COS 426 : Precept 4
Half-Edge
Agenda

- Assignment 2 description
- Half-edge data structure
  - Traversal
  - Modification
Assignment 2

• Part 1 - Analysis
  • Implement traversal operations
  • Calculate mesh properties
    • Vertex normal, avg. edge length, etc.

• Part 2 - Filters
  • Filters and Warps similar to assignment 1
  • Topological modifiers
Meshes

- Images had implicit adjacency information
  - Grid around a pixel
  - Easy to express operations
- What about meshes?
  - How to apply smoothing?
Meshes

- Meshes can be quite dense
Meshes

- How to access adjacency information quickly?

One-Ring Neighborhood
## Half-Edge Data Structure

<table>
<thead>
<tr>
<th>Half Edge</th>
<th>Vertex</th>
<th>Face</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex</td>
<td>Position</td>
<td>Half-Edge</td>
</tr>
<tr>
<td>Opposite Half-Edge</td>
<td>Outgoing Half-Edge</td>
<td>...</td>
</tr>
<tr>
<td>Face</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Next Half-Edge</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Half-Edge Data Structure

- Half-Edge
- Vertex
- Opposite Half-Edge
- Face
- Next Half-Edge
Half-Edge Data Structure

One of the two outgoing edges will be used
Half-Edge Data Structure

One of the three edges will be used
Half-Edge Visualization
Traversal

- How to get one-ring neighbors?
Traversals

• How to get one-ring neighbors?

```cpp
original_he = vertex.he;
he = original_he;
do {
    // do something with data
    he = he.opposite.next;
} while (he != original_he)
```

• Assignment will ask you for other kind of adjacency queries

  • Vertices around Face, Faces around Vertex etc.
Traversals

- Vertex Normals are defined as weighted average of adjacent faces (weighted by face area).

- How would you compute vertex normals given per face normal and area?
Traversal

- Vertex Normals are defined as weighted average of adjacent faces (weighted by face area)

```cpp
original_he = vertex.he;
he = original_he;
do {
    // do something with data
    he = he.opposite.next;
} while (he != original_he)
```
Traversal

- Vertex Normals are defined as weighted average of adjacent faces (weighted by face area)

- (Can also be done by using facesOnVertex)

```c
original_he = vertex.he;
he = original_he;
v_normal.set( 0, 0, 0 );
do {
    f_normal = he.face.normal;
    area = he.face.normal.area;
    v_normal.add(f_normal*area);
    he = he.opposite.next;
} while ( he != original_he)
v_normal.normalize();
```
Traversal

- Similarly, in uniform Laplacian smoothing each vertex is moved towards the average of it and its neighbors.

```plaintext
original_he = vertex.he;
he = original_he;
do {
    // do something with data
    he = he.opposite.next;
} while (he != original_he)
```
Traversals

- Similarly, in uniform Laplacian smoothing each vertex is moved towards the average of it and its neighbors.

```plaintext
original_he = vertex.he;
he = original_he;
avg_pos.set( 0, 0, 0 );
do {
    avg_pos.add(he.vertex);
    he = he.opposite.next;
} while ( he != original_he)
avg_pos.add(-vertex*num_neigh);
new_pos = vertex + avg_pos*delta;
```
Traversal

- Cotan Laplacian smoothing

$$w = \frac{\cot(\alpha_{ij}) + \cot(\beta_{ij})}{2}$$

avg_pos.add(he.vertex); → avg_pos.add(w*he.vertex);
num_neigh → total_w
splitEdgeMakeVert(v1, v2, factor)

v3 = addVertex(weightedAvgPos(v1, v2, factor));

he1.vertex = v3;
he2.vertex = v3;

he3 = addHalfEdge(v3, v2, f1);
he4 = addHalfEdge(v3, v1, f2);

he1.next = he3;
he2.next = he4;

he3.next = he1_next;
he4.next = he2_next;

he1.opposite = he4;
he4.opposite = he1;
he2.opposite = he3;
he3.opposite = he2;
Data Structure Modification

- `splitFaceMakeEdge( f, v1, v2, vertOnF, switchFaces )`

```plaintext
f2 = addFace();
he5 = addHalfEdge( v1, v2, f1 );
he6 = addHalfEdge( v2, v1, f2 );
he5.opposite = he6;
he6.opposite = he5;
he5.next = he2;
he3.next = he5;
he1.next = he6;
he6.next = he4;

f1.halfedge = he5;
f2.halfedge = he6;

he5.next = he2;
he3.next = he5;
he1.next = he6;
he6.next = he4;

Remember to re-link he4 and he1 to point to f2
```
Data Structure Modification

• How would you go about subdividing a quad face?
  • You’re given split edge and split face
  • Just use those - guaranteed validity of dataset after use!

• Part of the assignment
  • Think about it for next week!