Simulation of Instrument-Tissue Interactions and System Integration

Cagatay Basdogan, Ph.D.
Jet Propulsion Laboratory
California Institute of Technology
Topics:

A) Collision detection and computational models of surgical instruments

B) Physically-based modeling for simulating soft tissue behavior

C) Haptic rendering of deformable objects

D) Software and hardware integration
Case Study:

Simulation of Catheter Insertion into the Cystic Duct

What you see ...  

What is really happening ...
A) Collision detection and computational models of surgical instruments

- Principles of collision detection
- How to check collisions faster?
- Computational models of laparoscopic instruments
Principles of Collision Detection

(object-object is too expensive !)

Point-Object

(e.g. catheter - cystic duct)

(\text{Line Segment} - \text{Object})

(e.g. forceps - cystic duct)
How to check collisions faster?
(Ref: “Graphics Gems, I-Collide, V-Collide, V-Clip”)

1) Bounding-Boxes

2) Hierarchical Database for Organ Geometry

- Polygon
  - Neighbors
  - Edge
  - Vertex
- Edge
- Vertex
Computational Models of Laparoscopic Instruments

Group A

Group B
B) Physically-based modeling for simulating soft tissue behavior

- Desired properties of deformable models
- Modeling of deformable objects
  1) particle-based
  2) FEM-based
- Implementing constraints
- Problems with particle-based techniques
- Problems with FEM techniques
Desired properties of force-reflecting deformable models

• reflect stable forces

• display smooth deformations

• handle various boundary conditions and constraints

• display “physically-based” behavior in real-time
**Modeling of Deformable Objects**

see my web-site for the details

---

**Particle-based:**

\[ F = ma \]

\[ \begin{align*}
F_{\text{spring}} & = \ldots \\
F_{\text{damping}} & = \ldots \\
F_{\text{gravity}} & = \ldots \\
\end{align*} \]

\[ a(t + \Delta t) = \frac{F}{m} \]

\[ v(t + \Delta t) = v(t) + \Delta t \ a(t + \Delta t) \]

\[ p(t + \Delta t) = p(t) + \Delta t \ v(t + \Delta t) \]

---

**FEM-based:**

\[ F = KU \] (static analysis)

\[ \ldots \]

\[ F = MU + CU + KU \] (dynamic analysis)
Comparison

Particle-based: easy to implement, flexible

FEM-based: comprehensive
Constraints
see my web-site for the details

Examples:
• a node is fixed in 3D space
• a node is constrained to stay on a path
• curvature constraint
• constant volume

Implementation:

1) Particle-based models (Ref: Witkin/Baraff, SIGGRAPH Notes)
   a) Penalty
   b) Lagrange multipliers
2) FEM
## Problems with Particle-Based Techniques

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A)</strong> Adding damping to stabilize oscillations</td>
<td><strong>B)</strong> Adding constraints</td>
</tr>
<tr>
<td><strong>C)</strong> Too many elements</td>
<td><strong>D)</strong> Too few elements</td>
</tr>
<tr>
<td><strong>E)</strong> Non-homogeneous distribution of elements</td>
<td><strong>F)</strong> finer adjustment of spring and damper coefficients</td>
</tr>
</tbody>
</table>

![Diagram](image)
## Problems with FEM Techniques

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>requires</th>
</tr>
</thead>
<tbody>
<tr>
<td>A)</td>
<td>Change in topology</td>
<td>Re-meshing</td>
</tr>
<tr>
<td>B)</td>
<td>Dynamic analysis</td>
<td>Simplifications in the model</td>
</tr>
<tr>
<td>C)</td>
<td>Matrix inversion</td>
<td>Pre-computation</td>
</tr>
<tr>
<td>D)</td>
<td>Memory allocation</td>
<td>Simplifications in the geometry</td>
</tr>
</tbody>
</table>
C) Haptic Rendering of Deformable Objects

• Principles of haptic rendering
• Key components of a haptic rendering algorithm
• Computational Architecture
Principles of Haptic Rendering
see my web-site for the details

\[ F = k \Delta x \]
Key Components of a Haptic Rendering Algorithm

1) Contact history

2) Local coherence
Physics-Based Deformable Model

Collision Detection

Geometry

Position Orientation

Displacement Force

Computational Architecture
D) Software and Hardware Integration: tips and tricks

• Programming tips to speed up your computations
• Modeling tips to speed up your computations
• Simulation set-up
Programming tips to speed up your computations

• Synchronize your haptic and graphic loops through a shared database

```
| Haptic Database | Shared Database | Visual Thread |
```

• Construct a multi-layered computing structure

```
Thread #1: Compute Forces/Disp. 100 Hz → Extrapolate Forces/Disp.
Thread #2: Display Forces 1 kHz → Display Images 30 Hz
Thread #3: Extrapolate Forces/Disp. → Compute Forces/Disp. 100 Hz
```

• Construct a hierarchical data structure

• Update your geometric coordinates less frequently

\[
\Delta t_{\text{haptic}} = 0.001 \text{ sec (display forces)} \\
\Delta t_{\text{iteration}} = 0.01 \text{ sec (update coordinates)}
\]
Modeling tips to speed up your computations

• deform your objects locally
• take advantage of single point interactions
• condense your matrices in FEM
• transform your coordinates to modal coordinates (for dynamic analysis)
• pre-compute (matrices, unit displacements/force)
• loosely couple your force and deformation model
• adaptive meshing
• take advantage of human perceptual limitations
Simulation of Catheter Insertion

Real

Simulation Set-Up
Acknowledgements

Chih-Hao Ho
Mandayam Srinivasan
Mark Ottensmeyer
Ela Ben-Ur
Jim Westwood

tutorial notes will be available online:

http://eis.jpl.nasa.gov/~basdogan