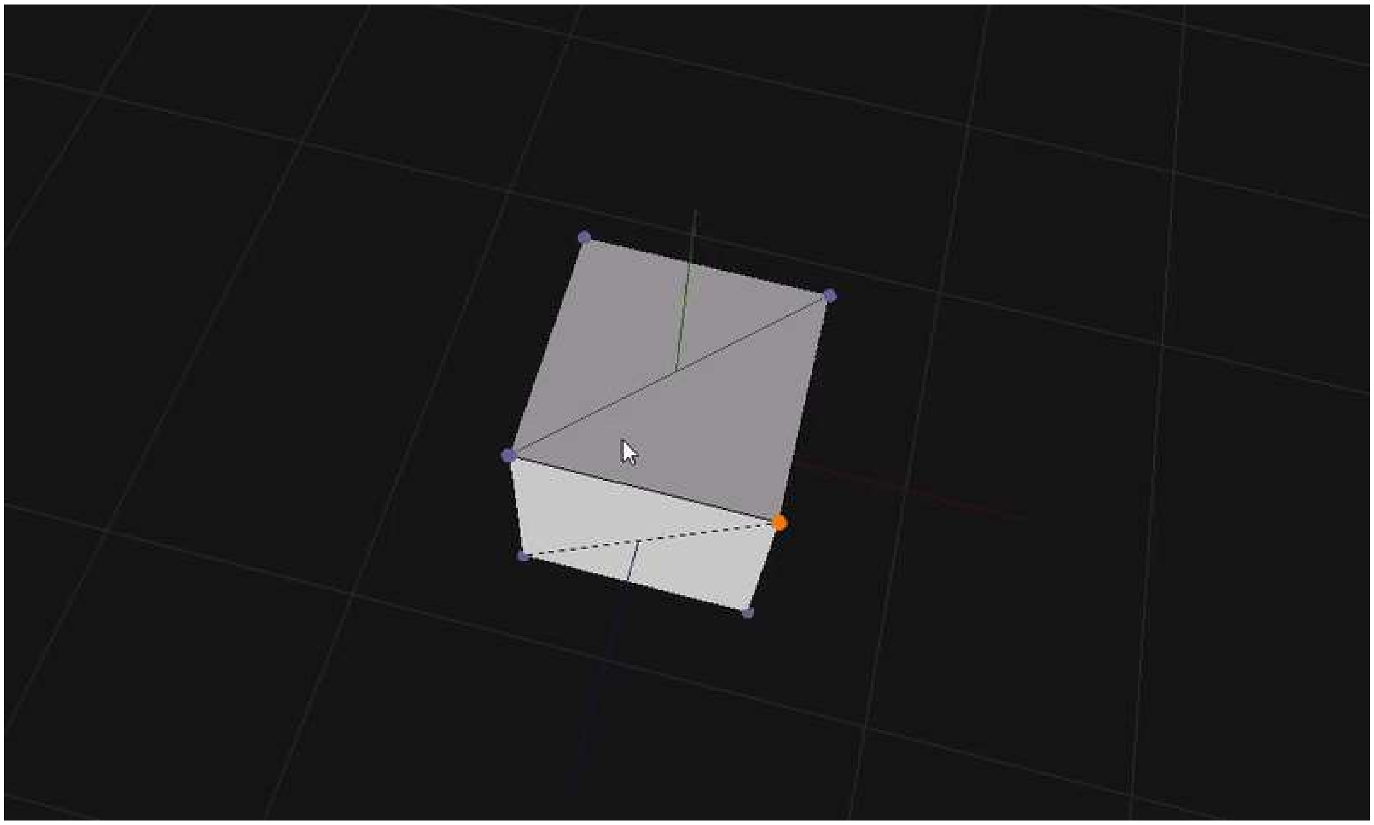
Assignment 3 - Mesh Editing

In this assignment you will build a half-edge data structure from a given mesh, implement edge and face operations to edit the mesh, triangulate simple convex polygons, And use all these operations to design an object of your choice.



Getting Started

You can use the same python and environment that you set up in the previous assignments, but there area additional requirements to install.

```
python -m pip install -r requirements.txt
```

These requirements include `libigl` and `pyrr` . The `libigl` module we will use for loading mesh geometry (Note that igl.read_obj Only support meshes with constant face-degree (triangles, quads, etc.) but not mixed degree faces), you will not need to use these 2 libraries in the objective implementations, except when you want to change the model you are loading.

Provided Code

The provided code includes 2 files:

- `halfedge_data_structure.py` where most of the implementation will be done.
- `core.py` which contains the main code for the visualization and user interaction, among other things.

For this assignment, we are only working with solid meshes, so no boundaries (holes). Also, the mesh should be a manifold. Quads and Ngons are supported, but assumed to be planar and convex.

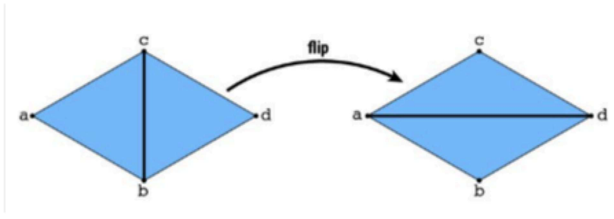
Key	Description
`F`	Flip selected edge (if an edge is selected)
`S`	Split selected edge (if an edge is selected)
`E`	hold for Face Extrude Mode (if a face is selected)
`B`	hold for Face Bevel Mode (if a face is selected)
`I`	hold for Face Inset Mode (if a face is selected)
`S`	hold for Face Scale Mode (if a face is selected)
`X`	set view to X axis(side view)
`Y`	set view to Y axis(top view)
`Z`	set view to Z axis(front view)
Left-Click	Select the an element on the mesh
Click+Drag	Rotate the camera around the mesh, if a face is selected, face operation will be performed instead as described above
Scroll	Zoom in/out
`W`	Write/Save the mesh to a file
`ESC`	Close the window

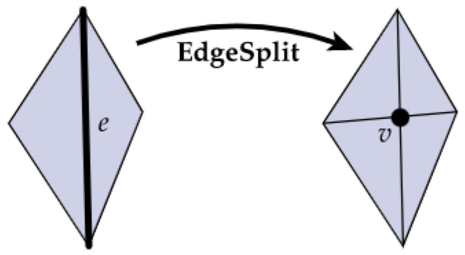
These contols will help you explore and run the operations you implement. See the code and comments for more information on how the provided code sets up each of these interactions.

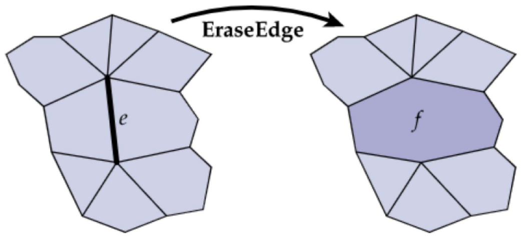
Objectives

1. (2 mark) Construct a Half-Edge data structure from the given mesh data (vertices, faces), specifically implement the `HalfedgeMesh.build()` method in `halfedge_data_structure.py` . In this method, you need to fill the properties of each half-edge, vertex, edge, and face in the mesh and connect them properly. Hint: You can use an edge dict to keep track of the half-edges you create.

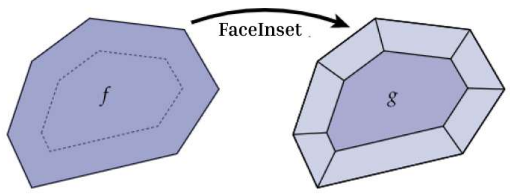
2. (1 mark) Implement the simple triangulate method `triangulate()` which triangulates a simple convex polygon.

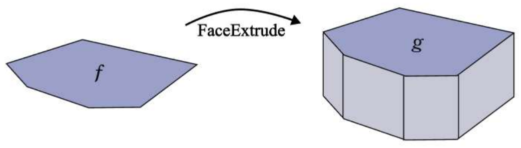


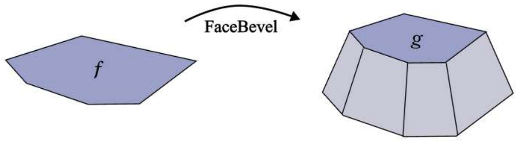




3. (3 marks) Edge Operations, implement the following edge operations methods, which can be run on a selected mesh edge:
   - 3a: `HalfedgeMesh.flip_edge()` : given an edge with two adjacent faces, flip the edge to connect the other 2 vertices, should only work if the 2 faces are triangles.
   - 3b: `HalfedgeMesh.split_edge()` : given an edge, split it by adding a new vertex in the middle of the edge and additionally connect the new vertex to the 2 corners facing the edge.
   - 3c: `HalfedgeMesh.erase_edge()` : Given an edge, remove it from the mesh, and merge the 2 faces connected by that edge. (3c Needs Objective 2 to be done first to work properly)







4. (3 marks) Face Operations, implement the following face operations methods:

- 4a: All of the face operations listed in 4b share the same topology update, so first implement `HalfedgeMesh.extrude_face_topological()` which can be used to implement the rest of the face operations. (Note that Objective 2 should be working before working on these objectives)
- 4b: After generating faces for each side, implement the following face operation methods:
    - `HalfedgeMesh.inset_face()` : scales the selected face inwards (or outwards) in its plane.
    - `HalfedgeMesh.extrude_face()` : translate the selected face along its normal.
    - `HalfedgeMesh.bevel_face()` : it combines the effects of extruding and scaling in/out, as shown in the image above.

5. (1 mark) Use the operations you implemented to model a simple object, and save the mesh to a file. Be creative!

Dependency: all Objectives depend Objectives 1. 3c (& 4a) depend on Objective 2. everything else is independent.

Finished?

Great! Submit your python implementation which should consist of these 2 files `halfedge_data_structure.py` and `core.py` (with your implementation) to the course website, along with your saved model that you created and saved to `output.obj` . Put your name, student number Use the comments at the top of the python files you submit. Likewise, put any other information you would include in a readme file into comments at the top of one of these files (e.g., request to use your late penalty waiver).

Note that you are encouraged to discuss assignments with your classmates, but not to the point of sharing code and answers. All code and written answers must be your own. Please see the course outline and the fine print in the slides of the first lecture.