Intro. to Geometry Processing 600.756

http://www.cs.jhu.edu/~misha/Spring17a/

Goals

To learn the fundamental techniques required for a wide-class of geometry processing applications:

- (Discrete) Differential Geometry
- Linear Solvers
- PDEs
- Spectral Analysis

How this Works

- We meet once a week to discuss the reading (led by Fabian).

  You will do the reading before class!!!

- There will (most likely) be homework assignments to help you become familiarized with key concepts.

Reading

- Keenan Crane's notes on discrete differential geometry:

  Digital Geometry Processing with Discrete Exterior Calculus

- Each week will discuss the next chapter
  (or continue with the current one, or go off on a paper-tangent)

Assignment 1

Preliminaries:

1. Install MeshLab: http://www.meshlab.net
2. Download the 3D models*
3. Download the code-base*
   - Make sure it compiles/runs on whatever (non-Mac) platform you use
   - Make sure you can read in a mesh and write it out in either binary or ASCII
     - Validate your output by loading it in MeshLab

*These will likely evolve over the semester

Assignment 1

Preliminaries:

- We will be working with manifold meshes (possibly with boundary).
- They will be read/written as:
  - vertices: positions + ...

    PlyVertex< float > , PlyColorVertex< float > , PlyOrientedVertex< float >

    All support standard arithmetic operations

    All have a member "Point3D< float > position"
  - triangles: triplets of indices into the vertex list

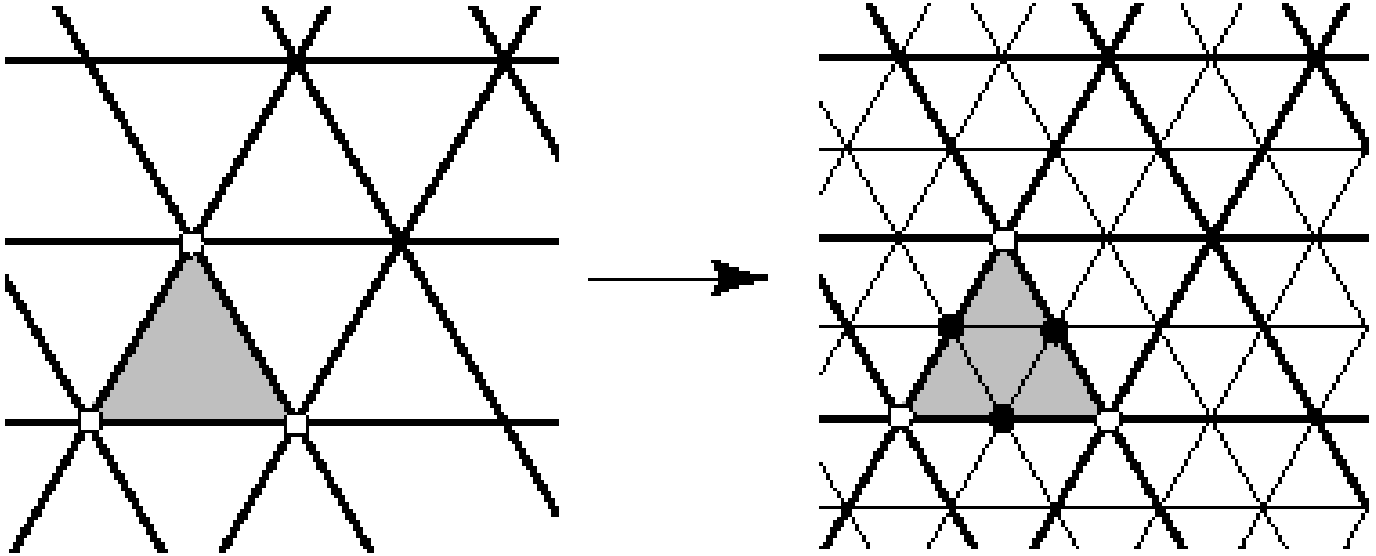
    All have an operator that returns the index of the i-th vertex

    unsigned int& operator[] (unsigned int i){ ... }

Assignment 1

Mesh Refinement:

1. Modify the code so that it performs a 1-to-4 subdivision of each triangle (with the new vertex equal to the average of its endpoints).
2. Do this so that there is only one instance of each new vertex.



Zorin & Schroeder
SIGGRAPH 99
Course Notes

Assignment 1

Mesh Smoothing:

1. Modify the code so that it takes two new parameters:
   a. $\boldsymbol{I}$: An integer parameter
   b. $\boldsymbol{\alpha}$: A floating point value in the range [0,1]
2. Run $\boldsymbol{I}$ iterations and in each one:
   - For each vertex $v \in V$:
     - Compute the average position of the vertex's neighbors: $\bar{v}$
     - Replace the vertex's position with the weighted average of its neighbors:

$$v \leftarrow v \cdot (1 - \alpha) + \bar{v} \cdot \alpha$$

3. Make this fast (you can use OpenMP if you like)
4. Describe/discuss the output you get as a function of the number of iterations and the blending weight.