Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Overview

- Last Week’s Question
- Elastic Collision Detection
- Collision Detection for Reduced Models
- Surface-Based Elastics
- New Question
Question

• How could we reduce the cost of simulation for a very finely discretized surface?

• Are there cheap ways of getting volumetric behavior without a full tetrahedralization?

• How can collision constraints be integrated?

• How to simulate plasticity?
Solutions

- bounding volume tree w/ tetrahedra at leaves
  - simulate parent nodes instead of leaves (if stresses are close)
- simulate on a simplified mesh (make details into bump maps)
- adaptive tetrahedralization based on force magnitudes
- come up with tetrahedralization that best captures the simulation based on precomputed simulations
- springs connected to a “skeleton”
- plasticity based on sparse springs connecting the surface mesh to itself
- embed fine tetrahedral mesh as barycentric coordinates on a coarse tetrahedral mesh, solve on coarse mesh
- angular springs in a surface discretization of the dynamics
- nonuniform tetrahedral mesh based on the curvature of the surface mesh
- greater distance to the surface -- the larger the tetrahedron
- “shell” tetrahedralization with springs on the interior
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Collision Detection

- **Broad Phase:**
  - Guess collisions between objects.

- **Narrow Phase:**
  - Determine collision points.
Broad Phase
Fast Interval Operations

- Temporal coherency: keep list between timesteps.
- Use insertion sort. **Expected O(n) runtime.**
- Update overlaps *during* insertion sort.
- Three cases:
  - A minimum and a maximum flip. **Toggle overlap bit.**
  - Two minima flip. **Don’t toggle.**
  - Two maxima flip. **Don’t toggle.**
Narrow Phase

- Find exact collision point.
- Use a geometric partitioning algorithm.
- Two types:
  - Bounding Volume Hierarchies
  - Spatial Partitioning
BVH vs. Spatial Partitioning

**BVH:**
- Object centric
- Spatial redundancy

**SP:**
- Space centric
- Object redundancy

(From Doug James’s Slides.)
BVH vs. Spatial Partitioning

**BVH:**
- Object centric
- Spatial redundancy

**SP:**
- Space centric
- Object redundancy

(From Doug James’s Slides.)
BVH vs. Spatial Partitioning

**BVH:**
- Object centric
- Spatial redundancy

**SP:**
- Space centric
- Object redundancy

(From Doug James’s Slides.)
BVH vs. Spatial Partitioning

**BVH:**
- Object centric
- Spatial redundancy

**SP:**
- Space centric
- Object redundancy

(From Doug James’s Slides.)
Bounding Volume Hierarchies

- How to create a BVH:
  - Geometric Subdivision
  - Topological Subdivision
    - How implement?
    - Which is better?

- How to update a BVH:
  - Bottom Up (How?)
  - Directly (How?)
  - Which is faster?
Triangle Intersection

- Edge-Edge
- Vertex-Face
Summary

• Broad Phase:
  • Guess collisions between objects.

• Narrow Phase:
  • Determine collision points.
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Collision Detection for Reduced Models

Figure 2: Example deformation: (a) Reference shape \( p \) (b) Displacement field \( U_{*1} \) (c) Field \( U_{*2} \) (d) Deformed shape \( p' \).

\[
p' = p + Uq \quad \text{or} \quad p'_i = p_i + \sum_{j=1}^{M} U_{ij}q_j.
\]
et al. 2002; Brown et al. 2001; Larsson and Akenine-Matthews 2001; Brown et al. 2001; Guibas et al. 2002; Bradshaw and O'Sullivan 2004. Collision detection with bounding volume hierarchies is updating the hierarchy after every deformation, e.g., at interactive rates. As a result, much of the recent work on updating using hierarchical linear-time sphere refitting and constructing good bounding sphere hierarchies has been on how to efficiently update a wrapped hierarchy, BD-Trees can obtain tighter bounds on bounce positions than the deformable layered case (see Figure 4).

Note that the reduced coordinates (Eq. 1) our method can be applied in a more general setting (see section 3.2). BD-Trees exploit spatially coherent motion that can be described as a combination of (hopefully smooth) displacement fields. Mathematical methods for updating the bounding hierarchies (e.g., van den Bergen 1997; Ganovelli et al. 2000). Alternately, intelligent methods for updating the bounding hierarchies are used in this context. As a result, much of the recent work on updating using hierarchical linear-time sphere refitting and constructing good bounding sphere hierarchies has been on how to efficiently update a wrapped hierarchy, BD-Trees can obtain tighter bounds on bounce positions than the deformable layered case (see Figure 4).

The construction of BD-Trees has two stages: first, compute bounding spheres for each node in the hierarchy at the root level, and then use a method similar to [Quinlan 1994] for binary sphere tree construction to sequentially update the hierarchy. Without loss of generality, we consider sphere trees constructed on polygonal meshes as hierarchical representations of deformable models henceforth, and briefly discuss how these could be extended to polygonal collision detection with bounding volume hierarchies. Throughout, we will describe the bounding volume hierarchy construction as a wrapped hierarchy (left) and a layered hierarchy (right). The base geometry is shown in green, and the spheres (see Figure 3). Layered hierarchies are important in preprocessing and updating. The construction process is to start with the loose fitting bounding sphere produced during hierarchy construction, with center at the centroid and radius equal to the maximum distance from this center to any vertex of the base geometry. BD-Trees tend to perform. Suppose we have a derived output-sensitive subset of model polygons.

In these methods, and not just visible portions; a notable advantage is that the entire model has to be rendered for collision detection in one frame. Many of these methods could be directly applied to deformable models, and not just visible portions; a notable advantage is that the entire model has to be rendered for collision detection in one frame. In Section 1.1 Related Work many methods are updating the hierarchy after every deformation, e.g., at interactive rates. As a result, much of the recent work on updating using hierarchical linear-time sphere refitting and constructing good bounding sphere hierarchies has been on how to efficiently update a wrapped hierarchy, BD-Trees can obtain tighter bounds on bounce positions than the deformable layered case (see Figure 4).

Recently, hardware-accelerated collision detection methods have received increased attention; see [Manocha et al. 2002] for a recent overview. These general purpose methods are very useful, but ten thirth of collision detection operations are specialized for morphing that has features similar to ours (and is a specialization of deformable models). Hierarchy construction can be based on either spatial proximity between features (e.g., [Hubbard 1995; Dingliana and O'Sullivan 2000]). Collision detection with bounding volume hierarchies is updating the hierarchy after every deformation, e.g., at interactive rates. As a result, much of the recent work on updating using hierarchical linear-time sphere refitting and constructing good bounding sphere hierarchies has been on how to efficiently update a wrapped hierarchy, BD-Trees can obtain tighter bounds on bounce positions than the deformable layered case (see Figure 4).

The construction process is to start with the loose fitting bounding sphere produced during hierarchy construction, with center at the centroid and radius equal to the maximum distance from this center to any vertex of the base geometry. BD-Trees tend to perform. Suppose we have a derived output-sensitive subset of model polygons.
Sphere Center Update

\[
  c' = c + \sum_{i \in \Lambda} \beta_i u_i = c + \sum_{i \in \Lambda} \beta_i \left( \sum_{j=1}^{M} U_{ij} q_j \right)
\]

\[
  = c + \sum_{j=1}^{M} \left( \sum_{i \in \Lambda} \beta_i U_{ij} \right) q_j
\]

\[
  \equiv c + \sum_{j=1}^{M} \bar{U}_j q_j = \bar{c} + \bar{U} \bar{q} \equiv c'
\]
Sphere Center Update

\[
\max_{i \in \Lambda} \|p'_i - c'\|_2 = \max_{i \in \Lambda} \|(p_i - c) + \sum_{j=1}^{M} (U_{ij} - \bar{U}_j)q_j\|_2
\]

\[
\leq \max_{i \in \Lambda} \|p_i - c\|_2 + \sum_{j=1}^{M} \left( \max_{i \in \Lambda} \|U_{ij} - \bar{U}_j\|_2 \right) |q_j|
\]

\[
\equiv R + \sum_{j=1}^{M} \Delta R_j |q_j| = R + \Delta R^T q^{ABS} \equiv R'
\]
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Surface-Based Elastics

\[ A = \sum_{j \in \text{nbr}(i)} (x_j(t) - x_i(t)) (x_j^0 - x_i^0)^T. \]

\[ c_i(t) = \frac{1}{|\text{nbr}(i)|} \sum_{j \in \text{nbr}(i)} \left( R(x_i^0 - x_j^0) + x_j(t) \right). \]

\[ L_i(t) = \frac{c_i(t) - x_i(t)}{h^2}. \]
Volumetric Behavior

\[
B_i(t) = \frac{(|x^0_{ib}|/|x_{ib}(t)| - 1) \cdot x_{ib}(t)}{h^2},
\]

Source: Xiaohan Shi, Kun Zhou, Yijing Tong, Mathieu Desbrun, Hujun Bao, Baining Guo. 
Example-Based Dynamic Skinning in Real Time. ACM TOG (SIGGRAPH 2008)
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• **Surface-Based Elastics**
• New Question
Overview

• Last Week’s Question
• Elastic Collision Detection
• Collision Detection for Reduced Models
• Surface-Based Elastics
• New Question
Questions

• How could we represent a human body on a computer with few dimensions.

• What kind of optical technology could we use to capture a human body?

• How can we convert the captured data into the human body representation.

• In animation, what do you think are the most important aspects of human motion to capture / model?

• Physically / Stylistically?