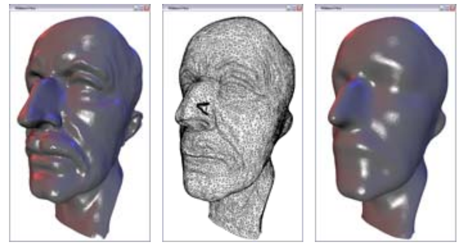
Working with Meshes

Surfaces/Meshes

We'll stick to triangles



Discrete Surfaces

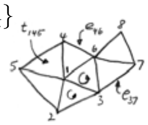
Setup

"pointers" "floats"

- topology & geometry
- simplicial complex: "triangle mesh"
  - 2-manifold $K = \{V, E, T\}$

    $V = \{v_i\}$ $E = \{e_{ij}\}$ $T = \{t_{ijk}\}$
  - Euler characteristic

    $F - E + V = 2(1-g) = \chi$



What's a Mesh?

Formally

- abstract simplicial complex K
  - singletons, pairs, triples,... of integers

abstract simplices

$V = \{1, 2, 3, \ldots\}$ $E = \{\{i, j\}, \{k, l\}, \ldots\}$

$F = \{\{i, j, k\}, \{j, i, l\}, \ldots\}$

- containment property

  $\rho \in K \wedge \sigma \subset \rho \Rightarrow \sigma \in K$
- partial order $\preceq$, face, coface, $\emptyset$

Simplicial Complex

Topological realization

- identify V with unit vectors in $\mathbb{R}^N$

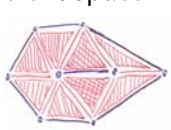
  $|K| = \bigcup_{\sigma \in K} |\sigma|$ ← convex hull of vertex images
- subset topology of ambient space
- closure, star, and link

make subcomplex

incidence

$\mathrm{Cl}\, L = \{\rho \mid \rho \preceq \sigma, \sigma \in L\}$

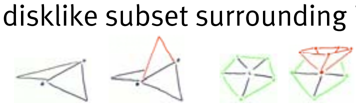
$\mathrm{St}\, L = \{\rho \mid \sigma \preceq \rho, \sigma \in L\}$



Topological Structure

2-manifold (with boundary)

- every point has an open, (half-) disklike subset surrounding it



- |K| 2-manifold iff |St v| ≈ $\mathbb{R}^2$

  $|\mathrm{St}\,\sigma| = \bigcup_{\rho \in \mathrm{St}\,\sigma} \mathrm{int}\, |\rho|$

Topological Invariants

Euler characteristic

- for surfaces: F-E+V=χ=2(g-1)
  - not required to be simplicial
- more generally for simplicial complexes

$$\chi(K) = \sum_{\emptyset \neq \rho \in K} (-1)^{\dim \rho}$$

  - proof by induction (shelling)

Simplicial Complex

Geometric realization

- the concrete embedding $\pi_v(|K|)$

$$\pi_v : \mathbb{R}^n \to \mathbb{R}^3$$

  - vertex images specify everything
  - piecewise linear approximation
  - presumably approximation of underlying smooth surface

Mesh Structure

Input

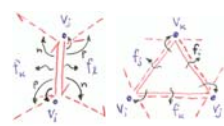
```
x1 y1 z1 …
x2 y2 z2 …
x3 y3 z3 …
…
i j k
j k l
…
```

- typically
  - list of vertices (how long?)
  - list of triangles (until EOF)
- need to build mesh structure
  - infer topology
  - check topology
  - oriented (orientable?)

Building the Mesh

What do we need?

- array of pointers to vertices
- choices for basic topology primitive
  - (half-)edges
    - different variants
  - triangles
- we'll use triangles



Types of Operations

What do we need to support?

- iterate over all vertices (easy)
- iterate over all triangles (easy)
- for a triangle visit
  - incident vertices (easy)
  - incident triangles (easy)

Types of Operations

What do we need to support?

- for a vertex visit
  - star $\forall v_i : \{t_{ijk}\} \subset T$
  - link $\forall v_i : \{e_{jk} | t_{ijk} \in T\}$
  - different flavors $\forall v_i : \{v_j | e_{ij} \in E\}$
- need back pointer
  - vertex points to one incident triangle
  - careful at boundary!

Types of Operations

What about edges?

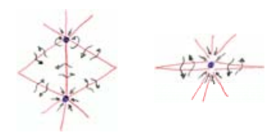
- visit all edges
  - not explicitly represented…
- do we need edges? Yes!
  - discover triangle adjacencies
  - map pairs of integers to triangles

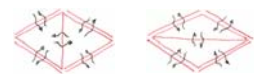
$$e_{ij} \mapsto \{t_{ijk}, t_{jil}\}$$

Operations to Support

For later (think about it now…)

- edge collapse
  - legality?
- edge flip



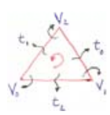


Data Structures

Triangles

- consistent ordering of vertex and triangle incidences

```
Triangle{
  Vertex *v[3];
  Triangle *t[3];
}
```



  - triangles across from vertices

What Data Where?

Attributes

- normal, color, texture coordinates
  - later: forces, velocities, mass
- why not just lay everything out in arrays?
  - changes in structure!
  - very hard to debug…

Examples

Vertex normals

- gradient of volume

$$n_i = 1/2 \sum_{t_{ijk}} (p_j - p_i) \times (p_k - p_i) \qquad \boxed{N_i = \frac{n_i}{|n_i|}}$$

$$\forall v_i : n_i = \vec{0} \qquad \forall t_{ijk} : a_{ijk} = (p_j - p_i) \times (p_k - p_i)$$

$$\forall t_{ijk} : \begin{cases} n_i += a_{ijk} \\ n_j += a_{ijk} \\ n_k += a_{ijk} \end{cases} \qquad \forall v_i : N_i = \frac{n_i}{|n_i|}$$

Example

Gaussian curvature

$$\forall v_i : K_i = 2\pi - \sum_{t_{ijk}} \alpha^i_{jk}$$

$$\forall v_i \in V \setminus \partial V : K_i = 2\pi$$

$$\forall v_i \in \partial V : K_i = \pi$$

$$\forall t_{ijk} : \begin{cases} K_i -= \text{atan2}(|a_{ijk}|, (p_j - p_i) \cdot (p_k - p_i)) \\ K_j -= \text{atan2}(|a_{ijk}|, (p_k - p_j) \cdot (p_i - p_j)) \\ K_k -= \text{atan2}(|a_{ijk}|, (p_i - p_k) \cdot (p_j - p_k)) \end{cases}$$

Principles

As you write code...

- assumptions are ok, but you must assert them explicitly
  - orientability
  - 2-manifold property
- avoid storing the same information multiple times
  - nasty to keep current under changes

Other Tricks

As you write code

- use two sided lighting
- abstract the iterators!
  - what about boundary vertices?
- keep iterators sorted
  - interior then boundary vertices
  - interior then boundary triangles