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. Along with your code, please compose your numeric 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 your report is self-contained.

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

Objectives

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

1. Read/Write .off files: these functions should allow loading .off files into MATLAB or unloading these files into the Hard Disk. Here is a link to the .off file format specification and another link for several meshes 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 show. 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 $A_V(v_i) = \sum_{j \in N_i} A_f(t_j)/3$, where $A_V(v_i)$ is the vertex area and $A_f(t_j)$ is the triangle area. Tip: vectorize your code using the 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 and return discrete functions. For instance, $I^F_V \in \mathbb{R}^{|V| \times |F|}$ interpolates quantities from faces to the vertices, that is $I^F_V(i,j) = \frac{A_{x(j)}}{3A_{y(i)}}$ if face $j$ belongs to the one-ring of vertex $i$ and 0 otherwise. The analogous operator which interpolates from vertices to faces is defined implicitly, i.e., $I^V_F \in \mathbb{R}^{|F| \times |V|}$ such that $I^V_F = A^{-1}_{x} (I^F_V)^T A_{V}$, where $A_*$ are treated in this case as diagonal matrices.

**Analysis**

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 would be to check numerically how fast the discretization error decreases as we take denser meshes. To this end, we will use meshes with the prefix “sphere_s$i$”, where $i \in \{0, \ldots, 5\}$. For every mesh, you should compute the pair $(e_i, E_i)$, where $e_i$ is the average edge length for this mesh, and $E_i$ is the error of the current triangulation compared to the ground truth. Notice that since the vertices are likely to be sampled on the (true) sphere, it makes sense to measure the error from the triangle centers. Basically, you should measure the error in the following fashion: $E_i = \frac{1}{|F|} \sum_j \|X_j - S_j\|$, where $j$ is an index of a face, $X_j$ is the face center, $S_j$ is the ground truth point on the (true) sphere and $|F|$ is the number of faces. Finally, plot all of the computed pairs into a single graph. 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 $I^F_V$ has a kernel? What about $I^V_F$? 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., $|V| < 1000$.

**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.

Good Luck!