Prof. Dr. Leif Kobbelt
J. Born, M. Ibing,
J. Möbius, P. Schmidt

Basic Techniques in Computer Graphics

Assignment 8

Date Published: December 19th 2016, Date Due: January 9th 2017

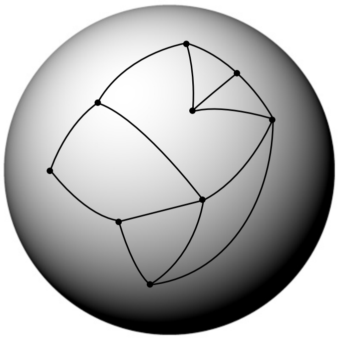
- All assignments (programming and text) have to be done in teams of 3–4 students. Teams with less than 3 or more than 4 students will receive no points.
- Hand in **one solution per team per assignment**.
- Every team must work independently. Teams with identical solutions will receive no points.
- Solutions are due 18:00 on January 9th 2017. Late submissions will receive zero points. No exceptions!
- Instructions for **programming assignments**:
  - Download the solution template (a zip archive) through the L^{2}P course room.
  - Unzip the archive and populate the `assignmentXX/MEMBERS.txt` file. Any team member not listed in this file will not receive any points! (Also see the instructions in the file.)
  - Complete the solution.
  - Prepare a new zip archive containing your solution. It must contain exactly those files that you changed. **Only change those files you are explicitly asked to change in the task description.** The directory layout must be the same as in the archive you downloaded. (At the very least it must contain the `assignmentXX/MEMBERS.txt`.)
  - Upload your zip archive through the L^{2}P before the deadline.
  - Your solution must compile and run correctly **on our lab computers** using the exact same `Makefile` provided to you. If it does not, you will receive no points.
- Instructions for **text assignments**:
  - Prepare your solutions on paper.
  - If you write your solution by hand, write neatly! Anything we cannot decipher will receive zero points. No exceptions!
  - If you hand in more than one sheet, staple your sheets together. (No paper clips!)
  - Put the names and matriculation numbers of all team members onto every sheet.
  - Unless explicitly asked otherwise, always justify your answer.
  - Be concise!
  - Put your solution into the designated drop box at our chair before the deadline. (1st floor, E3 building.)

Prof. Dr. Leif Kobbelt
J. Born, M. Ibing,
J. Möbius, P. Schmidt

Exercise 1 Meshes **[3 Points]**

(a) **[1 Points]**

Consider the depicted closed mesh (drawn on a spherical surface to underline its closedness). Depict the dual mesh of this mesh (in a way that every vertex, edge, and face is visible, i.e. don't draw on the backside of the sphere). Remember that duality in its core is a purely topological concept, i.e. you can place the dual vertices arbitrarily in the primal faces, edges don't have to be straight, faces don't have to be planar.

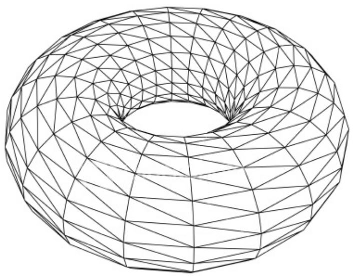


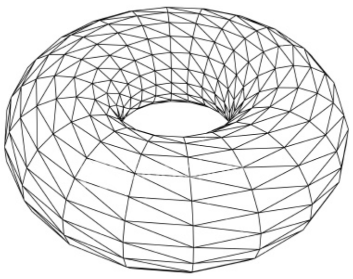
(b) **[1 Points]**

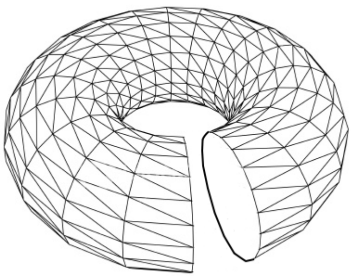
In the lecture you have seen how the average vertex valence of a triangle mesh as well as the relation between the number of faces and the number of vertices in a triangle mesh can be derived from the Euler formula. Remember that the Euler formula not only holds for triangle meshes, but also for arbitrary polygon meshes. Derive the average vertex valence of a closed hexagon mesh (of genus 1) as well as the relation between the number of faces and the number of vertices in such a mesh from the Euler formula.

(c) **[1 Points]**

In the lecture you have learned about the Euler formula $V - E + F = 2(1 - g) = \chi$ for polygon meshes, where g is the genus of the mesh and χ is called the *Euler characteristic* of the mesh. Which Euler characteristic does the closed triangle mesh (i) have and why? Which Euler characteristic does the closed polygon mesh (ii) have and how can you derive that from (i) (take a close look!)? Now we remove one ring of faces and edges from the triangle mesh (i) and obtain (iii) (which is rendered with backface culling, hence you cannot see the interior). Which Euler characteristic does the resulting non-closed mesh (iii) (with two loops of boundary edges) have and how can you derive that from (i)? Now imagine an arbitrary non-closed triangle mesh with exactly one loop of boundary edges (edges with only one incident face) that has the same number of boundary edges as each of the two holes in (iii) and the highest Euler characteristic possible in this case. Which Euler characteristic does this triangle mesh have? Due to the same number of boundary edges, you can insert two copies of this mesh to close the two holes in (iii). How does the Euler characteristic change when closing the two holes in (iii) this way? What genus does the resulting closed mesh have?


(i)

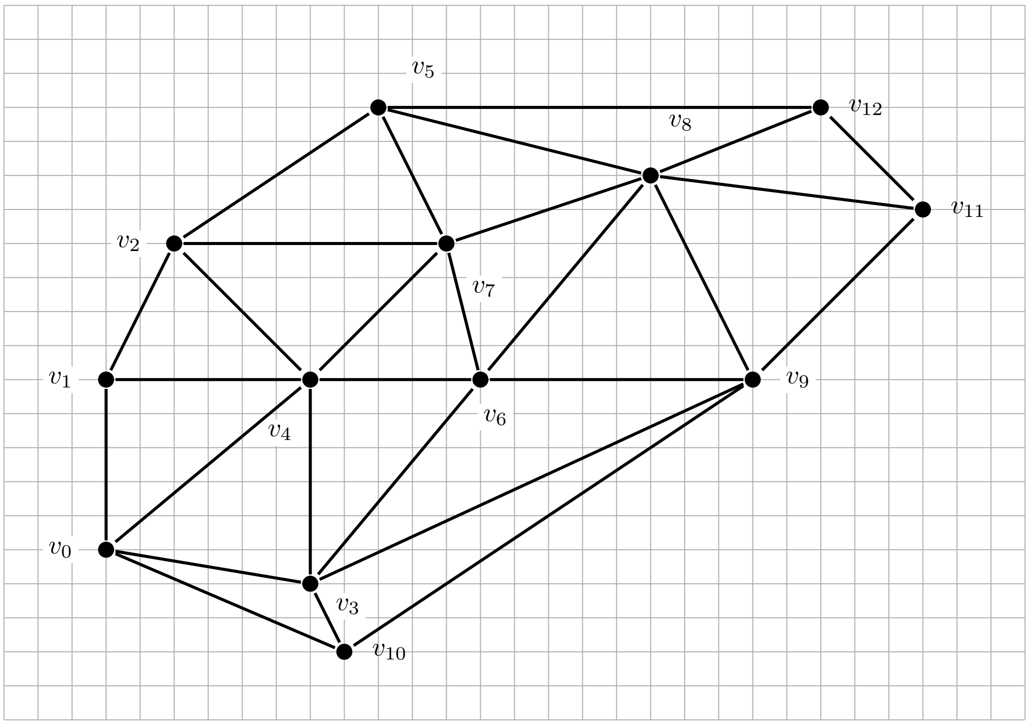

(ii)


(iii)

Exercise 2 Valence Coding **[4 Points]**

(a) Encoding **[2 Points]**

Use valence coding to store the topology of the mesh depicted below. Complete the following list $(5, 5, 6, \dots)$ with the initial triangle (v_6, v_7, v_4).

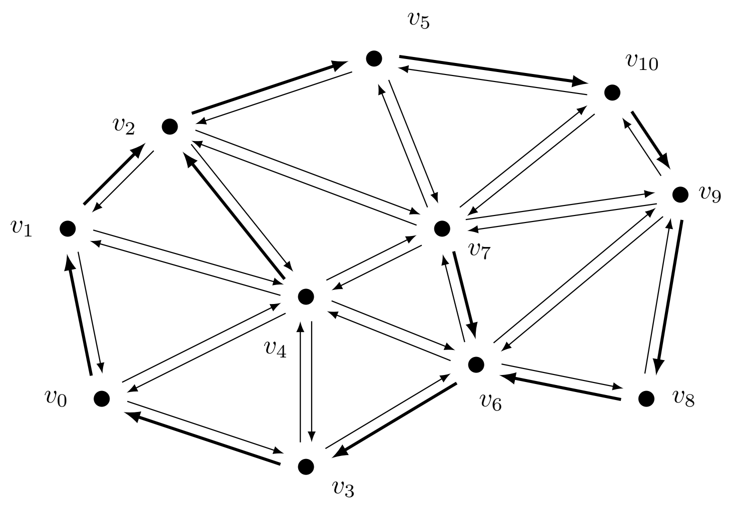


(b) Decoding **[2 Points]**

Decode the following valence code $(4, 5, 7, 5, 5, 4, 5, 4, 4, 4, 7)$, and derive the respective topology. The last vertex is marked as a *ghost*-vertex.

Exercise 3 Halfedge Data Structures **[3 Points]**

Consider the following triangle mesh represented by a halfedge data structure:



Navigation on a halfedge data structure is achieved by applying a sequence of the following elementary operations on the halfedges H and vertices V of the mesh:

- $n(h) : H \to H$: returns the **next** halfedge following h inside the same face (in counterclockwise direction).
- $o(h) : H \to H$: returns the **opposite** halfedge of h.
- $v(h) : H \to V$: returns the vertex that a halfedge h **points to**.
- $h(v) : V \to H$: returns an **outgoing** halfedge from a vertex v. In the above picture, the outgoing halfedge for each vertex is indicated by a bold arrow.

For example, in the given triangle mesh, $v(h(v_4)) = v_2$, and $v(n(h(v_4))) = v_1$.

(a) **[0.5 Points]**

Describe a sequence of operations that navigates from vertex v_0 to v_7.

(b) **[0.5 Points]**

Using a sequence of the operations provided above, implement a new operation $p(h) : H \to H$ that returns the **previous** halfedge of h (i. e. the halfedge h' such that $n(h') = h$). You can assume that the given mesh is a triangle mesh without boundaries.

(c) **[2 Points]**

You are given an additional operation $p(v) : V \to \mathbb{R}^3$ returning the 3D position of a vertex v. Using this, give an algorithm in pseudo-code that computes a vertex normal for a given vertex v by averaging the normals of the incident faces. You can assume that the given mesh is a triangle mesh without boundaries.