Considerations toward a Dynamic Mesh Data Structure

Sebastian Pena Serna, Andre Stork, Dierter W. Fellner
Motivation

- 3D shapes is essential for representing 3D physical reality in different domains
- Meshes are a versatile and common representation for the 3D reality
  - The mesh generation process does not always produce quality results
  - The domain application demands special requirements
  - The processes applied on the meshes need to support operations

Objective: analyze considerations toward designing a data structure for dynamic meshes in a generic and robust manner
Introduction

- Dynamic mesh:
  - Dynamic changes in the geometry of the mesh
  - Dynamic changes in the topology of the mesh

- Considerations:
  - Memory and Performance
  - Neighboring Information
  - Mesh Modifications
Memory and Performance

- Data structures for static meshes compactly encode the topology
- These achieve minimal memory consumption and maximal performance

- Data structures for dynamic meshes cannot encode the topology
- The memory consumption is difficult to optimize
- A mechanism to rapidly update the neighboring information and to increase or decrease the number of entities is needed
  - Memory buffers
Memory and Performance

Euler Formula for 2-manifold with genus 0
\[ F + V - E = 2 \]
\[ E = F + V - 2 \]

For 3-manifold with genus 0
\[ V + F - E = C + 1 \]
\[ E \approx (2 \times V) + C \]
\[ F \approx V + (2 \times C) \]
## Memory and Performance

<table>
<thead>
<tr>
<th>Mesh</th>
<th>$V$</th>
<th>$E$</th>
<th>$F$</th>
<th>$C$</th>
<th>$V + F - E$</th>
<th>$E \approx (2 \times V) + C$</th>
<th>$F \approx V + (2 \times C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tetrahedron</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>Sphere</td>
<td>13</td>
<td>42</td>
<td>50</td>
<td>20</td>
<td>21</td>
<td>46</td>
<td>53</td>
</tr>
<tr>
<td>Gargoyle</td>
<td>221039</td>
<td>1152799</td>
<td>1723886</td>
<td>792125</td>
<td>792126</td>
<td>1234203</td>
<td>1805289</td>
</tr>
<tr>
<td>Hexahedron</td>
<td>8</td>
<td>12</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>17</td>
<td>10</td>
</tr>
<tr>
<td>Cylinder</td>
<td>380</td>
<td>1004</td>
<td>885</td>
<td>260</td>
<td>261</td>
<td>1020</td>
<td>900</td>
</tr>
<tr>
<td>Rubber</td>
<td>9367</td>
<td>24354</td>
<td>20812</td>
<td>5824</td>
<td>5825</td>
<td>24558</td>
<td>21015</td>
</tr>
</tbody>
</table>

![Graphs showing data](image)
Neighboring Information

- The initialization builds relationships between topological entities
- Hierarchical decomposition: Cell, Face, Edge, and Vertex
- n-dimensional entity is decomposed into its n-1-dimensional entities
- The hierarchical decomposition follows always the right-hand rule
# Neighboring Information

## Topological Templates

<table>
<thead>
<tr>
<th>$F_i$</th>
<th>$F_i$</th>
<th>$F_{i+1}$</th>
<th>$F_{i+2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_i, V_{i+1}, V_{i+3}$</td>
<td>$V_{i}-V_{i+1} \rightarrow C_{e_i}$</td>
<td>$V_{i+1}-V_{i+3} \rightarrow C_{e_{i+3}}$</td>
<td>$V_{i+3}-V_i \rightarrow C_{e_{i+4}}$</td>
</tr>
<tr>
<td>$V_{i+1}, V_i, V_{i+2}$</td>
<td>$V_{i+1}-V_i$</td>
<td>$V_i-V_{i+2}$</td>
<td>$V_{i+2}-V_{i+1}$</td>
</tr>
<tr>
<td>$V_{i+2}, V_{i+3}, V_{i+1}$</td>
<td>$V_{i+2}-V_{i+3} \rightarrow C_{e_{i+1}}$</td>
<td>$V_{i+3}-V_{i+1}$</td>
<td>$V_{i+1}-V_{i+2} \rightarrow C_{e_{i+2}}$</td>
</tr>
<tr>
<td>$V_{i+3}, V_{i+2}, V_i$</td>
<td>$V_{i+3}-V_{i+2}$</td>
<td>$V_{i+2}-V_{i} \rightarrow C_{e_{i+5}}$</td>
<td>$V_i-V_{i+3}$</td>
</tr>
</tbody>
</table>
Neighboring Information

- External query functions:
  - \( Vts(E_i), Eds(E_i), Star(e_i), Star(V_i) \)

- Internal query functions:
  - \( Eds(V_i), Fcs(e_i), Vts(e_i) \)

- Pre-computed or computed on demand:
  - memory consumption
  - querying performance
  - updating performance

- Topological templates enable both alternatives
Mesh Modifications

- Dynamic meshes deal with changes in the geometry and the topology of the mesh.
- Geometry remains constant and the topology is changed.
- Geometry is changed, invoking modification in the topology.
- Typical actions on the mesh are called topological operations.
  - Correct degeneracies on the mesh.
  - Improve the quality of the mesh.
Mesh Modifications

Topological Operations:
- edge-split
- edge-collapse
- edge-swap
Mesh Modifications

- Abstraction from the application
  - Move(Vi)
  - Add(Vi)
  - Kill(Vi)
  - Add(Ei)
  - Kill(Ei)
## Implementations

### Basic Operations
- Scale
- Mirror
- Decimate
Implementations

- Topological Operations
  - edge-split
  - edge-collapse
Implementations

- Decreasing and increasing the radius of features
Implementations

- Dragging holes (semantic features)
Implementations

- Analysis of Design Variations
Conclusions

■ Several mesh data structures for triangular meshes and some for tetrahedral meshes

■ Quadrangular and hexahedral data structures are very limited

■ Many data structures are designed for minimizing the memory consumption for specific domain applications

■ There are not enough data structures, which robustly represent 3D shapes and for supporting dynamic meshes

■ Considerations in terms of memory and performance, neighboring information, and mesh modifications

■ We will investigate the most appropriate trade-off between memory and performance
Thank You!

Sebastian Pena Serna  
Fraunhofer-Institut für Graphische Datenverarbeitung  
IGD  
Fraunhoferstraße 5  
64283 Darmstadt

Tel +49 6151 155 – 468  
sebastian.pena.serna@igd.fraunhofer.de  
www.igd.fraunhofer.de