Series 4

Group 6

Tasks 1:

> Describe two methods to approximate vertex normals of triangle meshes and discuss their advantages and disadvantages.

Vertex normals in a triangle mesh can be approximated a number of ways. Two common ones are:

Flat shading normals

In this method, the normal of each vertex is the same as the normal of the triangle it belongs to. After computing the normal of a triangle by taking the normalized cross product of two of its edges, this can then be applied to the vertices of the triangle.

Advantages

- The fastest method of computing normals

Disadvantages

- When used in shading, the resulting mesh will look faceted, as the normals are constant across each triangle
- Vertices that are shared by multiple triangles will have different normals depending on the context of the triangle they are in.

Smooth shading normals

In this method, the normals of faces are still calculated as in flat shading, but vertex normals are calculated by averaging the normals of the faces that share that vertex. This may be done in combination with weighing, for example by size of the triangles or their angle incident to the vertex.

Advantages

- In display, the resulting mesh will look smooth, as the normals are interpolated across each triangle
- One normal per vertex, regardless of the number of triangles that share it

Disadvantages

- Slower than flat shading, as it requires averaging and weighting.

We have implemented this method in practical part 1.

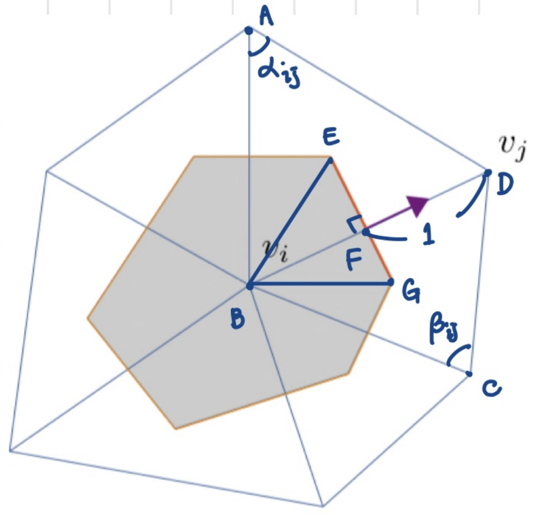
Task 2:

See last 2 pages.

Task 3:

Ex 2 : Theory

Task 3



Let $\overline{FD} = 1$.

Then $\overline{BF} = \overline{FD} = 1$, $\angle BEF = \alpha_{ij}$, $\angle BGF = \beta_{ij}$.

($\because$ E, G are circumcenters.)

Therefore, $\overline{EF} = \cot(\alpha_{ij})$,

$\overline{FG} = \cot(\beta_{ij})$.

$\therefore \|e^*_{ij}\| \pm = \cot(\alpha_{ij}) + \cot(\beta_{ij})$, $\|e_{ij}\| = 2$.

$$\therefore \frac{\|e^*_{ij}\|}{\|e_{ij}\|} = \frac{1}{2}\{\cot(\alpha_{ij}) + \cot(\beta_{ij})\}.$$

Task 4:

a)

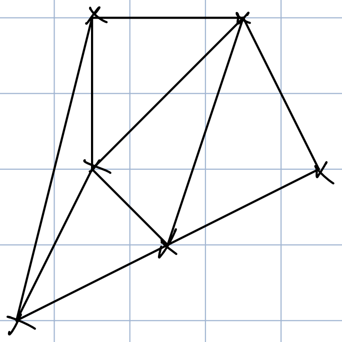
The Cotan-Laplacian can be calculated with the matrix L and the mass matrix M. M consists by Definition only of diagonal values >= 0 as it represents a weighting with the area of the voronoi cell surrounding a vertex. Thus, we should get a negative value in the Cotan-Laplacian, if L has for the case of $v_j \in N_i$ at least one value where $\cot \alpha_{ij}+\cot \beta_{ij}$ is negative. This is the case for $\alpha_{ij},\beta_{ij} > 90°$. E.g. a mesh with {(-1,4,0),(0,0,0),(0,8,0),(1,4,0)} where the first and last three values form a triangle each.

b)

The computation of the cotan-Laplacian operators can become numerically instable, if the cotan-Laplacian contains many values that are extremely large and/or small. An example case are triangles containing angles that are close to 0°/180° leading to large cotan values. E.g. two triangles in {(0, 0, 0),(1, 0, 0),(0, 1, 0),(0,1.000000000000000000000001,0)}.

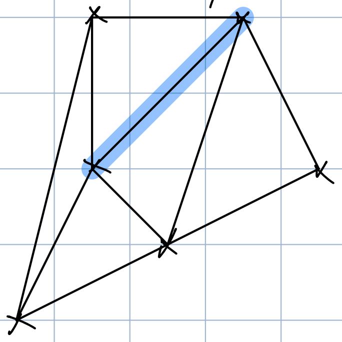
SPLIT

0. Base



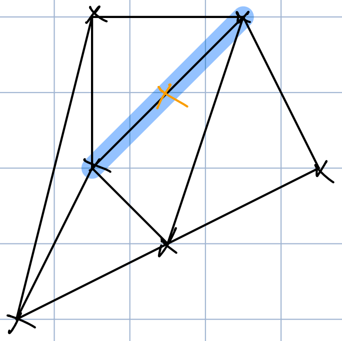
Splitting an edge is the process of dividing an existing edge into two parts. This can be done by inserting a new vertex into the middle of an edge, then updating the mesh topology to connect iwth it appropriatly

1. Identify



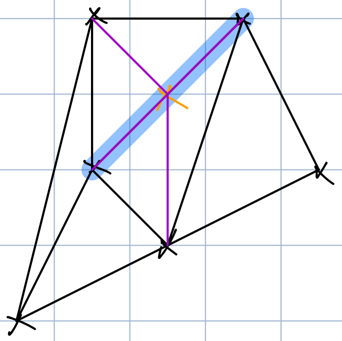
Identify the edge you want to split.

2. Insert



Create a new vertex along the edge you want to split. The vertex is commonly placed at the midpoint of the edge, but it can be placed anywhere along the edge depending on optimization requirements.

3. Retriangulate

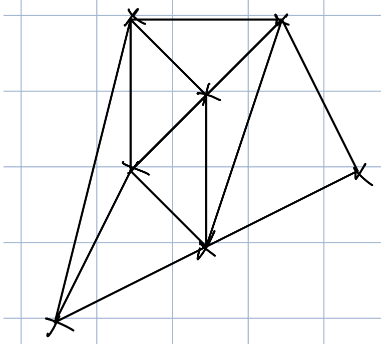


Now, split the original edge into two new edges. The new edges will each connect one of the original edge's vertices with the new vertex.

Update the faces that were adjacent to the original edge. In a triangle mesh, this will be two triangles, or one if the split edge was an outer edge. Each original triangular face is split into two faces, consisting of one vertex of the original edge, the vertex unrelated to the original edge, and the inserted vertex.

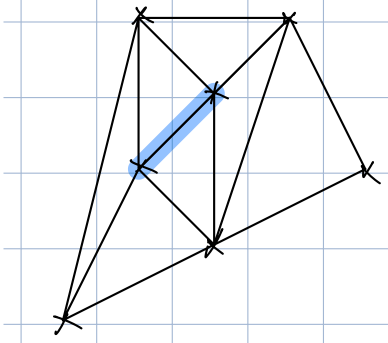
COLLAPSE

0. Base



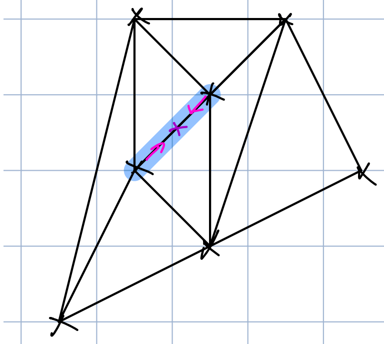
The process of edge collapse refers to the merging of two vertices of a mesh into one, reducing the number of verts, edges and faces in the mesh. It is essentially the opposite operation of edge splitting.

1. Select Edge



Select the edge you want to collapse.

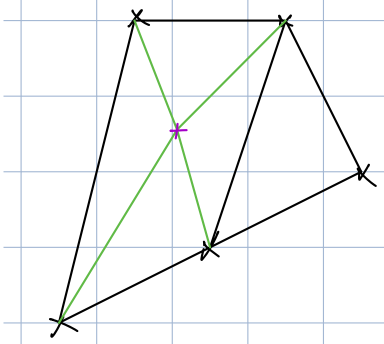
2. Unify verts



Unify the two vertices of the edge. this can be done in several ways. For example, one vertex can simply be moved to the position of the other. In this illustration, I chose to remove one vertex, and move the other to the midpoint of the edge that connected them.

The original edge has now become a point.

3. Update edges & faces



Update the edges and faces that were connected to the removed vertex to now connect to the kept (and moved) vertex. If this results in any degenerate faces (like a triangle with two identical vertices), remove those as well.