COS 426: Precept 4

Introduction to Half-Edges

Andy Zeng
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 (access in O(1) time)
  • 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

<table>
<thead>
<tr>
<th>Vertex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
</tr>
<tr>
<td>Outgoing Half-Edge</td>
</tr>
</tbody>
</table>

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

One of the three edges will be used
Half-Edge Visualization
Exercise: vertex traversal

- How to get one-ring neighbors?
Traversal

- How to get one-ring neighbors?
Traversals

- How to get one-ring neighbors?

```java
original_he = vertex.he;
current = original_he;
do {
    // do something with data
    current = 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?

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

- splitEdgeMakeVert (v1, v2, factor)

\[
v_3 = \text{addVertex} \left( \text{weightedAvgPos}(v_1, v_2, \text{factor}) \right);
\]

\[
\begin{align*}
&\text{he1.} \text{vertex} = v_3; \\
&\text{he2.} \text{vertex} = v_3; \\
&\text{he3} = \text{addHalfEdge}(v_3, v_2, f_1); \\
&\text{he4} = \text{addHalfEdge}(v_3, v_1, f_2); \\
&\text{he1.} \text{next} = \text{he3}; \\
&\text{he2.} \text{next} = \text{he4}; \\
&\text{he3.} \text{next} = \text{he1.} \text{next}; \\
&\text{he4.} \text{next} = \text{he2.} \text{next}; \\
&\text{he1.} \text{opposite} = \text{he4}; \\
&\text{he4.} \text{opposite} = \text{he1}; \\
&\text{he2.} \text{opposite} = \text{he3}; \\
&\text{he3.} \text{opposite} = \text{he2};
\end{align*}
\]
Data Structure Modification

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

```
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;
```

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 during tomorrow’s class!