Computer Animation
Algorithms and Techniques

Collisions & Contact
Collision handling
detection & response

- Particle-plane collision detection
- Polyhedron-polyhedron collision detection
  - overlap of Bounding volumes
  - Vertex inside polyhedron test
    - Concave case
    - Convex case
  - Edge-face intersection test

- Kinematic response
  - Penalty method
  - Impulse force of collision
Collision detection: point-plane

\[ E(p) = ax + by + cz + d = N \cdot p + d \]

- \( E(p) > 0 \)
- \( E(p) = 0 \)
- \( E(p) < 0 \)
Collision detection: time of impact

2 options
Consider collision at next time step
Compute fractional time at which collision actually occurred

Tradeoff: accuracy v. complexity
Collision response: kinematic

\[ v(t_i) - N \cdot v(t_i) \]

\[ v(t_{i+1}) = v(t_i) - N \cdot v(t_i) - k(N \cdot v(t_i)) \]

\[ = v(t_i) - (1 + k)N \cdot v(t_i) \]

\( k \) – damping factor

=1 indicates no energy loss

Negate component of velocity in direction of normal

No forces involved!

Rick Parent
Collision response: damped

Damping factor = 0.8
Collision response – penalty method
Collision response: penalty
Collision detection: polyhedra

Order tests according to computational complexity and power of detection

1. test bounding volumes for overlap
2. test for vertex of one object inside of other object
3. test for edge of one object intersecting face of other object
Collision detection: bounding volumes

Don’t do vertex/edge intersection testing if there’s no chance of an intersection between the polyhedra

Want a simple test to remove easy cases

Tradeoff complexity of test with power to reject non-intersecting polyhedra (goodness of fit of bounding volume)
Bounding Spheres

Compute bounding sphere of vertices
Compute in object space and transform with object

1. Find min/max pair of points in each dimension
2. Use maximally separated pair – use to create initial bounding sphere (midpoint is center)
3. For each vertex adjust sphere to include point
Bounding Boxes

Axis-aligned (AABB): use min/max in each dimension

Oriented (OBB): e.g., use AABB in object space and transform with object. Vertex is inside of OBB iff on inside of 6 planar equations
Bounding Slabs

For better fit bounding polyhedron: use arbitrary (user-specified) collection of bounding plane-pairs

Is a vertex between each pair?

\[ d_2 < N \cdot P < d_1 \]
Convex Hull

Best fit convex polyhedron to concave polyhedron but takes some (one-time) computation

1. Find highest vertex, V1
2. Find remaining vertex that minimizes angle with horizontal plane through point. Call edge L
3. Form plane with this edge and horizontal line perpendicular to L at V1
4. Find remaining vertex that for triangle that minimizes angle with this plane. Add this triangle to convex hull, mark edges as unmatched
5. For each unmatched edge, find remaining vertex that minimizes angle with the plane of the edge’s triangle
Collision detection: polyhedra

1. test bounding volumes for overlap
2. test for vertex of one object inside of other object
3. test for edge of one object intersecting face of other object

Vertex inside a polyhedron

Object penetration without a vertex of one object contained in the other
Collision detection: polyhedra

Intersection = NO
For each vertex, V, of object A
  if (V is inside of B) intersection = YES
For each vertex, V, of object B
  if (V is inside of A) intersection = YES

A vertex is inside a convex polyhedron
if it’s on the ‘inside’ side of all faces

A vertex is inside a concave polyhedron
if a semi-infinite ray from the vertex
intersects an odd number of faces
Collision detection: polyhedra

Edge intersection face test
Finds ALL polyhedral intersections
But is most expensive test

If vertices of edges are on opposite side of plane of face

Calculate intersection of edge with plane

Test vertex for inside face (2D test in plane of face)
Collision detection: swept volume

Time & relative direction of travel sweeps out a volume
Only tractable in simple cases (e.g. linear translation)

If part of an object is in the volume, it was intersected by object
Collision reaction

Coefficient of restitution

\[ v(t_i) - N \cdot v(t_i) \]

\[ v(t_{i+1}) = v(t_i) - N \cdot v(t_i) - k(N \cdot v(t_i)) \]

\[ = v(t_i) - (1 + k)N \cdot v(t_i) \]

\( k \) – coefficient of restitution

But now want to add angular velocity contribution to separation velocity
Rigid body simulation

- **Object Properties**
  - Mass
  - Position
  - Linear & angular velocity
  - Linear & angular momentum

- **Calculate forces**
  - Wind
  - Gravity
  - Viscosity
  - Collisions

- **Calculate change in attributes**
  - Position
  - Linear & angular velocity
  - Linear & angular momentum

- **Calculate accelerations**
  - Linear & angular using mass and inertia tensor

Rick Parent

Computer Animation
Impulse response

How to compute the collision response of two rotating rigid objects?
Impulse response

Given
Separation velocity is to be negative of colliding velocity

Compute
Impulse force that produces sum of linear and angular velocities that produce desired separation velocity
Rigid body simulation

Impulse force

\[ j = f \Delta t \]

Separation velocity

\[ v_{rel}^+ = -c v_{rel}^- \]
Update linear and angular velocities as a result of impulse force

\[ \mathbf{v}_A^+ = \mathbf{v}_A^- + \frac{jn}{M_A} \]
\[ \mathbf{v}_B^+ = \mathbf{v}_B^- + \frac{jn}{M_B} \]
\[ \mathbf{\omega}_A^+ = \mathbf{\omega}_A^- + I_A^{-1}(t)(\mathbf{r}_A \times jn) \]
\[ \mathbf{\omega}_B^+ = \mathbf{\omega}_B^- + I_B^{-1}(t)(\mathbf{r}_B \times jn) \]
Velocities of points of contact

\[ r_A = p_A - x_A(t) \]
\[ r_B = p_B - x_B(t) \]
\[ v_{rel} = (\dot{p}_A(t) - \dot{p}_B(t)) \cdot n \]
\[ \dot{p}_A(t) = v_A(t) + \omega_A(t) \times r_A \]
\[ \dot{p}_B(t) = v_B(t) + \omega_B(t) \times r_B \]
Rigid body simulation

\[
\begin{align*}
v_{rel}^+ &= n \cdot (\dot{p}_A^+(t) - \dot{p}_B^+(t)) \\
v_{rel}^- &= n \cdot (v_A^+(t) + \omega_A^+(t) \times r_A - v_B^+(t) - \omega_B^+(t) \times r_B) \\
\varepsilon v_{rel}^- &= n \cdot \left( \frac{j}{M_A} + \frac{j n}{M_A} - \omega_A^- + I_A^{-1} (r_A \times j n) \right) + \frac{j n}{M_B} \left( \omega_B^- - I_B^{-1} (r_B \times j n) \right) \\
J &= \frac{1}{M_A} + \frac{1}{M_B} + n \cdot (I_A^{-1} (r_A \times n) \times r_A + I_B^{-1} (r_B \times n) \times r_B) \\
V_{rel}^- &= \varepsilon v_{rel}^- \\
j \text{ applied to object A; } -j \text{ applied to B}
\end{align*}
\]
Resting contact

Complex situations: need to solve for forces that prevent penetration, push objects apart, if the objects are separating, then the contact force is zero.