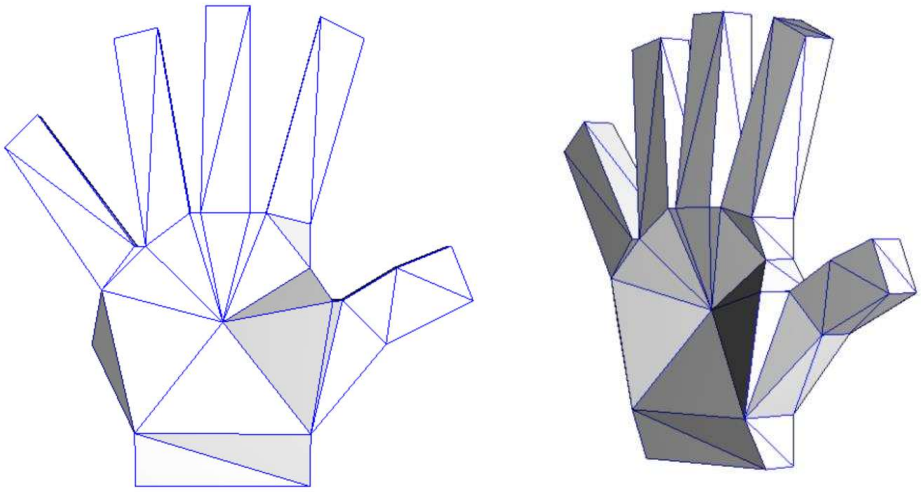


Fig. 1.

We will use the Charles-Loop subdivision method (Slides [7]:64-66) to create a smoother, but higher resolution version of the mesh model.

Include the following statement at the beginning of the program:

```
#include <OpenMesh/Tools/Subdivider/Uniform/LoopT.hh>
```

Include the following statement *after* the `typedef` declaration at the beginning of the program:

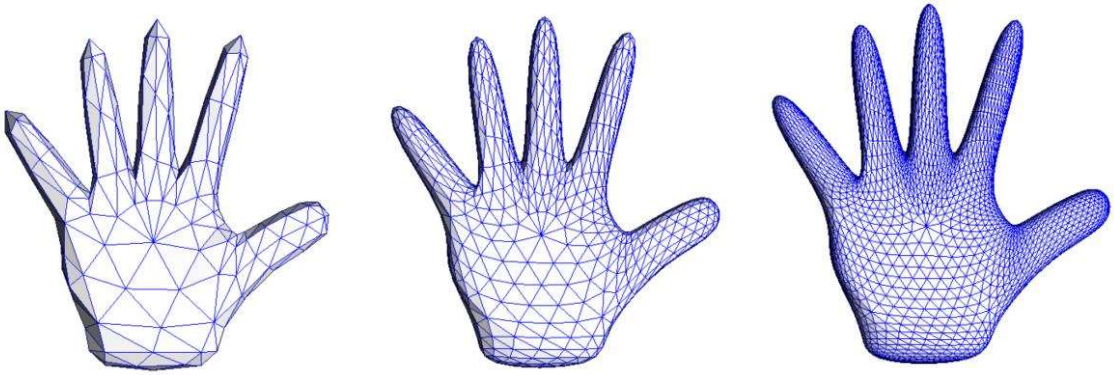
```
OpenMesh::Subdivider::Uniform::LoopT<MyMesh> loop;
```

Include the following statements in the `intialize()` function (before `glClearColor()` function call):

```
int niter = 1;                    //Number of iterations
loop.attach(mesh);
loop(niter);
loop.detach();

mesh.update_face_normals();
```

The program displays the subdivided mesh after one iteration of the Charles-Loop algorithm (Fig. 2(a)). This model has 226 vertices and 448 (112*4) faces. Incrementing the value of the variable `niter` produces higher resolution meshes as shown in Figs 2(b),(c).



(a)
`niter` = 1
Verts: 226
Faces: 448

(b)
`niter` = 2
Verts: 898
Faces: 1792

(c)
`niter` = 3
Verts: 3586
Faces: 7168

Fig. 2

[7]: COSC422 Lecture Slides "7 – Mesh Processing"