

---

# WORKING WITH MESHES

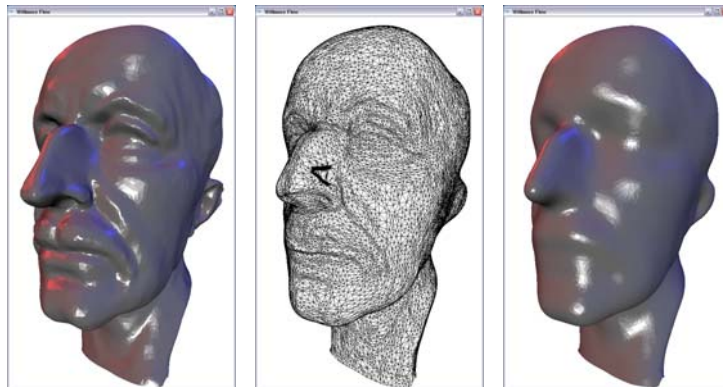
CSE 872 FALL 2009

1

## SURFACES / MESHES

---

We'll stick to triangles



CSE 872 FALL 2009

2

# DISCRETE SURFACES

Setup

“pointers”

“floats”

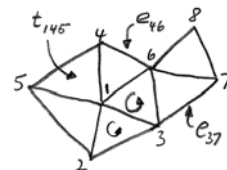
- topology & geometry
- simplicial complex: “triangle mesh”
- 2-manifold

$$K = \{V, E, T\}$$

$$V = \{v_i\} \quad E = \{e_{ij}\} \quad T = \{t_{ijk}\}$$

- 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

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

- $F = \{\{i, j, k\}, \{j, i, l\}, \dots\}$

- $\text{part } \sigma \in K \wedge \sigma \subseteq \rho \Rightarrow \sigma \in \text{Kface}, \emptyset$

# SIMPLICIAL COMPLEX

## Topological realization

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

$$|K| = \bigcup_{\sigma \in K} |\sigma|$$

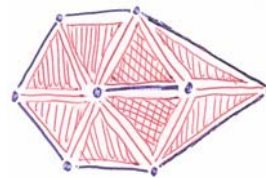
convex hull of vertex images

- subset topology of ambient space
- closure, star, and link

incidence subcomplex

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

$$StL = \{\rho \mid \sigma \preceq \rho, \sigma \in L\} \setminus (L - \emptyset)$$



# TOPOLOGICAL STRUCTURE

## 2-manifold (with boundary)

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



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

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

# TOPOLOGICAL INVARIANTS

---

## Euler characteristic

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

- not required to be simplicial

- more generally for simplicial complexes

- proof by induction (shelling)

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

# SIMPLICIAL COMPLEX

---

## Geometric realization

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

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

- vertex images specify everything

- piecewise linear approximation

- presumably approximation of underlying smooth surface

# MESH STRUCTURE

---

## Input

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

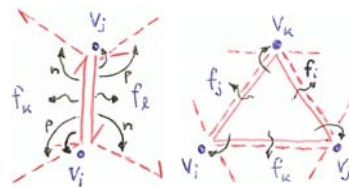
```
x1 y1 z1 ...  
x2 y2 z2 ...  
x3 y3 z3 ...  
...  
i j k  
j k l  
...
```

# BUILDING THE MESH

---

## What do we need?

- array of **pointers** to vertices
- choices for basic topology primitive
  - (half-)edges
  - 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

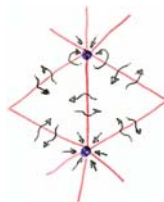
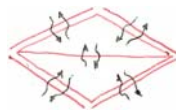
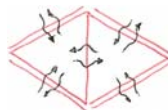
$$e_{ij} \mapsto \{t_{ijk}, t_{jil}\}$$

## OPERATIONS TO SUPPORT

---

For later (think about it now...)

- edge collapse
  - legality?
- edge flip



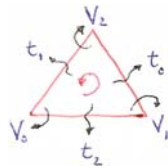
# DATA STRUCTURES

---

## 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?
  - OK, but ...
  - changes in structure!
  - very hard to debug...

## EXAMPLES

### Vertex normals

■ gradient of volume

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

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

$$\forall t_{ijk} : \begin{cases} n_{i+} = a_{ijk} \\ n_{j+} = a_{ijk} \\ n_{k+} = a_{ijk} \end{cases} \quad \forall v_i : N_i = \frac{n_i}{|n_i|}$$

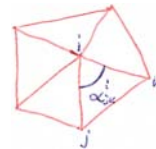
## EXAMPLE

### Gaussian curvature

$$\forall v_i : K_i = 2\pi - \sum_{t_{ijk}} \alpha_{ijk}^i$$

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

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



$$\forall t_{ijk} : \begin{cases} K_{i-} = \text{atan2}(|a_{ijk}|, (p_j - p_i) \cdot (p_k - p_i)) \\ K_{j-} = \text{atan2}(|a_{ijk}|, (p_k - p_j) \cdot (p_i - p_j)) \\ K_{k-} = \text{atan2}(|a_{ijk}|, (p_i - p_k) \cdot (p_j - p_k)) \end{cases}$$

## 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