

# Project 3: Subdivision

## 1 What to do?

The goal is to implement the Loop and Butterfly subdivision schemes for triangular meshes. The input mesh will be given in the same format as in Project 2. The first two entries will be integers, the triangle count and the vertex count. Then, there will follow a triangle table: a number of lines, each listing three integers (ID's of vertices bounding a triangle). Finally, there will be the vertex table: coordinates of all the vertices. For example, here is an input file that represents a tetrahedron:

```
4 4
0 1 2
2 1 3
2 3 0
0 3 1
0.000000 0.000000 1.000000
1.000000 0.000000 0.000000
0.000000 1.000000 0.000000
0.000000 0.000000 0.000000
```

You can assume that the triangles are consistently oriented and the input mesh is manifold.

Your program *must* compile on the CoC Linux machines and run as follows:  
`% project3 -d window_size < input_file_name`

You should send a tar file containing all the files (they should extract to '.' and compile with 'make'). Projects which do not behave as described above (and below) will not be graded.

We are providing a skeleton code (very similar to that in project 2). Apart from 'faster', 'slower' and 'stop/run' menu items, your programs should have 'loop' and 'butterfly' items that would apply one step of the two subdivision schemes to the currently displayed mesh. In particular, this means that if you select 'Loop' or 'butterfly'  $N$  times, you should see the input mesh after  $N$  subdivision steps. Use **flat shading** to shade your mesh (this will allow one to see the individual triangles). As in the last project, use display lists but remember to free them when they are not needed. Of course, your code should display things nicely... Feel free to reuse the code from your previous project. If you can't or don't want to, just scale the model after you read it so that its bounding box just barely fits into the cube that the sample code draws and the center of the cube overlaps with the center of the bounding box.

## 2 Grading

Half of the grade for the Loop scheme and half for the Butterfly scheme. If selecting 'Loop' or 'Butterfly' from the menu causes the program to crash or show nothing, you will get no credit for the corresponding part. Here are things we are going to look at:

1. Execution speed (there should not be a lag of more than half a minute between the time one selects one of the subdivision steps until the subdivided model shows up if the input to your subdivision routine has no more than 100,000 vertices; of course, it will be much smaller for small models): 15%, but automatically 0 if subdivision cannot be iterated (see 2.)
2. Ability to iterate subdivision (also mixing the Loop and butterfly schemes should be possible by selecting 'Loop' and 'butterfly' in an alternating fashion): 10%
3. One subdivision step works and looks OK: 25%

## 3 Deadlines, submission...

The deadline for this project is April 4th midnight. The usual penalty for late turnins applies ( $3^{n-1}$ ,  $n$  =number of days late).

## 4 Extra credit

Extra credit offer will apply to this project! Another menu item to implement for extra 20 project points will appear in about a week.