Simulation of Complex Nonlinear Elastic Bodies using Lattice Deformers

Taylor Patterson
University of Wisconsin-Madison

Nathan Mitchell
University of Wisconsin-Madison

Eftychios Sifakis
University of Wisconsin-Madison
Equation (2) can therefore be considered an implicit definition for this equation leads to an equilibrium configuration for the mesh, and we abstractly encode this state with a vector. Simulation utilizing the same muscle control parameters can later be included in extracted expressions via a full dynamic to be defined as functions of the input control parameters without fundamental to our control strategy, since it enables facial expressions We use a quasistatic simulation scheme where each input of muscle control parameters. This linear dependence on the spatial configuration. These steady state positions are defined implicitly with the definition of nodal forces can be summarized as term for quasi-incompressibility. Tetrahedra which contain facial remain passive during simulation. Furthermore, tendon tissue is an order of magnitude stiffer than muscle tissue. Tendon often extends into the belly of certain muscles forming an internal muscle mesh to represent muscle tissue, tendon tissue, and bone attachment regions. Tetrahedra are each split in two new triangles containing an affine function of muscle activations. This linear dependence responding current muscle activation level with distribution of a fully activated muscle and is weighted by the correcting current muscle activation level with distribution of a fully activated muscle and is weighted by the correct this guess by growing regions initially selected based on mesh connectivity as well as. Additionally, we rigidly attach tetrahedrons in the origin and insertion take extra care to model this layer when selecting the regions of the muscle/tendon geometry is known as an aponeurosis and can play a large role in many muscle functions [33], [16]. We vision of elements could solve this problem as well. 6. The edges of the newly generated triangles that do 7. Skip all entries containing 4. In both cases, a new vertex are each split in two new triangles containing 2. search on the sorted list. They can be found in logarithmic time via a binary tree could be used to reduce the number of elements or even though the simulation can handle multiple materials. We have presented a method for the physically-based an-
Features

- Arbitrary Materials
- Incompressibility
- Sub-voxel Precision
- Robust
- Parallelism
Motivation
2.1. DEFORMATION MAP AND DEFORMATION GRADIENT

Deformation Map
\[ \phi = \mathbb{R}^3 \rightarrow \mathbb{R}^3 \]

Deformation Gradient
\[ F = \frac{\partial \phi}{\partial X} \]

Energy Density
\[ \Psi = \Psi(F) \]

Elastic Energy
\[ E = \int_{\Omega} \Psi(F) \delta X \]

Neohookean elasticity
\[ \Psi = \frac{\mu}{2} (\|F\|_F^2 - 3) - \mu \log(J) + \frac{\kappa}{2} \log^2(J) \]
The Basics

Start with Energy Density

Apply Incompressibility Adjustment

Compute Discrete Energy, \( \hat{E} \)

Derive Nodal Forces, \( f(x) \)

Find Equilibrium \( f(x) = 0 \implies \min \hat{E} \)

Materials & Incompressibility

Boundary Cell Treatment

Newton Raphson Iteration
Incompressibility: Motivation

Importance

- Increasing Realism
- Human Flesh is Incompressible

Challenges

- Do we really have true volume preservation?
- Are we susceptible to locking behavior?
- Are our equations well conditioned?
Incompressibility: Materials

\[ \Psi = \Psi_0(F) + \frac{\kappa}{2} M^2(F) \]

Linear elasticity

\[ \Psi = \mu \| \epsilon \|_F^2 + \frac{\kappa}{2} \text{tr}^2(\epsilon) \]

Corotated linear elasticity

\[ \Psi = \mu \| F - R \|_F^2 + \frac{\kappa}{2} \text{tr}^2(S - I) \]

Neohookean elasticity

\[ \Psi = \frac{\mu}{2} (\| F \|_F^2 - 3) - \mu \log(J) + \frac{\kappa}{2} \log^2(J) \]

Raise \( \kappa \) to enforce incompressibility.

\[ \kappa \to \infty \]

Implies complete incompressibility
Corotated Material  -  Inaccurate Volume Preservation

Neohookean Material  -  Accurate Volume Preservation
Incompressibility: Materials

\[ \text{Incompressibility} \]

\[ tr^2(\mathcal{S}^2(I)) = 0 \implies \frac{\sigma_X}{\sigma_X} + \frac{\sigma_Y}{\sigma_Y} + \frac{\sigma_Z}{\sigma_Z} \approx \frac{1}{3} \approx 1 \]
Incompressibility: Locking

Source: Irving et al. 2007
Incompressibility: Locking

Require Volume Preservation Everywhere

Elastic Energy

\[ E = E_0 + \frac{\kappa}{2} \int_{\Omega} \mathbf{M}^2(F) \delta X \]

\[ E = E_0 + \frac{\kappa}{2} \int_{\Omega} \overline{M^2}(F) \delta X \]
Incompressibility: Conditioning

1. Start with Energy Density
2. Apply Incompressibility Adjustment
3. Compute Discrete Energy, $\hat{E}$
4. Derive Nodal Forces, $f(x)$
5. Find Equilibrium $f(x) = 0 \implies \min \hat{E}$
Incompressibility: Conditioning

Find Equilibrium  \( f(x) = 0 \implies \min \hat{E} \)

No internal forces implies equilibrium after deformation

Zero forces implies a minimum in the potential energy
Incompressibility: Conditioning

\[ \Psi(F) = \Psi_0 + \frac{\kappa}{2} \overline{M}^2(F) \]

\[ \Psi(F, p) = \Psi_0(F) + \alpha p \overline{M}(F) - \frac{\alpha^2 p^2}{2\kappa} \]
Incompressibility: Conditioning

\[ \Psi(F,p) = \Psi_0(F) + \alpha p M(F) - \alpha^2 p^2 + \frac{1}{2} \kappa M^2(F) \]

- Conditioning fix implemented at the energy level, instead of the PDE level (more flexible)
- Support for arbitrary nonlinear materials (vs. only linear and corotated elasticity)

[Zhu et al. 2010]
Using saddle point formulation

Using conventional minimization formulation
Using saddle point formulation

Using conventional minimization formulation
Boundary Cells: Motivation

Importance

Objects are not made of cubes

Visible grid artifacts (discontinuities)

Challenges

How accurate are we?

Can we make it go fast?

Interpolation discontinuities can be avoided by using tricubic interpolation
Boundary Cell Treatment

1. Start with Energy Density
2. Apply Incompressibility Adjustment
3. Compute Discrete Energy, $\hat{E}$
4. Derive Nodal Forces, $f(x)$
5. Find Equilibrium $f(x) = 0 \implies \min \hat{E}$
Boundary Cell Treatment

\[ E_{obj} = \sum E_{cell} \]

\[ E_{cell} = \int_{cell} \Psi(F) \partial X \]
Boundary Cell Treatment

\[ E_{obj} = \sum E_{cell} \]

Whole cells are easy

\[ E_{cell} = \int_{[0,h]^3} \Psi(F) \partial X \]
Boundary Cell Treatment

\[ E_{obj} = \sum E_{cell} \]

Fractional cells are tricky

\[ E_{cell} = \int_{\Omega_c} \Psi(F) \partial X \]

\[ = ??? \]
Boundary Cell Treatment

$\Omega_c = \cdots$

$\Rightarrow$ Compute analytically
  Very simple $\Psi$'s only

$\Rightarrow$ Monte-Carlo
  Expensive

$\Rightarrow$ Our solution:
  Efficient quadrature scheme
Boundary Cell Treatment

\[ E_{\text{cell}} = \int_{\Omega_c} \Psi(F) \partial X \]

\[ \approx \sum_{i=1}^{k} c_i \Psi(X_i) \]

\[ \sum c_i = \text{vol} \left( \Omega_c \right) \]

Note: \( k = 4 \) suffices for second order accuracy!
Choose $X_i, C_i$ such that

$$\Omega_c$$

Continuous Distribution Average = Discrete Distribution Average

Continuous Distribution Variance = Discrete Distribution Variance
Recap

- Produces 2nd order accuracy
- Fixed work per cell (4 pts max)
- Need to compute 4 sets of forces
We present a new algorithm for near-interactive simulation of skeleton-driven high resolution elasticity models. Our methodology is used for soft tissue deformation in character animation. The algorithm is based on a novel discretization of corotational elasticity over a hexahedral lattice. Within this framework we enforce positive definiteness of the stiffness matrix to allow efficient quasistatics and dynamics. In addition, we present a multigrid method that converges with very high efficiency. Our design targets performance through parallelism using a fully vectorized and branch-free SVD algorithm as well as a stable one-point quadrature scheme. Since body collisions, self collisions and soft constraints are necessary for real-world examples, we present a simple framework for enforcing them. The whole approach is demonstrated in an end-to-end production-level character skinning system.

CR Categories:
I.5.8 [Simulation and Modeling]. Types of Simulation—Animation

Keywords:
skinning, corotated elasticity, physics-based modeling, elastic deformations

Links:
DL PDF WE
Objective: Solve for \( f(x) = 0 \)

Solve \( f(x) = 0 \) \( \Rightarrow \) \( x_0 \leftarrow \) Initial guess

for \( k = 0, 1, \ldots \)

Solve \( \left\{ \frac{-\delta f}{\delta x} \right|_{x_k} \right\} \delta x = f(x_k) \)

Update \( x_{k+1} = x_k + \delta x \)

\( \Rightarrow \lim_{k} x_k = \) Sub-voxel accurate solution of \( f(x) = 0 \)
Objective: Solve for $f(x) = 0$

Solve $f(x) = 0 \xrightarrow{\text{Newton}} x_0 \leftarrow \text{Initial guess}$

for $k = 0, 1, \ldots$

Compute with one point method

Solve

$$\left\{ -\frac{\delta f}{\delta x} \right|_{x_k} \right\} \delta x = f(x_k)$$

Update $x_{k+1} = x_k + \delta x$

$\Rightarrow \lim x_k = \text{Sub-voxel accurate solution of}$

$$f(x) = 0$$
Objective: Solve for $f(x) = 0$

Solve $f(x) = 0 \overset{\text{Newton}}{\Rightarrow} x_0 \leftarrow \text{Initial guess}$

for $k = 0, 1, \ldots$

Compute with four point method

Solve $\left\{ \left. -\frac{\delta f}{\delta x} \right|_{x_k} \right\} \delta x = f(x_k)$

Update $x_{k+1} = x_k + \delta x$

$\Rightarrow \lim x_k = \text{Sub-voxel accurate solution of}$

$f(x) = 0$
Parallelism

Note: Intel AVX provides SIMD width of 8. Thus we use blocks of 2x2x2 cells.

Reassemble onto grid by accumulating between blocks.
### Performance

<table>
<thead>
<tr>
<th>Human model</th>
<th>Force Differentials</th>
<th>One QMR Iteration</th>
<th>Newton Iteration (including 100 QMR iterations)</th>
<th>Typical frame (4-6 Newton iterations)</th>
<th>Typical frame (3-5 Newton iterations)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(101K cells)</td>
<td>0.041</td>
<td>0.067</td>
<td>7.035</td>
<td>31.825</td>
<td>1.466</td>
</tr>
<tr>
<td></td>
<td>1 core (AVX) 1 thread</td>
<td>2 cores (AVX) 2 threads</td>
<td>4 cores (AVX) 4 threads</td>
<td>4 cores (AVX) 8 threads</td>
<td>1 core (AVX) 1 thread</td>
</tr>
<tr>
<td></td>
<td>0.023</td>
<td>0.037</td>
<td>4.066</td>
<td>18.186</td>
<td>0.842</td>
</tr>
<tr>
<td></td>
<td>0.015</td>
<td>0.025</td>
<td>2.727</td>
<td>12.193</td>
<td>0.640</td>
</tr>
<tr>
<td></td>
<td>0.015</td>
<td>0.024</td>
<td>2.645</td>
<td>11.919</td>
<td>0.626</td>
</tr>
<tr>
<td>(13.5K cells)</td>
<td>0.005</td>
<td>0.007</td>
<td>0.422</td>
<td>1.466</td>
<td>1 core (AVX) 1 thread</td>
</tr>
<tr>
<td></td>
<td>0.003</td>
<td>0.005</td>
<td>0.244</td>
<td>0.842</td>
<td>2 cores (AVX) 2 threads</td>
</tr>
<tr>
<td></td>
<td>0.002</td>
<td>0.003</td>
<td>0.184</td>
<td>0.640</td>
<td>4 cores (AVX) 4 threads</td>
</tr>
<tr>
<td></td>
<td>0.002</td>
<td>0.003</td>
<td>0.177</td>
<td>0.626</td>
<td>4 cores (AVX) 8 threads</td>
</tr>
</tbody>
</table>
Conclusion

- Arbitrary Materials
- Incompressibility
- Sub-voxel Precision
- Robust
- Parallelism
Limitations - Future Work

⇒⇒ No collisions yet...
   ... but [McAdams et al 2011] should work as an add-on
   ... and we already support soft constraints

⇒⇒ Krylov solvers vs Multigrid

⇒⇒ Adaptivity

⇒⇒ Cutting / Stitching / Topology change
Questions?

Thank you for listening.

Any questions?