AM-MIRI: Lab 4 Exercises

Here is a list of exercises on iso-surface extraction from scalar fields, using OpenMesh to model the surfaces. In `/assig/sgi/Sessions/lab4.2/DiscreteModels2018.pdf` you have a copy of the slides used in class. The directory `/assig/sgi/Sessions/lab4.2/Data` holds some small-size scalar fields in the format documented in the slides, that you can use to test your implementations. Models are in files `X.txt`, and the expected result is available in files `X@num.jpg`, where `num` is the iso-value used. There is also a short animation in `Blooby0100.avi`.

Finally, folder `/assig/sgi/Sessions/lab4.2/taulaMC` contains the implementation of the Marching Cubes look-up table discussed in class.

Exercise 1 — Compile QtViewer

Using the version of OpenMesh in `/assig/sgi/OM-6.3` (or in `/assig/sgi/OpenMesh-6.3.tar.bz2`), or —at your choice— a later version, compile and link the QtViewer program. You may use the version included in the library sources or the slightly-modified version in `/assig/sgi/QtViewer`, which can be built by itself using `qmake; make -j`.

Exercise 2 — Read simple volume models

Add code to the QtViewer so that there is a new menu entry for loading scalar fields stored in text files consisting of $N^3 + 1$ lines, each line containing a number, so that the first line contains N (the dimension of the voxelization, which is the same in all three dimensions), and the following lines contain the values of the scalar field sorted so that the value for voxel (i, j, k) is at line $i \cdot N^2 + j \cdot N + k + 1$ (lines numbered starting at zero, which is the line containing N). Add also a mechanism for choosing a threshold value. (It helps if while loading the scalar field you track the minimum and maximum values, and allow thresholds only in that interval). All models are small, so there should be no memory or size issues.

Exercise 3 — Minimal visualization

In order to see the data and the effect of the threshold, construct a mesh with a small octagon around each node with a value smaller than the threshold. Arrange that this mesh be rendered by QtViewer.

You should be able to change the threshold and dynamically see the set of points below the threshold change (i.e. for each new threshold, the mesh is remodeled accordingly)

Exercise 4 — Marching Cubes

Using the look-up table provided (in `/assig/sgi...`), add code to compute the Marching Cubes iso-surface of the data loaded. As in Ex. 3, arrange that when the user changes the threshold, the mesh is recomputed.

Exercise 5 — Threshold animation

Add a button that will trigger a loop animation between a minimum and a maximum value of the threshold, by incrementing slightly the threshold and re-rendering in a loop.