Simulation in Computer Graphics

Deformable Objects

Matthias Teschner

Computer Science Department
University of Freiburg
Outline

- introduction
- forces
- performance
- collision handling
- visualization
Motivation

- sets of particles are used to model time-dependent phenomena such as ropes, cloth, deformable objects
- forces between particles account for resistance to stretch, shear, bend, volume changes ...

1D, 2D, and 3D mass-point systems, University of Freiburg
Example

- discretization of an object into mass points
- representation of internal forces between mass points, e.g. spring forces
- computation of the dynamics, positions and velocities at discrete time points
Applications

- entertainment technologies
  - cloth
  - facial expressions
- computational medicine
  - medical training
  - pre-operative surgical planning

Bridson, Fedkiw, Anderson, Stanford University
Outline

- introduction
- forces
  - examples
  - energy constraints
  - damping
  - plasticity and other effects
- performance
- collision handling
- visualization
Internal Forces

- Internal forces are symmetric with respect to at least two mass points

\[ F_{ij}^{\text{int}} + F_{ji}^{\text{int}} = 0 \]

- Sum of internal forces is zero

\[ \sum_i \sum_j F_{ij}^{\text{int}} = 0 \]

- Internal forces cause no torque

\[ F_{ijk}^{\text{int}} + F_{jki}^{\text{int}} + F_{kij}^{\text{int}} = 0 \]
Internal Forces

- Internal forces are defined for at least two points. The equation \( F_{ij}^{\text{int}} + F_{ji}^{\text{int}} = 0 \) implies that \( F_{ii}^{\text{int}} = 0 \).
- Internal forces do not influence the global dynamic behavior of a mass-point system (linear and angular momentum is preserved).
- Spring force is an internal force.
- External forces can change linear and angular velocity of a mass-point system.
- Is gravitational force an internal force?
Gravity

- objects with positions \( x_1, x_2 \) and masses \( m_1, m_2 \) attract each other with forces \( \mathbf{F}_1, \mathbf{F}_2 \)

\[
\mathbf{F}_1 = -\mathbf{F}_2 = G \frac{m_1 m_2}{|x_2 - x_1|^2} \frac{x_2 - x_1}{|x_2 - x_1|}
\]

- \( G \) is the gravitational constant

\( G \approx 6.67 \times 10^{-11} \text{N m}^2\text{kg}^{-2} \)

- internal force, if applied to both objects
Gravity

- on earth
  - gravity is dominated by earth
  - mass $m_1$ of the earth is constant
  - distance from surface to center is nearly constant
  - $\mathbf{g} = G \frac{m_1}{|\mathbf{x}_2 - \mathbf{x}_1|^2} \frac{\mathbf{x}_2 - \mathbf{x}_1}{|\mathbf{x}_2 - \mathbf{x}_1|} = \text{const}$
  - $\mathbf{g}$ is an acceleration pointing towards the earth center
- in virtual environments, the direction depends on the coordinate system, e.g. $\mathbf{g} = 9.81 \cdot (0, 0, -1)^T \text{m s}^{-2}$
- force exerted to $m_2$ due to gravity: $\mathbf{F}_2 = m_2 \cdot \mathbf{g}$
- external force, if not applied to both objects
Elastic Spring

- $k$ - spring stiffness
- $L$ - initial spring length
- $l$ - current spring length

- linear force-deformation relation (Hooke’s law)
  \[ F = k(L - l) \]

- simple mechanism for internal forces
- elasticity: ability of a spring to return to its initial configuration in the absence of forces
Mass-Spring System in 3D

- non-linear relation of forces and mass-point positions

\[ \mathbf{F}_0 = \sum_i k_i (|\mathbf{x}_i - \mathbf{x}_0| - L_i) \frac{\mathbf{x}_i - \mathbf{x}_0}{|\mathbf{x}_i - \mathbf{x}_0|} \]
A Simple Deformable Object

- discretize the object into mass points
- define the connectivity (topology, adjacencies of mass points)
- set model parameters
  - point: mass, position, velocity
  - spring: stiffness, initial length
- compute forces: spring force, gravity
- update positions and velocities of all mass points with a numerical integration scheme, e.g.

\[ \mathbf{v}_{i}^{t+h} = \mathbf{v}_{i}^{t} + h \frac{1}{m_{i}} \mathbf{F}_{i}^{t} \quad \mathbf{x}_{i}^{t+h} = \mathbf{x}_{i}^{t} + h \mathbf{v}_{i}^{t+h} \]
Explicit Numerical Integration

- e.g., Heun
  - for all mass points compute $k_{i,1}, l_{i,1}$
  - for all mass points compute $k_{i,2}, l_{i,2}$
  - for all mass points compute $x_i^{t+h}, v_i^{t+h}$

- $l_{i,1}$ depends on position $x_i^t$ and all connected neighbors $x_j^t$,

- $l_{i,2}$ depends on the predicted position $x_i^{t} + k_{i,1}h$ and on the predicted positions of all neighbors $x_j^{t} + k_{j,1}h$,

\[
\begin{align*}
k_{i,1} &= \dot{x}_i^t \\
l_{i,1} &= \dot{v}_{x_i^t, \ldots x_j^t} \\
k_{i,2} &= \dot{x}_i^t + l_{i,1}h \\
l_{i,2} &= \dot{v}_{x_i^{t+k_{i,1}h}, \ldots, x_j^{t+k_{j,1}h}} \\
x_i^{t+h} &= x_i^t + h\left(\frac{1}{2}k_{i,1} + \frac{1}{2}k_{i,2}\right) \\
v_i^{t+h} &= v_i^t + h\left(\frac{1}{2}l_{i,1} + \frac{1}{2}l_{i,2}\right)
\end{align*}
\]
Implicit Numerical Integration

- e.g., implicit Euler
- \( \mathbf{x}^t = (\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_n^T)^T \)
- \( \mathbf{v}^t = (\mathbf{v}_1^T, \mathbf{v}_2^T, \ldots, \mathbf{v}_n^T)^T \)
- \( \mathbf{F}^t(\mathbf{x}^t) = (\mathbf{F}_1^T, \mathbf{F}_2^T, \ldots, \mathbf{F}_n^T)^T \)
- \( \mathbf{M} = \text{diag}(m_1, m_1, m_1, \ldots, m_n, m_n, m_n) \in \mathbb{R}^{3n \times 3n} \)
- \( \mathbf{M}\mathbf{v}^{t+h} = \mathbf{M}\mathbf{v}^t + h\mathbf{F}^{t+h}(\mathbf{x}^{t+h}) \)
- force linearization
- \( \mathbf{M}\mathbf{v}^{t+h} = \mathbf{M}\mathbf{v}^t + h\mathbf{F}^t(\mathbf{x}^t) + h^2 \frac{\partial \mathbf{F}^t}{\partial \mathbf{x}^t} \mathbf{v}^{t+h} \quad \frac{\partial \mathbf{F}^t}{\partial \mathbf{x}^t} = \mathbf{J}^t \in \mathbb{R}^{3n \times 3n} \)
- \( (\mathbf{M} - h^2\mathbf{J}^t) \mathbf{v}^{t+h} = \mathbf{M}\mathbf{v}^t + h\mathbf{F}^t(\mathbf{x}^t) \)
Implicit Numerical Integration

- in the Jacobian $J^t$, a spring force between $x^t_i$ and $x^t_j$ is represented by four sub matrices

\[ J^t_{i,j} \in \mathbb{R}^{3\times3}, \quad J^t_{j,i} \in \mathbb{R}^{3\times3}, \quad J^t_{i,i} \in \mathbb{R}^{3\times3}, \quad J^t_{j,j} \in \mathbb{R}^{3\times3} \]

that are accumulated at positions $(3i, 3j)$, $(3j, 3i)$, $(3i, 3i)$, $(3j, 3j)$

\[ J^t_{i,i} = \frac{\partial F^t_i}{\partial x^t_i} \in \mathbb{R}^{3\times3} \]

\[ J^t_{i,i} = \frac{\partial}{\partial x^t_i} k_s \left( (x_j - x_i) - L_s \frac{x_j - x_i}{|x_j - x_i|} \right) \]

\[ = k_s \left( -I + \frac{L_s}{|x_j - x_i|} \left( I - \frac{1}{|x_j - x_i|^2} (x_j - x_i)(x_j - x_i)^T \right) \right) \]

\[ = -J^t_{i,j} = J^t_{j,i} = -J^t_{j,j} \]
Outline

- introduction
- forces
  - examples
  - energy constraints
  - damping
  - plasticity and other effects
- performance
- collision handling
- visualization
Spatial Discretization

- deformable objects are commonly discretized into mass points and simplices
  - line segments in 1D, triangles in 2D, tetrahedrons in 3D

![Spatial Discretization](image)

- 1349 mass points
- 4562 tetras
- 6888 springs

- 2949 mass points
- 10257 tetras
- 15713 springs

- 8 mass points
- 5 tetras
**Generalized Springs**

- can be used to preserve, e.g.,
  - a distance between two points
  - an area defined by three points
  - a volume defined by four points
- forces are derived from constraints \( C \)
- \( C \) depends on mass point positions
- \( C(x_1, \ldots, x_n) = 0 \) iff the constraint is met, e.g. a current distance equals a goal distance, a current area equals a goal area, ...
- motivation [demo](#)
Constraint Forces

- potential energy $E$ based on constraint $C$
  \[ E(x_1, \ldots, x_n) = \frac{1}{2} k C(x_1, \ldots, x_n)^2 \]

- $E = 0$ iff the constraint is met
- $E > 0$ iff the constraint is not met

- force at mass point $j$ based on the potential energy $E$
  \[
  F_j(x_1, \ldots, x_n) = -\frac{\partial}{\partial x_j} E(x_1, \ldots, x_n)
  = -k C(x_1, \ldots, x_n) \frac{\partial C(x_1, \ldots, x_n)}{\partial x_j}
  \]
Constraint Forces

- for a constraint \( C \), the sum of constraint forces at all involved mass points is equal to zero
  \[ \sum_j F_j(x_1, \ldots, x_n) = 0 \]

- linear and angular momentum of the system \((x_1, \ldots, x_n)\) are preserved

- constraint forces are internal forces (conservative forces)
Distance Preservation

- preserve distance $L$ between $x_1$ and $x_2$

$$C_d(x_1, x_2) = |x_1 - x_2| - L$$

$$x_1 = (x_1, y_1, z_1)^T \quad x_2 = (x_2, y_2, z_2)^T$$

$$C_d(x_1, y_1, z_1, x_2, y_2, z_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} - L$$

$$\frac{\partial C_d}{\partial x_1} = \left( \begin{array}{c} \frac{\partial C_d}{\partial x_1} \\ \frac{\partial C_d}{\partial y_1} \\ \frac{\partial C_d}{\partial z_1} \end{array} \right) = \frac{1}{|x_1 - x_2|} \left( \begin{array}{c} x_1 - x_2 \\ y_1 - y_2 \\ z_1 - z_2 \end{array} \right) = \frac{x_1 - x_2}{|x_1 - x_2|}$$
Distance Preservation

- forces $F_d$ based on constraint $C_d$

$$F_d(x_1) = -k_d C_d \frac{\partial C_d}{\partial x_1} = -k_d (|x_1 - x_2| - L) \frac{||x_1 - x_2||}{|x_1 - x_2|}$$

$$F_d(x_2) = -k_d C_d \frac{\partial C_d}{\partial x_2} = k_d (|x_1 - x_2| - L) \frac{||x_1 - x_2||}{|x_1 - x_2|}$$

- $F_d$ are spring forces with stiffness constant $k_d$
Area Preservation

- preserve area $A$ of a triangle $(x_1, x_2, x_3)$
- edges $e_1 = x_3 - x_1$  $e_2 = x_3 - x_2$
- constraint $C_a(x_1, x_2, x_3) = \frac{1}{2} |e_1 \times e_2| - A$
  
  $$t = e_1 \times e_2 \quad s = k_a \frac{C_a}{0.5 |e_1 \times e_2|}$$
- forces $F_a(x_1) = se_2 \times t$
  $F_a(x_2) = st \times e_1$
  $F_a(x_3) = st \times (e_2 - e_1)$
Volume Preservation

- preserve volume $V$ of a tetrahedron $(x_1, x_2, x_3, x_4)$
- edges $e_1 = x_2 - x_1, e_2 = x_3 - x_1, e_3 = x_4 - x_1$
- constraint $C_v(x_1, x_2, x_3, x_4) = \frac{1}{6} e_1(e_2 \times e_3) - V$
- forces $F_v(x_1) = k_v C_v (e_2 - e_1) \times (e_3 - e_1)$
  $F_v(x_2) = k_v C_v e_3 \times e_2$
  $F_v(x_3) = k_v C_v e_1 \times e_3$
  $F_v(x_4) = k_v C_v e_2 \times e_1$
volume preservation can be used to mimic curvature preservation at adjacent triangles

volume forces can mimic bending forces

curvature can be preserved by preserving the volume of the virtual tetrahedron \((x_1, x_2, x_3, x_4)\)
Constraint Forces - Summary

- powerful mechanism to preserve various characteristics (constraints)
- are internal forces, preserve linear and angular momentum
- are defined for sets of mass points
- can be combined, weighted with stiffness constants
- drawbacks
  - can be computationally expensive
  - non-intuitive parameters in case of combined constraints
  - can be redundant or competing
Outline

- introduction
- forces
  - examples
  - energy constraints
  - damping
  - plasticity and other effects
- performance
- collision handling
- visualization
Damping Forces

- are proportional to a velocity
- act in the direction opposite to a velocity
- model friction
- can improve the stability of a system
- should not slow down the movement of a system
Point Damping

- damping force according to the velocity of a mass point
- force is applied in opposite direction to the velocity
  \[ m\ddot{x}_t = F_t - \gamma\dot{x}_t \]
- force at a point is "used" for acceleration and damping
  \[ m\ddot{x}_t + \gamma\dot{x}_t = F_t \]
- e.g., mass point under gravity does not accelerate iff gravity and damping cancel out each other
Explicit Point Damping

- damping does not always damp
- $\mathbf{v}$ - velocity without damping
- $\mathbf{v}'$ - velocity with added damping
- $\mathbf{F}^d$ - damping force
**Implicit Point Damping**

- considering the current velocity for damping can cause problems
  \[ \mathbf{v}_{t+h} = \mathbf{v}_t + h \frac{1}{m} (\mathbf{F}_t - \gamma \mathbf{v}_t) = \mathbf{v}_t + h \frac{1}{m} (\mathbf{F}_t - \mathbf{F}_t^d) \]

- considering the velocity of the next time step reduces problems
  \[ \mathbf{v}_{t+h} = \mathbf{v}_t + h \frac{1}{m} (\mathbf{F}_t - \gamma \mathbf{v}_{t+h}) = \mathbf{v}_t + h \frac{1}{m} (\mathbf{F}_t - \mathbf{F}_{t+h}^d) \]

- can still be directly solvable for \( \mathbf{v}_{t+h} \), e.g.,
  \[ \mathbf{v}_{t+h} = \frac{1}{1 + \frac{h}{m} \gamma} \left( \mathbf{v}_t + h \frac{1}{m} \mathbf{F}_t \right) \]
Explicit Spring Damping

- damping force according to the relative velocity of adjacent mass points $x_1$ and $x_2$
- normalized direction
  \[ \hat{d}_t = \frac{x_2 - x_1}{|x_2 - x_1|} \]
- difference of the magnitudes of velocities projected onto $\hat{d}_t$
  (magnitude of the relative velocity)
  \[ m_t = v_{2,t} \hat{d}_t - v_{1,t} \hat{d}_t \]
- damping forces
  \[ F_{1,t} = \gamma m_t \hat{d}_t \quad F_{2,t} = -\gamma m_t \hat{d}_t \]
Implicit Spring Damping

- generally more robust

\[ F_{1,t} = \gamma m_{t+h} \hat{d}_{t+h} \quad F_{2,t} = -\gamma m_{t+h} \hat{d}_{t+h} \]

- implementation in two integration steps
  - first step predicts positions and velocities without damping
  - second step corrects predicted quantities with added damping

- implementation in one integration step
  - predict positions and velocities within the damping force computation, e.g. using Euler
  - prediction and actual integration can be done with different schemes
Demos

benefits and drawbacks of damping
Damping - Summary

- point and spring damping influence the stability
- implicit forms are preferable due to time discretization
- reduces oscillations
- point damping affects the global object dynamics
- integration schemes can add artificial point damping (which cannot be controlled by the user)
- spring damping does not affect the global object dynamics
Outline

- introduction
- forces
  - examples
  - energy constraints
  - damping
  - plasticity and other effects
- performance
- collision handling
- visualization
Elasticity and Plasticity

- elastic deformation is reversible
- plastic deformation is not reversible

original shape

deforation due to forces

elastic object
reversible deformation

plastic object
irreversible deformation
Elasticity and Plasticity

- decomposition of deformation
  \[ \epsilon = \epsilon_{\text{elastic}} + \epsilon_{\text{plastic}} \]

- decomposition of corresponding forces
  \[ F = k\epsilon = k\epsilon_{\text{elastic}} + k\epsilon_{\text{plastic}} \]

- only elastic forces are considered
  \[ F_{\text{elastic}} = k\epsilon - k\epsilon_{\text{plastic}} = k\epsilon_{\text{elastic}} \]
**Implementation**

- **initialization**
  \[ \epsilon_{\text{plastic}} = 0 \]

- **update**
  
  compute \( \epsilon \)

  \[ \epsilon_{\text{elastic}} = \epsilon - \epsilon_{\text{plastic}} \]

  if \( \epsilon_{\text{elastic}} > \text{yield} \) then \( \epsilon_{\text{plastic}} = \epsilon_{\text{plastic}} + \text{creep} \ \epsilon_{\text{elastic}} \)

  if \( \epsilon_{\text{plastic}} > \text{max} \) then \( \epsilon_{\text{plastic}} = \text{max} \)

- **yield, creep, max** are user-defined parameters
Elastic and Plastic Deformation
Resting Distance, Area, Volume

- plastic deformation corresponds to adjusting the resting distance between mass points
- principle can also be applied to other properties, e.g. area, volume
- adjustment of resting states causes internal forces
- can be used for effects such as contraction
**Strain Limiting**

- limited deformation
- geometric, position-based implementation

\[
\text{if } |x_1 - x_2| > \alpha L \text{ then } \\
\begin{align*}
x_1 &= x_1 + \frac{m_2}{m_1 + m_2}(|x_2 - x_1| - \alpha L)\frac{x_2 - x_1}{|x_2 - x_1|} \\
x_2 &= x_2 - \frac{m_1}{m_1 + m_2}(|x_2 - x_1| - \alpha L)\frac{x_2 - x_1}{|x_2 - x_1|}
\end{align*}
\]
Strain Limiting

- implementation approximates a bi-phasic force-deformation relation
- position update can be performed after the integration step
- iterative implementation for mass-point systems
- preserves linear and angular momentum
  - corresponds to some internal forces

\[ \alpha L \]

(deformation vs. force graph)
Forces - Summary

- external forces change the linear and angular momentum of a system, e.g. gravity, point damping
- internal forces can preserve characteristics, e.g. distances, areas, volumes
- damping forces improve the stability of a system
- resting length adjustments, symmetric position or momentum adjustments can mimic internal forces, e.g. for plasticity, stiff springs
- challenge: stable simulation of stiff, non-oscillating deformable objects without explicit or artificial point damping
Outline

- introduction
- forces
- performance
- collision handling
- visualization
Performance

- criteria
  - system updates per second (frames per second)
  - simulation time step

- parameters
  - number of primitives (mass points, distances, volumes ...)
  - internal and external forces
  - numerical integration scheme
  - additional costs for, e.g., collision handling, rendering, ...
Performance - Example

- cube with 4096 mass points, 16875 tetrahedrons, 22320 springs, distance and volume forces, gravity, Pentium 4, 2GHz

<table>
<thead>
<tr>
<th>method</th>
<th>error order</th>
<th>time step [ms]</th>
<th>comp. time [ms]</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>expl. Euler</td>
<td>1</td>
<td>0.5</td>
<td>9.5</td>
<td>0.05</td>
</tr>
<tr>
<td>RK 2</td>
<td>2</td>
<td>3.8</td>
<td>18.9</td>
<td>0.20</td>
</tr>
<tr>
<td>impl. Euler</td>
<td>1</td>
<td>49.0</td>
<td>172.0</td>
<td>0.28</td>
</tr>
<tr>
<td>RK 4</td>
<td>4</td>
<td>17.0</td>
<td>50.0</td>
<td>0.34</td>
</tr>
<tr>
<td>Verlet</td>
<td>3</td>
<td>11.5</td>
<td>9.5</td>
<td>1.21</td>
</tr>
</tbody>
</table>
Outline

- introduction
- forces
- performance
- collision handling
- visualization
Plane Representation

- In 3D, a plane can be defined with a point $\mathbf{x}_p$ on the plane and a normalized plane normal $\mathbf{n}_{\text{plane}}$.
- The plane is the set of points $\mathbf{x}$ with $\mathbf{n}_{\text{plane}} \cdot (\mathbf{x} - \mathbf{x}_p) = 0$.
- For a point $\mathbf{x}$, the distance to the plane is $d = \mathbf{n}_{\text{plane}} \cdot (\mathbf{x} - \mathbf{x}_p)$. 

![Diagram of plane representation]
Collision Response

- if a collision is detected, i.e. $d < 0$, a collision impulse is computed that prevents the interpenetration of the mass point and the plane (wall)
- we first consider the case of a particle-particle collision with $\mathbf{n}$ being the normalized direction from $\mathbf{x}_2$ to $\mathbf{x}_1$
- the response scheme is later adapted to the particle-plane case
Coordinate Systems

- velocities $v$ before the collision response and velocities $V$ after the collision response are considered in the coordinate system defined by collision normal $n$ and two orthogonal normalized tangent axes $t$ and $k$

- e.g. 
  \[
  \begin{pmatrix}
  v_{1,n} \\
  v_{1,t} \\
  v_{1,k}
  \end{pmatrix}
  =
  \begin{pmatrix}
  n_x & n_y & n_z \\
  t_x & t_y & t_z \\
  k_x & k_y & k_z
  \end{pmatrix}
  \begin{pmatrix}
  v_{1,x} \\
  v_{1,y} \\
  v_{1,z}
  \end{pmatrix}
  
- the velocity $V$ after the response is transformed back
  \[
  \begin{pmatrix}
  V_{1,x} \\
  V_{1,y} \\
  V_{1,z}
  \end{pmatrix}
  =
  \begin{pmatrix}
  n_x & t_x & k_x \\
  n_y & t_y & k_y \\
  n_z & t_z & k_z
  \end{pmatrix}
  \begin{pmatrix}
  V_{1,n} \\
  V_{1,t} \\
  V_{1,k}
  \end{pmatrix}
  

Concept

- conservation of momentum

\[
\begin{align*}
m_1 V_{1,n} - m_1 v_{1,n} &= P_n \\
m_1 V_{1,t} - m_1 v_{1,t} &= P_t \\
m_1 V_{1,k} - m_1 v_{1,k} &= P_k \\
m_2 V_{2,n} - m_2 v_{2,n} &= -P_n \\
m_2 V_{2,t} - m_2 v_{2,t} &= -P_t \\
m_2 V_{2,k} - m_2 v_{2,k} &= -P_k
\end{align*}
\]

- coefficient of restitution, \( e = 1 \) elastic, \( e = 0 \) inelastic

\[V_{1,n} - V_{2,n} = -e(v_{1,n} - v_{2,n})\]

- friction opposes sliding motion along \( t \) and \( k \)

\[P_t = \mu P_n \quad P_k = \mu P_n\]
Linear System

- nine equations, nine unknowns

\[
\begin{pmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
-\mu & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\mu & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & m_1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & m_1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & m_1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 \\
\end{pmatrix}
\begin{pmatrix}
P_n \\
P_k \\
P_t \\
V_{1,n} \\
V_{1,t} \\
V_{1,k} \\
V_{2,n} \\
V_{2,t} \\
V_{2,k} \\
\end{pmatrix}
=
\begin{pmatrix}
-e(v_{1,n} - v_{2,n}) \\
0 \\
0 \\
0 \\
m_1 v_{1,n} \\
m_1 v_{1,t} \\
m_1 v_{1,k} \\
m_2 v_{2,n} \\
m_2 v_{2,t} \\
m_2 v_{2,k} \\
\end{pmatrix}
\]
Linear System

\[
\begin{bmatrix}
\frac{m_1 m_2}{m_1 + m_2} & 0 & 0 & - \frac{m_2}{m_1 + m_2} & 0 & 0 & \frac{m_1}{m_1 + m_2} & 0 & 0 \\
\frac{m_1 m_2 \mu}{m_1 + m_2} & 1 & 0 & - \frac{m_2 \mu}{m_1 + m_2} & 0 & 0 & \frac{m_1 \mu}{m_1 + m_2} & 0 & 0 \\
\frac{m_1 m_2 \mu}{m_1 + m_2} & 0 & 1 & - \frac{m_2 \mu}{m_1 + m_2} & 0 & 0 & \frac{m_1 \mu}{m_1 + m_2} & 0 & 0 \\
\frac{m_2}{m_1 + m_2} & 0 & 0 & \frac{1}{m_1 + m_2} & 0 & 0 & \frac{1}{m_1 + m_2} & 0 & 0 \\
\frac{m_2 \mu}{m_1 + m_2} & 0 & \frac{1}{m_1} & - \frac{m_2 \mu}{m_1 (m_1 + m_2)} & \frac{1}{m_1} & 0 & \frac{\mu}{m_1 + m_2} & 0 & 0 \\
\frac{m_1 \mu}{m_1 + m_2} & 0 & 0 & \frac{1}{m_1 + m_2} & 0 & 0 & \frac{1}{m_1 + m_2} & 0 & 0 \\
- \frac{m_1}{m_1 + m_2} & 0 & 0 & - \frac{1}{m_1 + m_2} & 0 & 0 & - \frac{m_1 \mu}{m_2 (m_1 + m_2)} & \frac{1}{m_2} & 0 \\
- \frac{m_1 \mu}{m_1 + m_2} & 0 & - \frac{1}{m_2} & - \frac{\mu}{m_1 + m_2} & 0 & 0 & - \frac{m_1 \mu}{m_2 (m_1 + m_2)} & 0 & \frac{1}{m_2}
\end{bmatrix}
\]
Particle-Plane

- plane has infinite mass and does not move: \( v_2 = V_2 = 0 \)
- columns 2, 3, 7, 8, 9 do not contribute to the solution
- to solve for the particle velocity \( V_1 \) after collision response, rows 4, 5, 6 have to be considered

\[
\begin{pmatrix}
\frac{m_2}{m_1+m_2} & \frac{1}{m_1+m_2} & 0 & 0 \\
\frac{m_2 \mu}{m_1+m_2} & \frac{m_2 \mu}{m_1(m_1+m_2)} & \frac{1}{m_1} & 0 \\
\frac{m_2 \mu}{m_1+m_2} & \frac{m_2 \mu}{m_1(m_1+m_2)} & 0 & \frac{1}{m_1}
\end{pmatrix}
\begin{pmatrix}
-ev_{1,n} \\
m_1 v_{1,n} \\
m_1 v_{1,t} \\
m_1 v_{1,k}
\end{pmatrix}
= 
\begin{pmatrix}
V_{1,n} \\
V_{1,t} \\
V_{1,k}
\end{pmatrix}
\]

- plane has infinite mass

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
\frac{\mu}{m_1} & 1 & 0 & 0 \\
\frac{\mu}{m_1} & 0 & 1 & 0 \\
\frac{\mu}{m_1} & 0 & 0 & \frac{1}{m_1}
\end{pmatrix}
\begin{pmatrix}
-ev_{1,n} \\
m_1 v_{1,n} \\
m_1 v_{1,t} \\
m_1 v_{1,k}
\end{pmatrix}
= 
\begin{pmatrix}
V_{1,n} \\
V_{1,t} \\
V_{1,k}
\end{pmatrix}
\]
Implementation

$$V_{t,n} = -ev_{t,n}$$
$$V_{t,t} = v_{t,t} - \mu(e + 1)v_{t,n}$$
$$V_{t,k} = v_{t,k} - \mu(e + 1)v_{t,n}$$

- $\mu$ is difficult to handle
- $|V_{t,t}| \leq |v_{t,t}|$ and $\text{sign}(V_{t,t}) = \text{sign}(v_{t,t})$ should be guaranteed
- $V_{t,t} = \mu v_{t,t}$, $V_{t,k} = \mu v_{t,k}$, $0 \leq \mu \leq 1$ is a useful simplification
Position Update

- the collision impulse updates the velocity
- however, the point is still in collision \((d < 0)\)
- for low velocities, the position update in the following integration step may not be sufficient to resolve the collision
- therefore, the position should be updated as well, e.g. \(x_{t+h} = x_t + d \cdot n\) which projects the point onto the plane
- the position update is not physically-motivated, it just resolves problems due to discrete time steps
Outline

- introduction
- forces
- performance
- collision handling
- visualization
Concept

- geometric combination of
  - a low-resolution tetrahedral mesh for simulation and
  - a high-resolution triangular mesh for visualization
- coupling by Barycentric coordinates of a surface point with respect to a corresponding tetrahedron
Surface-Volume Coupling

- A point $x_s$ can be represented with the points of a tetrahedron.
Barycentric Coordinates in 3D

- a point $x_s$ can be represented using $(x_1, x_2, x_3, x_4)$
  
  $$x_s = x_1 + \alpha_2(x_2 - x_1) + \alpha_3(x_3 - x_1) + \alpha_4(x_4 - x_1)$$
  
  $$x_s = (1 - \alpha_2 - \alpha_3 - \alpha_4)x_1 + \alpha_2x_2 + \alpha_3x_3 + \alpha_4x_4$$
  
  $$x_s = \alpha_1x_1 + \alpha_2x_2 + \alpha_3x_3 + \alpha_4x_4 \quad \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$$

- $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ are Barycentric coordinates of $x_s$ with respect to $(x_1, x_2, x_3, x_4)$
Properties

- \( 0 < \alpha_i < 1 \)
  - \( x_s \) is inside the convex combination of \((x_1, x_2, x_3, x_4)\), i.e. inside the tetrahedron

- \( \alpha_i = 0 \lor \alpha_i = 1 \)
  - \( x_s \) is on the surface of the tetrahedron

- \( \alpha_i < 0 \lor \alpha_i > 1 \)
  - \( x_s \) is outside the tetrahedron
Computation

\[ x_s = x_1 + \alpha_2(x_2 - x_1) + \alpha_3(x_3 - x_1) + \alpha_4(x_4 - x_1) \]

- leads to the following system

\[
\begin{pmatrix}
(x_2 - x_1) & (x_3 - x_1) & (x_4 - x_1)
\end{pmatrix}
\begin{pmatrix}
\alpha_2 \\
\alpha_3 \\
\alpha_4
\end{pmatrix}
= x_s - x_1
\]

- singular, if two edges of the tetrahedron are parallel!
- \( \alpha_1 \) is computed as \( \alpha_1 = 1 - \alpha_2 - \alpha_3 - \alpha_4 \)
**Implementation**

- **data structure**
  - for each point of the surface mesh, store Barycentric coords and the corresponding tetrahedron

- **pre-processing**
  - for each surface point, determine the closest tetrahedron of the simulation mesh (point of the surface mesh should be located inside a tetrahedron)
  - for each surface point, compute its Barycentric coords with respect to the corresponding tetrahedron

- **in each simulation step**
  - for each surface point, compute its position from its Barycentric coords and the positions of the mass points of the corresponding tetrahedron

- **demo**
Summary Mass-Point Systems

- forces, e.g. gravity, energy constraints, damping, plasticity
- numerical integration schemes (see particle systems)
- collision handling for planes
- visualization, combination of low-resolution simulation meshes with high-resolution visualization meshes
References