Digital Geometry Processing (236329)
Homework #2

Due date: 01/12/2022

Introduction

In this exercise you will implement basic tools for working with meshes. To evaluate your code, you are required to test it on several meshes. Please compose your results into a well-documented report where you show plots for various meshes/functions. Note that you should explain every result you get, i.e., explain what you did and what can be seen in the figure so that you report is self-contained. The report should contain demonstrations of all the tasks. Submit your report and your MATLAB code.

Note that you should avoid loops wherever possible. While it might seem not important at first, working later with high resolution meshes will become impossible when using a non-optimized code.

Objectives

Your goal is to write several functions (or classes) in MATLAB that implement the following:

1. Read/Write .off files: these functions should allow loading .off files into MATLAB or saving meshes as .off files. Here is a link to the .off file format specification. In the course website there are several meshes that your code should be able to load. **Tip**: notice that .off indices start from 0 and that whole sections of files can be read without loops!

2. Data structure for meshes: as we discussed in the tutorial, *shared vertex* data structure with an *adjacency matrix* should be well suited for the implementation of many of the algorithms that we will learn. Nevertheless, we encourage you to experiment with other data structures as well.

3. Mesh visualization: this function should render the mesh onto a MATLAB figure, possibly allowing the user to manipulate the scene (rotate, etc.). Now, you can test the functions you wrote to display several meshes and attach the results to your report. **Tip**: shared vertex matrices + patch.

4. Vertex/Triangle areas: these functions should return a vector of the vertex areas and triangle areas, respectively. While triangle areas are well defined, many options exist for vertex areas. We will simply use the sum of the neighboring (to the current vertex) triangle areas divided by three. That is, if v_i is a vertex and N_i is its 1-ring neighborhood of faces, then $A_{\mathcal{V}}(v_i) = \sum_{j\in N_1} A_{\mathcal{F}}(t_j)/3$, where $A_{\mathcal{V}}(v_i)$ is the vertex area and $A_{\mathcal{F}}(t_j)$ is the triangle area. Explain why the area of adjacent faces should be divided by 3. **Tip**: vectorize your code using the vertex-face adjacency matrix, what should be the matrix values (instead of ones)?

5. Function visualization: your code should allow the user to display a discrete function on top of a mesh. Please make sure that you support both functions on vertices (i.e., one value per vertex) and functions on faces (i.e., one value per face). You can test your code from 4 by

displaying the area functions for several meshes and attach the results to your report. **Tip**: see the documentation for patch with *CData* and *FaceVertexCData*.

6. Interpolation: at times, we might need to interpolate a given function from vertices to faces and vice versa. Here, we represent the interpolants as linear operators, i.e., as matrices which act on discrete functions (vectors) and return discrete functions. For instance, $I_{\mathcal{V}}^{\mathcal{F}} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{F}|}$ interpolates quantities from faces to the vertices, that is $I_{\mathcal{V}}^{\mathcal{F}}(i,j) = \frac{A_{\mathcal{F}}(j)}{3A_{\mathcal{V}}(i)}$ iff face j belongs to the one-ring of vertex i and 0 otherwise, where $A_{\mathcal{F}}(j)$ is the area of face j, $A_{\mathcal{V}}(i)$ is the area of vertex i. The analogous operator which interpolates from vertices to faces is defined implicitly, i.e., $I_{\mathcal{F}}^{\mathcal{V}} \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{V}|}$ such that $I_{\mathcal{F}}^{\mathcal{V}} = A_{\mathcal{F}}^{-1}\left(I_{\mathcal{V}}^{\mathcal{F}}\right)^T A_{\mathcal{V}}$, where A_* are diagonal matrices with vertex/face areas. Multiplying $I_{\mathcal{V}}^{\mathcal{F}}$ by a column vector with a scalar value per face, interpolates these values to vertices, and multiplying $I_{\mathcal{F}}^{\mathcal{V}}$ by a column vector with a scalar value per vertex, interpolates these values to faces.
If a function that is defined per vertex is interpolated to the faces and back to the vertices, will the result be identical to the original function? What about functions that are defined per face and interpolated back and forth? Explain.

7. Topological measures: this function should return the genus and the number of boundary edges of an input mesh. Tip: Euler characteristic and an adjacency matrix for edges, i.e., for each edge, store its neighboring faces.

Analysis with MATLAB

The following manipulation/analysis tasks should be completed. In addition, you should think of two non-trivial tasks similar in "spirit" to the ones below, describe them in your report and attach the corresponding results.

1. Triangulation error: here our goal is to check numerically how fast the discretization error decreases as we take denser meshes, by comparing the unit sphere with meshes that approximate it. We will use meshes with the prefix "sphere_si", where $i \in \{0,..,5\}$.
For every mesh, you should compute the error of the current triangulation compared to the ground truth (the unit sphere). You should measure the error in the following fashion:
$E_i = \frac{1}{|\mathcal{F}|}\sum_j \|X_j - S_j\|$, where j is an index of a face, X_j is the face center, S_j is the point on the (true) unit sphere that corresponds to X_j and $|\mathcal{F}|$ is the number of faces.
Plot the error as a function of e_i, the average edge length of mesh i.
What is the rate of convergence? Does it make sense? **Tip**: to compute S_j you should use the equation for the unit sphere: $x^2 + y^2 + z^2 = 1$, which x, y, z correspond to X_j?

2. Valence: the amount of incoming/outgoing edges from a certain vertex, also called the valence of a vertex, is an important quantity. Compute the valence of every vertex for several meshes and plot the result. What is the average valence? **Tip**: adjacency matrix (please avoid loops here).

3. Kernel of linear operators: the kernel of a linear operator A is defined as the set of all vectors which are mapped to zero under A, i.e., $A(v) = 0$. The kernel of an operator is important for the analysis of a method, as it implies certain possible issues (think of a non-zero function being mapped to zero all of a sudden). Does the operator $I_{\mathcal{V}}^{\mathcal{F}}$ has a kernel? What about $I_{\mathcal{F}}^{\mathcal{V}}$? In case

the kernel of these operators is non-empty, compute it, and show some of the functions in the kernel. **Tip**: null and try it on *small* meshes, e.g., $|\mathcal{V}| < 1000$.

Submission

The HW can be done in either pairs or singles. Please submit a single **zip** file containing your report and MATLAB code using the electronic submission button in the course's website. For evaluation of your work, we will mainly focus on the attached report, please consider it while composing your report and strive to make it as detailed as possible.
Please do not include your names in the submission, ID only.

Good Luck!