Edge Contraction Algorithm

`Topics:` edge contraction, decimation, hierarchy, numerical error.

Edge contraction. The basic operation in simplifying a triangulated surface is the contraction of an edge. Let K be a pure 2-complex and assume for the moment that $|K|$ is a 2-manifold. The contraction of an edge $ab \in K$ removes ab together with the two triangles abx, aby and it mends the hole by gluing xa to xb and ya to yb as illustrated in Figure 1. Vertices a and b are

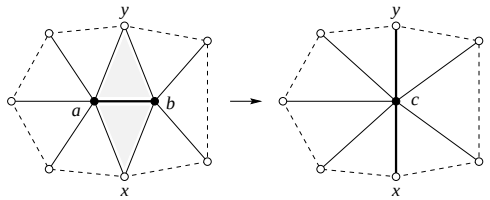

Figure 1: The contraction of edge ab. Vertices a and b are glued to a new vertex c.

glued to form a new vertex c. All simplices in the star of c are new, and the rest of the complex stays the same. To express this more formally we define the *cone* from a point x to a simplex τ as the union of line segments connecting x to points $p \in \tau$:

$$x \cdot \tau \quad = \quad \mathrm{conv}\,(\tau \cup \{x\}).$$

It is defined only if x is not an affine combination of the vertices of τ. With this restriction, $x \cdot \tau$ is a simplex of one higher dimension: $\dim(x \cdot \tau) = 1 + \dim \tau$. For a *set* of simplices the cone is defined if it is defined for each simplex, and in this case $x \cdot T = \{x \cdot \tau \mid \tau \in T\}$. We also need generalizations of the star and the link from a single simplex to a set of simplices. Denote the closure without the (-1)-simplex as $\overline{T} = \mathrm{Cl}\,T - \{\emptyset\}$. The *star* and *link* of T are

$$\begin{aligned} \mathrm{St}\,T &= \{\sigma \in K \mid \sigma \geq \tau \in T\}, \\ \mathrm{Lk}\,T &= \mathrm{Cl}\,\mathrm{St}\,T - \mathrm{St}\,\overline{T}. \end{aligned}$$

For closed sets T the link is simply the boundary of the closed star. For example in Figure 1 the link of the set $\overline{ab} = \{ab, a, b\}$ is the cycle of dashed edges and hollow vertices bounding the closed star of $\overline{ab}$. The *contraction* of the edge ab is the operation that changes K to

$$L \quad = \quad K - \mathrm{St}\,\overline{ab} \;\cup\; c \cdot \mathrm{Lk}\,\overline{ab}.$$

This definition applies generally and does not assume that K is a manifold.

Decimation. The surface represented by K is simplified by performing a sequence of edge contractions. To get a meaningful result we prioritize the contractions by the numerical error they introduce. Contractions that change the topological type of the surface are rejected. Initially, all edges are evaluated and stored in a priority queue. The process continues until the number of vertices shrinks to the target number m. Let $n \geq m$ be the number of vertices in K.

```
while n > m and priority queue non-empty do
  extract top edge ab from priority queue;
  if contracting ab preserves topology then
    contract ab; n--
  endif
endwhile.
```

The priority queue takes time $O(\log n)$ per operation. Besides extracting the edge whose contraction causes the minimum error we remove edges that no longer belong to the surface and we add new edges. The number of edges removed and added during a single contraction is usually bounded by a small constant, but in the worst case it can be as large as $n - 1$. Before performing an edge contraction we test whether or not it preserves the topological type of the surface. This is done by checking all edges and vertices in the link of $\overline{ab}$. Precise conditions to recognize edge contractions that preserve the type will be discussed in the next lecture.

Hierarchy. We visualize the actions of the algorithm by drawing the vertices as the nodes of an upside-down

forest. The contraction of the edge ab combines vertices a and b into a new vertex c. In the forest this is reflected by introducing c as a new node and declaring it the parent of a and b. The leaves of the forest are the vertices of K, and the roots are the vertices of the decimated complex L, see Figure 2. We define a function

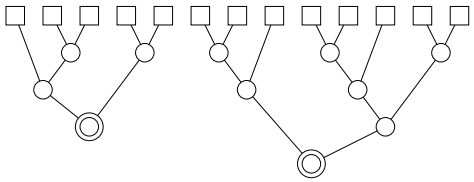

Figure 2: Vertices of K are shown as square nodes, intermediate vertices as circle nodes, and vertices of the final complex L as double circle nodes.

$g : \mathrm{Vert}\, K \to \mathrm{Vert}\, L$ that maps each vertex $u \in K$ to the root $g(u)$ of the tree in which u is a leaf. The preimage of a vertex $v \in L$ is the set of leaves $g^{-1}(v) \subseteq K$ of the tree with root v. The preimages of the roots partition the set of leaves:

$$\mathrm{Vert}\, K \quad = \quad \bigcup_{v \in L} g^{-1}(v),$$

where the union is over a collection of pairwise disjoint sets. Later, we will extend function g from vertices to edges and triangles. This will be useful in the study of structural connections between the surfaces K and L.

Numerical error. As mentioned above, a vertex $v \in \mathrm{Vert}\, L$ represents a subset $g^{-1}(v) \subseteq \mathrm{Vert}\, K$ of the vertices in K. It makes sense to measure the numerical error at v by comparing v to the part of the original surface it represents. Specifically, we define the error at v as the sum of square distances of v from the planes spanned by triangles in the star of $g^{-1}(v)$. See Figure 3, which shows a vertex $v \in L$ and the triangles in the star of $g^{-1}(v)$. The preimage of v is the collection of seven solid vertices in the right half of the figure. The star of the preimage contains the five shaded triangles and the ring of white triangles around them. The shaded triangles have all their vertices in $g^{-1}(v)$ and the white triangles have either one or two vertices in the preimage.

Let H_v be the set of planes spanned by triangles in $\mathrm{St}\, g^{-1}(v)$. The sum of square distances is defined for every point $x \in \mathbb{R}^3$, so we can think of the error measure as a function $E_v : \mathbb{R}^3 \to \mathbb{R}$. Specific properties of this function will be discussed in the third lecture after this one. For now we just observe that the error function

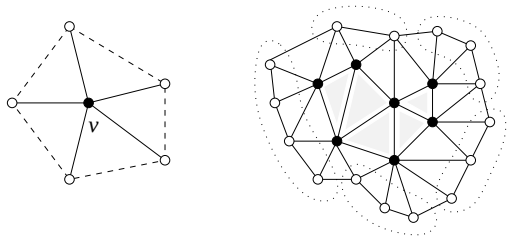

Figure 3: Vertex v and its star to the left and the corresponding piece of K to the right. The solid vertices on the right are preimages of v and the hollow vertices are preimages of the neighbors of v.

for the union of two sets of planes can be computed by inclusion-exclusion. Specifically, if $H_c = H_a \cup H_b$ then

$$E_c(x) \quad = \quad E_a(x) + E_b(x) - E_{ab}(x),$$

where E_{ab} is the error function defined by the intersection, $H_{ab} = H_a \cap H_b$. This formula together with a compact representation of error functions will be the mechanism we use to compute the error at a new vertex c. Given a set of planes there is generally a unique point that minimizes the corresponding error function. Instead of computing c directly from a and b we first construct $H_c = H_a \cup H_b$ and second choose c at the minimum of the error function E_c defined by H_c.

Bibliographic notes. The idea of using edge contractions for surface simplification appears first in Hoppe et al. [3]. They select contractions together with other local surface modification operations in an attempt to optimize a measure of distance between the original and the decimated surface. Hoppe [2] revisits the idea and shows how to use a given sequence of contractions for efficiently switching back and forth between representations on different levels of detail. The algorithm in these notes selects contractions greedily using the quadratic error measure as suggested by Garland and Heckbert [1].

[1] M. Garland and P. S. Heckbert. Surface simplification using quadratic error metrics. *Comput. Graphics*, Proc. siggraph 1997, 209–216.

[2] H. Hoppe. Progressive meshes. *Comput. Graphics*, Proc. siggraph 1996, 99–108.

[3] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stützle. Mesh optimization. *Comput. Graphics*, Proc. siggraph 1993, 19–26.