A2: Meshedit

MeshEdit is the first of three components of Scotty3D. The primary repository is located at **https://github.com/CMU-Graphics/Scotty3D**. The **documentation website** linked from the repository will be the primary source of information about this assignment, this document only contains administrative details about grading and submission.

Due date for Part 1 Local Operations: March 17th, 11:59pm

Due date for Part 2 Global Operations: March 24th, 11:59pm

Evaluation

For this assignment, you will implement methods in `student/meshedit.cpp`.

The User Guide on the website describes a large number of features that are in principle available in MeshEdit mode. **You do not have to implement all of these features to receive full credit on the assignment!** However, you do have to successfully implement a *subset* of the features. Implementing additional features beyond the required subset will earn you extra credit points.

The particular requirements, and the percentage of the grade they correspond to, are:

- **Four** local operations: `EdgeCollapse`, `EdgeFlip`, `EdgeSplit` and `FaceBevel` (10 pts each)
- `Triangulation`, `LinearSubdivision`, and `CatmullClarkSubdivision` (10 pts each)
- **One** of: `LoopSubdivision`, `IsotropicRemeshing`, or `Simplification` (30 pts each)
- Create **one** beautiful 3D model using Scotty3D (20 pts)
- Submit a suggested improvement to the A2 documentation on Piazza (5 pts)
- Submit a writeup (5 pts)
- (Total score percentage is number of points out of 130)

In other words, *everyone* has to implement EdgeCollapse, EdgeFlip, EdgeSplit, FaceBevel, triangulation, linear subdivision, and Catmull-Clark. These features are the bare minimum needed to model interesting subdivision surfaces; triangulation is necessary in order to do the global remeshing task(s). *Note that some of the global tasks require that you implement specific local operations!* For instance, the implementation of Simplification depends on EdgeCollapse. The local operations are as follows (these operations are described in the User Guide):

- `VertexBevel`
- `EdgeBevel`
- `FaceBevel` - everyone must implement
- `EraseVertex`
- `EraseEdge`
- `EdgeCollapse` - everyone must implement
- `FaceCollapse`
- `EdgeFlip` - everyone must implement
- `EdgeSplit` - everyone must implement

The global operations, and their dependency on local operations, are as follows:

- `Triangulation` - everyone must implement
- `LinearSubdivision` - everyone must implement
- `CatmullClarkSubdivision` - everyone must implement
- `LoopSubdivision` - depends on `EdgeSplit` and `EdgeFlip`
- `IsotropicRemeshing` - depends on `EdgeSplit`, `EdgeFlip`, and `EdgeCollapse`
- `Simplification` - depends on `EdgeCollapse`

You are free to change the subset of features you choose to implement at any point during the assignment, but you should **clearly indicate which features you chose** (including those implemented for extra credit) by putting this information in your writeup (described below). You might find it worthwhile using the `validate` member function of the `Halfedge_Mesh` class to help debug your operations.

Note: Do NOT remove the comments that say "Task 1", "Task 2", etc., which we will use for grading.

Extra credit: each additional local operation beyond the requirements will be worth 2% of the assignment grade; each additional global operation will be worth 4% of the assignment grade. The maximum possible grade on the assignment is 110%.

America's Next Top 3D Model

Every student is required to submit a 3D model created using their implementation of Scotty3D, which will be automatically entered into a class-wide 3D modeling competition. Models will be critiqued and evaluated based on both technical sophistication and aesthetic beauty. **Note:** Use of any other 3D package (e.g., free or commercial 3D modelers like *Maya* or *Blender*) is prohibited! This model must be created from scratch by applying the operations implemented as part of the assignment to a primitive shape and saving the result.

NOTE: We *are* expecting some high-quality output here---or at least some creativity! Don't just brush this one off. :-)

Include this model in the root directory of your submission as `model.dae`.

Documentation

Clear, well-written documentation is a critical part of software development. Since you (the students) are the ones who will most benefit from good documentation---or will suffer through bad documentation---we are "crowd sourcing" improvements to the course material. **What did you find confusing**---*and then finally figure out?*

For 5 points of your assignment grade, you will need to submit suggested edits to the assignment documentation. These could be:

- Suggested changes or additions to the assignment writeup
- Suggested changes or additions to assignment website (for A2, A3, and A4)
- Suggested changes or additions to comments in the skeleton code

Do not wait until the due date to submit these suggested edits. They will be most helpful to your classmates (and to us!) if they are submitted ahead of time, so that we can *immediately* incorporate any good suggestions into the actual course material. To submit your edits, you must therefore:

- Go on Piazza
- Find the thread with the appropriate documentation thread label.
- Add a comment with your suggested edit. You may post anonymously to classmates, but not instructors!

The TAs will monitor this label, and immediately update the docs with any/all useful suggestions. This Piazza post will also be recorded as part of your assignment grade. Note that we do **not** have to accept your edit in order for you to receive full credit on the assignment. However, we will grade your edits based on whether they appear to be a "good faith effort" to make a useful comment. In other words, did you really think about what is clear/unclear and provide a useful edit? Or did you just write something totally random at the very last minute? :-) Especially useful edits (e.g., those that provide nice insights, or point out serious bugs/errors) may receive extra credit.

Writeup

Additionally, you will submit a short document explaining what you have implemented, and any particular details of your submission. If your submission includes any implementations which are not entirely functional, please detail what works and what doesn't, along with where you got stuck. This document does not need to be long; correctly implemented features may simply be listed, and incomplete features should be described in a few sentences at most.

The writeup must be a pdf, markdown, or plaintext file. Include it in the root directory of your submission as `writeup.pdf`, `writeup.md`, or `writeup.txt`. If you did extra credit, you **must** note it under a heading "Extra Credit," otherwise it will not be counted.

Failure to submit this writeup on either Checkpoint or Final will incur a 5 pt penalty on the assignment it is forgotten on.

Code Environment

This codebase should compile on Linux, macOS, and Windows on a typical environment. The build instructions given on the project page will walk you through installing dependencies and building the code.

Handin Instructions

As with the previous assignments, we will be submitting on Autolab. You should create a tar archive of your entire `src` subdirectory along with the writeup (e.g. `writeup.txt`) and 3D modeling submission (`model.dae`).

```
$ pwd
> (...)/Scotty3D
$ tar cvzf handin.tgz src/ writeup.txt model.dae
> a src
> a src/main.cpp
(...)
> a writeup.txt
> a model.dae
```

The Autolab system will run a simple script that checks for the extra files. If the output indicates that something is missing, fix it or risk losing points!
