Exercise 4. Surface Quality and Smoothing
Surface Smoothing

- Spectral analysis
- Diffusion flow
  - Uniform Laplace operator
  - Laplacian-Beltrami operator
- Energy minimization
Uniform Laplacian Surface Smoothing

- Uniform Laplace operator
  \[ L_U(v) = \left( \frac{1}{n} \sum_{i} v_i \right) - v \]
- Mesh smoothing
  \[ v' = v + \frac{1}{2} \cdot L_U(v) \]
- Implement uniform Laplace operator in `QualityViewer::calc_uniform_mean_curvature()` in `QualityViewer.cc`
- Implement uniform Laplacian smoothing in `SmoothViewer::uniform_smooth()` in `SmoothViewer.cc`

**Figure 1:** Mean curvature approximation on the face dataset before and after smoothing

A new project, **Smoothing**, has been added to the framework from the previous exercise. It reuses files from the **ValenceViewer** project and adds additional classes.

The **QualityViewer** class extends the **MeshViewer** and adds visualization modes like curvatures, triangle shapes, and reflection lines. The **SmoothingViewer** extends the **MeshViewer** and adds the smoothing operations which are triggered by the **N** and **U** keys. You will need to implement portions in the two new classes: **QualityViewer** and **the SmoothingViewer**.

To load one of the meshes for this exercise:
- Right click on the 04-Smoothing Project > Properties > Configuration Properties

To load the bunny, set **Command Arguments** to:

`..\..\data\bunny.off`

To load the face 1, set **Command Arguments** to:

`..\..\data\max.off`

To load the face 2, set **Command Arguments** to:

`..\..\data\scanned\face.off`

**3.1 Uniform Laplace curvature and smoothing**

a) The uniform Laplace operator approximates the Laplacian of the discretized surface using the centroid of the one-ring neighborhood. For a vertex \( v \) let us denote the \( n \) neighbor vertices with \( v_i \). The uniform Laplace approximation is

\[ L_U(v) = \left( \frac{1}{n} \sum_{i} v_i \right) - v \]

The half length of the vector \( L_U \) is an approximation of the mean curvature.

b) Implement uniform Laplace smoothing in the `uniform_smooth(unsigned int iters)` function of the **SmoothViewer** class. It has to apply \( iters \) smoothing operations on the mesh, where one smoothing operation moves the vertices of the mesh halfway along the \( L_U \) vector:

\[ v' = v + \frac{1}{2} \cdot L_U(v) \]

Hint: do not forget to update normals after vertex coordinates change.

Test your solution by loading the **scanned** face.off model. Choose the "Uniform mean curvature" mode and apply uniform smoothing by pressing the **U** button. You should get images similar to Figure ??.

**3.2 Triangle shapes**

Many applications require triangle meshes with nice triangles. Equilateral triangles usually are considered "nice", skinny or flat triangles are "bad". A measure to capture this quality is the circumradius to minimum edge length ratio. The lower this ratio is, the closer the triangle is to the equilateral (ideal) triangle. To derive a formula for the circum-radius to minimum edge length ratio...
Uniform Laplacian Surface Smoothing
• Assess triangle quality by the circumradius to the minimum edge length ratio

• Circumradius is computed by

\[ A = \frac{|a| \cdot |b| \cdot |c|}{4 \cdot r} = \frac{|a \times b|}{2} \]

• Implement in QualityViewer::calc_triangle_quality() in QualityViewer.cc
Triangle Quality
Laplace-Beltrami Operator

\[ L_B(v) = \frac{1}{2A} \sum_i ((\cot \alpha_i + \cot \beta_i)(v_i - v)) \]

• Compute mean curvature using Laplace-Beltrami weights in QualityViewer::
  
  `calc_mean_curvature()` in QualityViewer.cc

• Implement smoothing in SmoothViewer::
  
  `smooth()` in SmoothViewer.cc
Laplace-Beltrami curvature and smoothing
3.4 Gaussian curvature

In the lecture you have been presented an easy way to approximate the Gaussian curvature on a triangle mesh. The formula uses the sum of the angles around a vertex and the same associated area which is used in the Laplace-Beltrami operator:

\[ G = \frac{2\pi - \sum \theta_j}{A} \]

Implement the `calc_gauss_curvature()` function in the `QualityViewer` class so that it stores the Gaussian curvature approximations in the `vgausscurvature` vertex property! Note that the `vweight` property already stores \( \frac{1}{2}A \) value for every vertex, you do not need to calculate \( A \) again. For the bunny dataset you should get a Gaussian curvature approximation like on Figure ??.

3.5 For the passionate (optional)

Implement the "tangential smoothing" which moves vertices only in the tangent plane of the vertex, thus focuses on enhancing triangle shapes. For this, you need to project the uniform Laplace approximation back to the tangent plane of the vertex. Use this projection vector to compute the new position of the vertex. Notice that you need to store the original normal of the vertex additionally, in order to keep the vertices always on the original tangent plane, even after several smoothing iterations.
Gaussian Curvature
Submission

- **Deadline:** *Wednesday, March 11, 2015 11:59pm*
- Upload a .zip compressed file named “Exercise4-YourName.zip” to Blackboard, same as before
- Include a “read.txt” file describing how you solve each exercise and the encountered problems
• email (include “CSCI_599” in title):
  olszewski.kyle@gmail.com, peilun.hsieh@usc.edu

• Highly recommended to post your questions on Blackboard
http://cs599.hao-li.com

Thanks!