Towards a Unified Dynamical Solver for Computer Graphics

Jos Stam
Autodesk, Inc.
Toronto, Canada
Motivation

Ruysdael
Too many Solvers

rigid bodies  cloth  fluids

Solvers have to interact
Unified Solver

King Dome, Seattle, March 26, 2000
Goals

Handle general shapes

Plausible motion (use physics)

Robust, stable

Fast

Simple, easy to code
Approach

General shape model

Robust and stable dynamical solver for the shapes
Shape Model

**Theorem**
Any smooth surface $S$ can be approximated by a triangulation $T$.

Generalize to point, lines and solids
Shape Model

Want to handle particles, curves, surfaces and solids in a single framework.

0-simplex  1-simplex  2-simplex  3-simplex

→ k-simplex
Shape Model

Glue simplices together → simplicial complex
Shape Model
Shape Model
Shape Model

Definition purely topological

2 edges:
- (8,9) + (7,8)

7 triangles:
+ (0,1,4), + (1,2,5)
- (0,3,4), - (1,4,5)
+ (3,4,7), + (4,5,6),
+ (4,6,7)

(i,j,k) = -(i,k,j)
Shape Model

Single class for all primitives:

```java
class simplex {
    int k;
    int sign;
    int vertex[k+1];
    int child[k+1];
    int n_parents;
    int parent[n_parents];
}
```

Code not specific for a primitive: `simple`
Shape Model

Create smooth surfaces through subdivision

B-splines
Shape Model

2-simplices: Loop subdivision
Shape Model

3-simplices:

4 tetrahedra + 1 octohedron

Schaefer, Hakenberg and Warren 2004
Shape Model

3 possible diagonals $\rightarrow$ bias
Shape Model

Half-edge based smoothing

Extends to triangles and tetrahedra
Shape Model
Create two half-edges for each edge:

\[
\begin{align*}
\text{he}[4*e+0] & \quad \text{he}[4*e+1] \\
\circ & \quad \circ \\
\text{he}[4*e+2] & \quad \text{he}[4*e+3] \\
\circ & \quad \circ \\
\end{align*}
\]

\[
P0 = P1; \quad P1 = 0;
\]

\[
\begin{align*}
\text{for} \ ( \text{int} \ e=0 \ ; \ e<n\_edges \ ; \ e++ \ ) \ {\{} \\
\quad \text{int} \ e0 = \text{edge}[e].\text{vertex}[0]; \\
\quad \text{int} \ e1 = \text{edge}[e].\text{vertex}[1]; \\
\quad \text{P1}[e0] += \text{he}[4*e+0]*\text{P0}[e0] + \text{he}[4*e+1]*\text{P0}[e1]; \\
\quad \text{P1}[e1] += \text{he}[4*e+2]*\text{P0}[e1] + \text{he}[4*e+3]*\text{P0}[e0]; \\
\{ \}
\end{align*}
\]

Easy GPU implementation (thanks Nicolas Burtnyk)
Shape Model

demo
Dynamics of Complexes

\[ x(t) = (x_1(t), x_2(t), \ldots, x_n(t)) \]
Dynamics of Complexes
Dynamics of Complexes

Newton’s Law:

\[ \ddot{x} = - \nabla f(x) + f_e \]

\[ x(0) = x_0 \]

\[ \dot{x}(0) = v_0 \]

\[ E = \frac{1}{2} \dot{x}^2 + f(x) - f_e \cdot x \]
The Spring

\[ \ddot{x} = -x \]

\[ x(0) = x_0 \]
\[ \dot{x}(0) = v_0 \]

\[ E = \frac{1}{2} \dot{x}^2 + \frac{1}{2} x^2 \]
The Spring: Phase Space
The Spring: Vector Field
The Spring: Explicit

\[(x^k, v^k) \rightarrow (x^{k+1}, v^{k+1})\]
The Spring: Explicit

\[(x^k, v^k)\]

\[(x^{k+1}, v^{k+1})\]
The Spring: Implicit

$$(x^k, v^k)$$

$$(x^{k+1}, v^{k+1})$$
The Spring: Symplectic

\[(x^k, v^k)\]
The Spring: Symplectic
The Spring: Symplectic
The Spring: Symplectic
The Spring
demo
Strategy

Implicit on velocity

Explicit on position

Difficulties:

(1) Collisions

(2) Stable Stiff Deformations
Collisions

Penalty vs space-time
Space-Time

time

space (1D)
Space-Time

time

dt

space (1D)
Space-Time

$V_t = a_t - b_t = 0$

$V_0 < 0$

$V_1 > 0$
Space-Time

\[ t = \frac{V_0}{(V_0 - V_1)} \]
Collision Response

elastic
Collision Response

inelastic
Collision Response

friction
Higher Dimensions

2D: point / edge
Higher Dimensions

2D: point / edge
Higher Dimensions

2D: point / edge

\[ V_t = (b_t - a_t) \times (c_t - a_t) = 0 \]

Quadratic equation
Higher Dimensions

Necessary but **not** sufficient
Higher Dimensions

3D: point / triangle and edge / edge
Higher Dimensions

3D: point / triangle and edge / edge

Volume of a tetrahedron
Higher Dimensions

Intersection when volume is zero

\[ V_t = \langle (b_t - a_t) \times (c_t - a_t), d_t - a_t \rangle \geq 0 \]

Cubic equation
Higher Dimensions

Summary:

If $V_0 \cdot V_1 > 0$ stop

Find $t$ such that $V_t = 0$

Check if primitives overlap at $t$

If yes handle collision
Adding Thickness
## Adding Thickness

<table>
<thead>
<tr>
<th></th>
<th>Quadratic (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="example.png" alt="Image" /></td>
<td><img src="example.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Quartic (4)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="example.png" alt="Image" /></td>
<td><img src="example.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sextic (6)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="example.png" alt="Image" /></td>
<td><img src="example.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sextic (6)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="example.png" alt="Image" /></td>
<td><img src="example.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Many Primitives

Use hierarchical data structures for speed

Space time AABB tree (simple)
Many Primitives

Use hierarchical data structures for speed
Many Primitives

Use hierarchical data structures for speed
Many Primitives

Use hierarchical data structures for speed
Many Primitives
Many Primitives
Many Primitives
Many Primitives

Etc. Expensive in General
Many Primitives

Our approach:
Iterate over entire time step
Until all collisions resolved.

Avoids lockups and Zeno’s paradox
Many Primitives
Many Primitives
Many Primitives
Many Primitives
Many Primitives
Many Primitives
Many Primitives
Many Primitives
Many Primitives
Many Primitives

1D demo
(t key for space-time)

3D demo
Deformations

Treat deformations as constraints

Inspiration: Jakobsen, GDC 2001
Deformations

Treat deformations as constraints

\[ C(x) = 0 \]

Find \( v \) such that \( C(x + v) = 0 \)

\[ x = x + v \]
Deformations

1D: stretch/compression

2D: shear

3D: bending/torsion
Deformations

1-simplex            2-simplex              3-simplex

1                      3                      6

1/2                    3                      12

1/3                    1/2                    6
Deformations

Bridson, Marino and Fedkiw SCA 2003
Deformations

Find $\alpha$ such that $C(x + v + \alpha d) = 0$

Using Newton iterations:

$$\alpha = -C(x + v) \frac{dC(x+v)}{d\alpha}$$

Update velocity:

$$v = v + \alpha d$$
Stretch
Shear
Bend
Bend
Bend
General Solver

Solve consists of iterations:

Collision, self-collision, stretch, bend, shear, etc.

Assign max #iterations to each relation then interleave
## General Solver

<table>
<thead>
<tr>
<th>Category</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bend (9)</td>
<td>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26</td>
</tr>
<tr>
<td>Shear (7)</td>
<td>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26</td>
</tr>
<tr>
<td>Stretch (26)</td>
<td>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26</td>
</tr>
<tr>
<td>Self-Collisions (7)</td>
<td>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26</td>
</tr>
<tr>
<td>Collisions (6)</td>
<td>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26</td>
</tr>
</tbody>
</table>
## General Solver

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bend</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Shear</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Stretch</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Self-Collisions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Custom</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Collisions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Demos

Curve demo

Cloth demo
Torture tests
Torture tests
Torture tests
Torture tests
Simple Liquids

Movie 1  Movie 2
Solids
Pressure Model

Use gas law: $pV = c\ M$

$T = \text{const, } M = \text{mass}$

Work with Duncan Brinsmead
Pressure Model

No pressure
Pressure Model

Volume conservation
Pressure Model

Under pressure
Pressure Model

Under a lot of pressure
Pressure Model

Air tightness + Pump rate
Pressure Model

More examples

Inflatable Girl  Inflatable Guy
Cloth Examples
More Cloth Examples

Rigid bodies

ballerina

Cloth drop

Table cloth drop
Conclusions

Still work in progress…

Refine Liquids, Curves, Cloth and Solids

Faster Collisions?

Feedback?