Assignment 2

February 22, 2016

INSTRUCTIONS

This is the second coursework for the course COMPGV18/COMPM080 on Acquisition and Processing of 3D Geometry. The subject of this coursework is to implement and evaluate Laplacian filtering in the context of mesh denoising. The coursework is worth 100 points. You are encouraged to use C/C++ for this assignment.

Please submit in a short report summarizing your work (to be submitted via Moodle). The report should start with a brief list of the questions you attempted and to what extent you have achieved each item. Next, write a short description of the methods you used for each part together with any conclusions you have drawn from your experiments. *Please include result images.*

You will be required to present your work in a one to one session (date to be decided) demonstrating what you have implemented.

LATE POLICY The coursework is due **March 20th** (Sunday 23:55). We will use the following late policy:
Submissions after the deadline but within 24 hours of it will be marked down to 90% of the number of points achieved.
Submissions later than 24 hours, but within 48 hours from the deadline will be marked down to 80% of the number of points achieved.
Submissions later than 48 hours will receive 0% (but will be marked).

1 Core Section

In the course of handling range data or 3D scans, noise is a common source of problem. Denoising in this context refers to algorithms that attempt to remove noise, while preserving actual object features. The key challenge is to differentiate between object features and imperfections arising due to random noise. You are expected to write your own code for all tasks.

Discrete Curvature

Given a mesh, compute at each vertex the discrete principal curvatures κ_1, κ_2. First you will need to compute mean curvature (H) and Gaussian curvature (K). There exist several approximations to the continuous mean curvature, they differ in how they weight the influence of the neighbourhood of the vertices.

1. Uniform Laplace: Compute the Laplace using uniform discretization to find the mean curvature H at each vertex. Make sure to handle cases where the approximations don't allow the calculation of valid estimates.

2. Having the mean curvature H at each vertex, you'll need the Gaussian curvature K to estimate the principal curvatures. Compute the angle deficit $(2\pi - \sum_j \theta_j)$ at each vertex of a mesh. Then compute the area of the centroidal polygon to get the Gaussian curvature K. Use H and K to estimate κ_1, κ_2.

3. Visualize and compare your results. Colour code the mesh vertices using the two principal curvatures. Do they match your intuition (*e.g.*, concave vs. convex regions)? (35 points)

4. Non-uniform (Discrete Laplace-Beltrami): Compute the Laplace-Beltrami using cotangent discretization to estimate mean curvature and re-estimate κ_1, κ_2. Do the results improve? (15 points)

Laplacian Mesh Smoothing

For the following tasks, please perform your experiments on a simple cube mesh as well as more complicated models.

5. Implement explicit Laplacian mesh smoothing ("Diffusion Flow on Meshes" in the week 5 slides), and colour code before/after results. What is a good λ step size to use? What happens, if your step size is too large? Make sure to use double-precision to avoid numerical problems[1]. (20 points)

6. Implement implicit Laplacian mesh smoothing from (Desbrun et al.: Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow, SIGG99, Equation (9)). Compare your results to the previous method. (20 points)

7. Evaluate the performance of Laplacian mesh *denoising* in the context of data smoothing. Design your tests (*e.g.*, adding increasing amounts of synthetic noise) and report your conclusions, specifically regarding when the method works and when it fails. (10 points)

Some model databases: (i) *e.g.*, Dragon: `http://graphics.stanford.edu/data/3Dscanrep/`, (ii) *e.g.*, Cube.off from `http://people.sc.fsu.edu/~jburkardt/data/data.html`

[1] `http://eigen.tuxfamily.org/dox/group__TutorialLinearAlgebra.html`