Homework: Mesh Reduction

In this exercise you will

1. Implement the algorithm described in "Surface Simplification Using Quadric Error Metrics" (QEM) https://www.cs.cmu.edu/~./garland/Papers/quadrics.pdf, which is used to simplify an arbitrary triangle mesh to a user desired simplification ratio, i.e. the ratio of the number of triangles between the simplified mesh and the original one.
2. Implement the algorithm described in "Simplifying Surfaces with Color and Texture using Quadric Error Metrics" https://www.cs.cmu.edu/~garland/Papers/quadric2.pdf, which is an extension of the first task. This task is to simplify an arbitrary triangle model to a user desired simplification ratio. The attributes, e.g. normal, skinning weights, RGB, and uv, defined on the surface are also simplified so that the appearance of the simplified mesh won't change much.

Given a 3D model and a user specified simplification ratio r as the input, your main task is to implement the QEM method to reduce the complexity of the 3D model. By executing the program multiple times with different rs, you should be able to obtain a set of 3D models that can be assembled into a Level-Of-Details (LOD) representation of the original 3D model. In that case, the LOD representation will be imported into the rendering system given by other homework for various rendering tasks.

Note: The sample datasets provided actually includes two set of models, i.e. pure meshes and meshes with appearance properties, where each will be used by the two tasks.

1. Plain Mesh Reduction

Input: A 3D mesh M, and a user desired mesh simplification ratio r

Output: A 3D mesh M' with $|M'| = (1-r) \times |M|$, where $|M'|$ and $|M|$ are the numbers of triangles of M' and M respectively.

To implement the original QEM algorithm, the core components are:

- A data structure to store the mesh that is composed of vertices, edges, and triangles, and the relationships among these elements,
- Computation of the cost for an edge when collapsing it,
- Computation of the position of the newly generated vertex when collapsing an edge,
- A while-loop framework to repeatedly rank and collapse edges until the desired the number of triangles of the simplified mesh is achieved.

2. Mesh Reduction with Appearance Preservation

Input: A 3D mesh M with a set of properties except the uv position defined for each vertex of the 3D mesh, e.g. normal, RGBA, skinning weights, etc, and a user desired mesh simplification ratio r

Output: A 3D mesh M' with $|M'| = (1 - r) \times |M|$, and the updated vertex properties.

To implement the extended QEM algorithm, building on top of the implementation for task 1, the following parts need special cares:

- A data structure that can allow a flexible set of properties defined for each vertex,
- the extended QEM in high-dimension.

3. Mesh Reduction with A UV Map

Input: A 3D mesh M with a set of properties defined for each vertex including additionally a uv position as well, and a user desired mesh simplification ratio r

Output: A 3D mesh M' with $|M'| = (1 - r) \times |M|$, and the updated vertex properties.

To implement the extended QEM algorithm with UV maps, building on top of the implementation for task 2, the following parts need special cares:

- A data structure that can either allow a vertex mapped to multiple uv positions, or a pre-processing of the 3D mesh is needed to cut it open according to the given UV map so that a vertex has a unique uv position,
- The data structure should also support possibly multiple textures in the input.