Analysis of Three-Dimensional Shapes
F. R. Schmidt, M. Vestner, Z. Lähner
Summer Semester 2017

Computer Vision Group
Institut für Informatik
Technische Universität München

Weekly Exercises 4

Room: 02.09.023
Wed, 14.06.2017, 14:00-16:00
Submission deadline: Tue, 13.06.2017, 23:59 to laehner@in.tum.de

Mathematics: Triangle Meshes

Exercise 1 (4 points)**.** Recap the hat functions on triangles, the first fundamental form and the derivation of the mass matrix in 2D from the lecture.

1. Derive the mass matrix for 3D triangular meshes with the hat functions ϕ_i as a basis. Each entry can be calculated by taking $M_{ij} = \int_{\mathcal{M}} \phi_i(p)\phi_j(p)\mathrm{d}p$. It will help to use a reference triangle with vertices $(0,0), (1,0), (0,1)$ and deconstruct the integral for each triangle. The final matrix should look like this:

$$M_{ij} = \begin{cases} \frac{\text{area}(T_1)+\text{area}(T_2)}{12} & (i,j) \text{ is an edge} \\ \sum_{T_k \in N(i)} \frac{\text{area}(T_k)}{6} & i = j \\ 0 & \text{otherwise} \end{cases}$$

   T_1, T_2 are triangles that share the edge (i,j). $N(i)$ is the set of triangles adjacent to vertex i.

2. Show that the mass matrix is symmetric and positive definite. With this properties we know that it defines an inner product $\langle \mathbf{f}, \mathbf{g} \rangle_{\mathbf{M}} = \mathbf{f}^\top \mathbf{M} \mathbf{g}$ (and therefore also induces a norm).

Programming: Also Triangle Meshes

Download the supplementary material from the homepage. It contains two files describing a 3D triangular mesh.

Exercise 2 (3 points)**.** We represent 3D shapes as triangle meshes $\mathcal{M} = (\mathcal{V}, \mathcal{F})$ embedded in $\mathbb{R}^3$. They consist of vertices $\mathcal{V} = \{v_1, ..., v_n\}$ and triangles $\mathcal{F} \subset \mathcal{V} \times \mathcal{V} \times \mathcal{V}$ connecting them to a closed surface. The coordinates of each vertex are denoted by $p(v_i) \in \mathbb{R}^3$. Note that the vertices of a triangle $t = (u, v, w) \in \mathcal{F}$ are ordered. We define triangles to be identical if they can be transformed into each other by a cyclic permutation, i.e. $(u, v, w) = (v, u, v) = (w, u, v)$ but $(u, v, w) \neq (w, v, u)$. The order defines the direction of the normal and is also called orientation.

1. There are different formats to store triangle meshes. The easiest one is the vert-tri format consisting of two files, one containing the 3D coordinates and the other one the triangles. The file `name.vert` contains n rows with three

double numbers separated by tabs. Each row represents one 3D coordinate. The file `name.tri` (note that the file name has to be the same as for the .vert) contains m rows with three positive integer values $x \leq n$. Each x refers to a row/vertex from the .vert file. The order of x in each row defines the orientation of the triangle. Write a function `read_vert_tri.m` that takes a filename and returns a struct `M` with fields `M.VERT` a $n \times 3$ array and `M.TRIV` a $m \times 3$ array. You can try your function on the cat0 files from the supplementary material (and see if you did everything correctly in the next part).

2. Triangulated surfaces can be plotted in Matlab with `trisurf(TRI, X, Y, Z)`. You need to give each column of `M.VERT` separatly to the function. The colors you see on the surface represent the height function.

3. Implement a function `facearea` that takes a triangle mesh struct as an input and returns a $\mathbb{R}^m$ vector containing the area of each triangle. You can plot your function by using `trisurf(TRI, X, Y, Z, f)` where f is a vector with as many entries as the numbers of vertices or faces. (Of course the entries in the vector must be ordered in the same way as the vertices or faces they correspond to.)

4. Implement a function `mass_matrix.m` that takes a triangle mesh and returns a `sparse` (that's a Matlab type) mass matrix as in Exercise 1. (No need to do anything we it, we will use it in later exercise sheets.)