Multi-Resolution Isotropic Strain Limiting

Huamin Wang James O'Brien Ravi Ramamoorthi

University of California, Berkeley

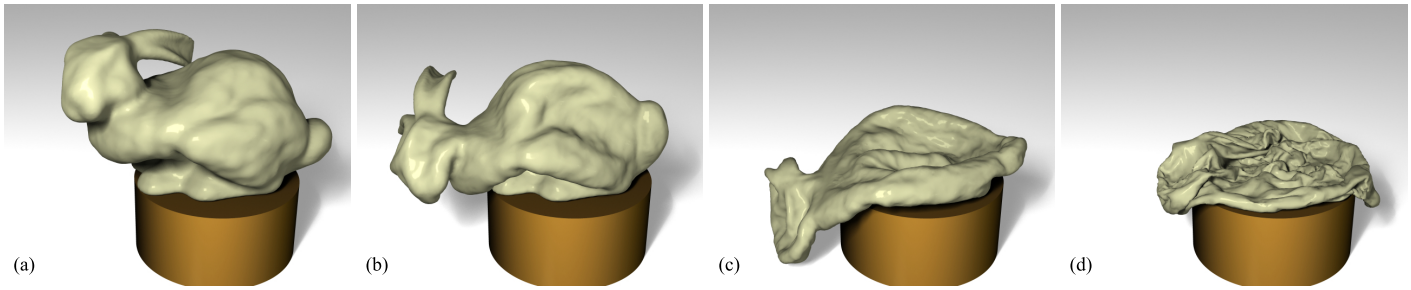

Figure 1: *Different strain and bending limits produce different final resting configurations for this thin-shell bunny. (a) strain* $[-5\%, 5\%]$*, bending angle* $[-3^{\circ/\text{cm}}, 3^{\circ/\text{cm}}]$*; (b) strain* $[-5\%, 5\%]$*, bending* $[-6^{\circ/\text{cm}}, 6^{\circ/\text{cm}}]$*; (c) strain* $[-10\%, 10\%]$*, bending* $[-12^{\circ/\text{cm}}, 12^{\circ/\text{cm}}]$*, and (d) strain* $[-10\%, 10\%]$*, no bending limit.*

Abstract

In this paper we describe a fast strain-limiting method that allows stiff, incompliant materials to be simulated efficiently. Unlike prior approaches, which act on springs or individual strain components, this method acts on the strain tensors in a coordinate-invariant fashion allowing isotropic behavior. Our method applies to both two- and three-dimensional strains, and only requires computing the singular value decomposition of the deformation gradient, either a small 2×2 or 3×3 matrix, for each element. We demonstrate its use with triangular and tetrahedral linear-basis elements. For triangulated surfaces in three-dimensional space, we also describe a complementary edge-angle-limiting method to limit out-of-plane bending. All of the limits are enforced through an iterative, non-linear, Gauss-Seidel-like constraint procedure. To accelerate convergence, we propose a novel multi-resolution algorithm that enforces fitted limits at each level of a non-conforming hierarchy. Compared with other constraint-based techniques, our isotropic multi-resolution strain-limiting method is straightforward to implement, efficient to use, and applicable to a wide range of shell and solid materials.

Keywords: multi-resolution, strain limiting, finite element method

1 Introduction

Many real-world materials exhibit macroscopic behavior that is compliant to small deformations, but beyond some threshold becomes highly resistant to larger deformations. Two very common examples include cloth and animal tissue. In cloth, the structure of fibers and threads easily accommodates small-to-moderate amounts of stretching, but once the structural slack has been taken up, resistance to further stretching increases dramatically. This biphasic behavior explains why soft, compliant fabrics such as silk can actually stop bullets [Goodfellow, 1887], and many researchers have identified this property as key in developing the characteristic wrinkle patterns observed in many fabrics [Provot, 1995; Bridson et al., 2003; Goldenthal et al., 2007; Müller, 2008; English and Bridson, 2008; Volino et al., 2009; Thomaszewski et al., 2009]. Similarly, animal tissue, such as skin or relaxed muscle, is soft and compliant to small strains but very tough and resistant for larger ones. The meditative practice of suspending a person by as few as two hooks through the skin provides a rather impressive example of this toughness [Forsyth and Simpson, 2008]. As with cloth, this nonlinear stress-to-strain relationship is key in capturing tissue's characteristic appearance.

Unfortunately, most simulation methods perform poorly for materials with highly incompliant constitutive regimes. Standard finite-element methods (including spring-and-mass systems) model incompliance using large material coefficients that can create severe time integration difficulties. Explicit integration methods will require infeasibly small time steps to avoid instability. Implicit methods may remain stable for reasonably sized time steps, but they experience other problems such as poor convergence for iterative solvers, high residuals that manifest as ghost forces, or excessive numerical damping. Further, many widely used methods are only semi-implicit, because they linearize the simulated system over a time step, and as a result they may exhibit artifacts or poor stability for stiff nonlinear systems.

An increasingly popular approach for dealing with this problem is to implement very stiff material behaviors using some form of constraint. A standard elastic model is used for small strains, but a hard limit on large strains is enforced using projection. This idea, known as *strain limiting*, has been widely employed with springs [Provot, 1995; Bridson et al., 2003; Goldenthal et al., 2007; Müller, 2008; English and Bridson, 2008; Volino et al., 2009], and recently Thomaszewski and his colleagues [2009] have applied the method to linear-basis triangular elements. Because the material's dynamic behavior is governed by compliant material properties, one can use a simple and cheap integration scheme, such as the

Contact email: {whmin, job, ravir}@eecs.berkeley.edu

To appear in the ACM SIGGRAPH Asia 2010 conference proceedings.

leap-frog integrator described by Müller [2008], while relying on a nonlinear Gauss-Seidel iterative projection to prevent excessive deformation. The projection process adjusts the positions and velocities of the nodes directly so that large amounts of elastic energy are not stored in the system, which remains stable. The energy discarded by the projection does manifest as additional damping, but only where material behavior would have exited the compliant base regimen.

In this paper we propose four complementary improvements to existing strain-limiting methods:

Isotropy — The method we describe operates on the per-element strain tensor in a coordinate-independent way. Rather than enforcing constraints on point-to-point distances or individual strain components, our method operates directly on the principal strains of the symmetric strain tensor. This allows the model to behave in an isotropic fashion that does not depend on the underlying parametrization.

Efficient triangular and tetrahedral Elements — While point-to-point constraints can be implemented efficiently, previous methods for continuum-based strain limiting on triangle meshes required inverting 6×6 matrices for each element [Thomaszewski et al., 2009]. Extension to tetrahedral elements would have presumably required inverting 12×12 matrices. Our method requires computing a singular value decomposition of the 2×2 or 3×3 deformation gradient of each offending element. This decomposition is often already available when the simulation method uses a corotational strain metric [Müller and Gross, 2004].

Bending-angle limits — Objects modeled as thin shells frequently combine triangular elements that measure in-plane strain with a discrete bending metric. (For example, [Bridson et al., 2003; Grinspun et al., 2003].) Just as biphasic elasticity can be useful, biphasic bending resistance also has uses. We describe a simple method for limiting the bending angles at the edges between triangles that complements our in-plane strain limiting.

Multi-resolution limit enforcement — While strain limiting offers a number of advantages compared to simply using stiff elements with an iterative implicit solver, the global solution method essentially reduces to nonlinear Gauss-Seidel or Jacobi iterations. For large systems, the convergence rate of Gauss-Seidel/Jacobi can become very slow. Hierarchical enforcement can greatly accelerate convergence [Müller, 2008], but the unilateral nature of the constraints makes it difficult to build coarse-level constraints in a way that captures fine-level limits without suppressing fine details. To address this problem, we present a training algorithm to automatically generate different limiting parameters at each level of a multi-resolution mesh hierarchy.

The multi-resolution limiting approach provides an efficient way to generate stiff behaviors by preventing thin shells and elastic bodies from undergoing large deformations, without using stiff constitutive relations that would prohibit large time steps or lead to instability issues. It functions as a drop-in component for an existing finite element simulator, as shown in Figure 2. We use the same structure as described by Bridson and his colleagues [2003] where the limiting method functions as a velocity filter between the velocity update and the position update.

2 Related Work

The use of simulation-based methods to animate deformable objects has been an active area of graphics research for over twenty years. Key concepts in the area were originally introduced by Terzopoulos with his collaborators [1987] and in other contemporaneous work. The survey article by Gibson and Mirtich [1997] details much of the early work on deformable modeling, while Nealen and coauthors survey more recent approaches [2005].

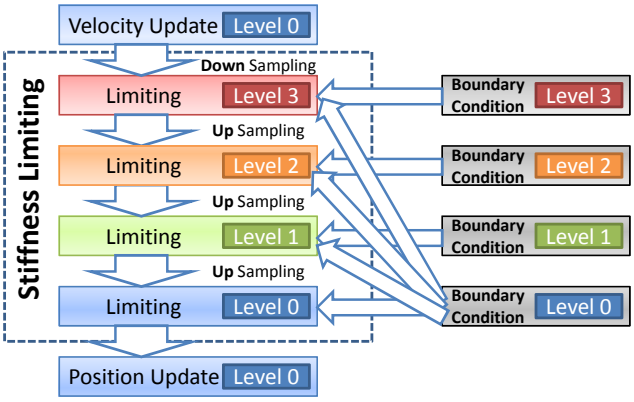

Figure 2: *The pipeline of a simulator using our strain limiting method as a velocity filter.*

Our strain-limiting method operates in the context of linear-basis triangular and tetrahedral elements. These were originally introduced to the graphics community by O'Brien and Hodgins [1999] and have since been widely adopted. (For example, [Teran et al., 2003; Etzmuβ et al., 2003; Irving et al., 2004; Chentanez et al., 2009; Thomaszewski et al., 2009].)

Provot [1995] introduced strain limiting for cloth simulations based on spring-and-mass simulations. The approach allows stiff behaviors without the drawbacks of stiff constitutive models. Desbrun and colleagues [1999] demonstrated that strain limiting can prevent a mass-spring cloth from being over-stretched in an interactive environment. Instead of directly enforcing constraints on positions, Bridson and collaborators [2003] treat the limiting method as a velocity filter so that the positions will be implicitly constrained after a position update. Müller [2008] extended the approach by developing a multi-resolution technique for strain limiting with spring-and-mass systems. Otaduy and his colleagues [2007] proposed a similar multi-resolution approach for meshes to handle boundary conditions, such as collisions.

A method of continuum-based strain limiting for triangle elements was proposed by Thomaszewski and colleagues [2009] to simulate cloth. Their method enforced limits on individual components of the strain tensor. While they achieved good results for examples where the rectilinear parameterization was aligned with the cloth's warp and weft directions, the method cannot model isotropic materials because the components of a tensor are not invariant with respect to coordinate transformations. For anisotropic cloth where the coordinate axes are aligned with the warp and weft directions of the material this property may be considered a feature, but it is a severe limitation in more general contexts. Further, even when anisotropic cloth is desired, independent limits on strain components offer only incomplete control over material behavior. Their enforcement step also required inverting a 6×6 matrix system for each element that exceeded its strain limits. This cost is somewhat acceptable for triangles and they do some of the work during a precomputation phase, but extending to tetrahedra would presumably require inverting a 12×12 matrix, which would likely be prohibitively expensive.

Similar to the method we propose here, Tsiknis [2006] also used singular value decomposition to limit principal strains. However, his formulation only allows one principal direction to be limited per step. If strain limits are violated in more than one principal di-

rection, the element needs to be processed multiple times to push it back into the limit. Instead, our strain-limiting method processes all principal directions simultaneously and the vertex update is directly calculated as matrix products.

Choi and his colleagues [2004] proposed an implicit constraint method to enforce bending angle constraints in cloth simulation. This method can animate cloth wrinkles with very large time steps, but it only handles a limited number of constraints (500 in their experiments). The constrained Lagrangian method was used by Goldenthal and his colleagues [2007] and also by English and Bridson [2008] to strictly enforce edge length constraints in springs and nonconforming elements using Lagrange multipliers. Assuming that the thin shell is inextensible, an implicit cubic method was proposed by Garg and his collaborators [2007] to efficiently generate stiff hinge-based bending behaviors with large time steps.

Constraints have also been used by Irving and colleagues in the context of tetrahedral finite elements to enforce volume preservation [2007]. They observed that strict volume preservation on a per-element basis can lead to undesirable locking behavior. To avoid this problem they enforce volume preservation collectively in the one-ring of elements surrounding each node. Our constraints do not trigger locking behavior because the method includes some amount of compliant deformation prior to the hard limit on strain. If exact volume preservation is desired, their one-ring volume constraints are compatible with our method.

3 Strain and Bending Constraints

Hard limits on the strain within an element or on the bending angle between two triangular elements can be expressed as constraints on the node positions. We formulate the constraints on strain in a coordinate independent fashion and show how they can be enforced individually by simple scaling of the offending element. Similarly bending constraints can be individually enforced also with local adjustments. These local adjustments can then be used as part of an iterative solver that enforces constraints globally.

3.1 Strain Limiting

Length constraints in a mass-spring system (for example [Provot, 1995]) are essentially one-dimensional strain limits. Given a spring connecting two end points $\mathbf{x_a}$ and $\mathbf{x_b}$, the length constraint is defined as:

$$s_{\min} \leq |\mathbf{x_a} - \mathbf{x_b}|/l_0 \leq s_{\max} \tag{1}$$

in which l_0 is the original spring length, and $s_{\min}$ and $s_{\max}$ are lower and upper bounds of the stretching/compression ratio. Alternatively, if we define the strain as:

$$e = (|\mathbf{x_a} - \mathbf{x_b}| - l_0)/l_0 \tag{2}$$

then $e+1$ is the compression ratio and the constraint clamps e to the range $[s_{\min} - 1, s_{\max} - 1]$. To enforce this constraint, we can either move both $\mathbf{x_a}$ and $\mathbf{x_b}$ along the spring, or just move one and let the other stay fixed if required by boundary conditions.

The goal of strain limiting involves preventing a triangle or tetrahedron from undergoing large deformations. Here we will mainly describe the solution to this problem for triangulated surfaces. The volumetric case can be derived in the same fashion and it will be briefly discussed later.

3.1.1 Notation and Background

Given a triangle with vertex positions $(\mathbf{x_0}, \mathbf{x_1}, \mathbf{x_2})$ in world space and $(\mathbf{r_0}, \mathbf{r_1}, \mathbf{r_2})$ in its reference configuration, we define $\mathbf{d_{1x}} = \mathbf{x_1} - \mathbf{x_0}$, $\mathbf{d_{2x}} = \mathbf{x_2} - \mathbf{x_0}$, $\mathbf{d_{1r}} = \mathbf{r_1} - \mathbf{r_0}$, and $\mathbf{d_{2r}} = \mathbf{r_2} - \mathbf{r_0}$ as triangle edge vectors after and before deformation. We then build a 2×2 matrix $\mathbf{D_x}$ with $\mathbf{d_{1x}}$ and $\mathbf{d_{2x}}$ as columns, and $\mathbf{D_r}$ with $\mathbf{d_{1r}}$ and $\mathbf{d_{2r}}$ as columns. The deformation gradient $\mathbf{F}$ is calculated using: $\mathbf{F} = \mathbf{D_x}\mathbf{D_r^{-1}}$. We diagonalize $\mathbf{F}$ into $\mathbf{F} = \mathbf{USV}^\mathsf{T}$ using Singular Value Decomposition (SVD), in which $\mathbf{U}$ and $\mathbf{V}$ are orthonormal matrices and $\mathbf{S}$ is a nonnegative diagonal matrix.

The diagonal entries s_0 and s_1 in $\mathbf{S}$ can be considered as compression ratios in two principal directions, invariant to rigid transformations and selection of coordinate system. They are also related to the principal strains (eigenvalues) of Green's strain tensor, $\mathbf{G} = (\mathbf{F}^\mathsf{T}\mathbf{F} - \mathbf{I})/2$, and approximately equal to the principal strains of Cauchy's strain tensor, $\mathbf{C} = (\mathbf{F}^\mathsf{T} + \mathbf{F})/\mathbf{2} - \mathbf{I}$. Thus any arbitrary deformation of the element can be represented as the result of a set of orthogonal stretches and compressions that correspond to scaling in the orthogonal directions of the principal strains. So called shearing deformation is merely an artifact of viewing these principal strains from a coordinate system that is not aligned with them. By working in the coordinate system of the principal strains, our limiting method only needs to enforce a set of orthogonal scaling constraints and does not need to solve a linear system for each strain component in an arbitrary coordinate system.

3.1.2 Two-Dimensional Strain Limiting

The basic concept behind our strain-limiting model is to reduce the two-dimensional strain-limiting problem into a combination of two one-dimensional problems, so that only two orthogonal length constraints need to be enforced in the principal directions:

$$\begin{aligned} s_0^* &= \text{clamp}\ (s_0, s_{\min}, s_{\max}) \\ s_1^* &= \text{clamp}\ (s_1, s_{\min}, s_{\max}) \end{aligned} \tag{3}$$

where $s_{\min}$ and $s_{\max}$ are the limits for isotropic scaling. For example, a material that is limited to compress/stretch $[-5\%, 5\%]$ would have $s_{\min} = 0.95$ and $s_{\max} = 1.05$. If anisotropic limiting were desired then we believe the s_i could be clamped in a direction-dependent fashion, however we have not yet demonstrated this type of behavior.

After s_0 and s_1 have been clamped, we construct a new diagonal matrix using $\mathbf{S}^* = diag(s_0^*, s_1^*)$, and we calculate the new deformation gradient as $\mathbf{F}^* = \mathbf{US}^*\mathbf{V}^\mathsf{T}$ and the new edge vector matrix as $\mathbf{D_x^*} = \mathbf{F}^*\mathbf{D_r}$. Without boundary conditions, we assume that the mass center is fixed so that the three vertex locations are updated as:

$$\begin{aligned} \mathbf{x_0^*} &= \mathbf{c_x} - \left(m_1\mathbf{d_{1x}^*} + m_2\mathbf{d_{2x}^*}\right) \Big/ (m_0 + m_1 + m_2) \\ \mathbf{x_1^*} &= \mathbf{x_0^*} + \mathbf{d_{1x}^*} \\ \mathbf{x_2^*} &= \mathbf{x_0^*} + \mathbf{d_{2x}^*}, \end{aligned} \tag{4}$$

where m_0, m_1 and m_2 are masses of the three vertices and $\mathbf{c_x}$ is the triangle's center of mass. If any single vertex is fixed as a boundary condition, then the scaling is done with that vertex as the origin. If two vertices are fixed then the scaling is about the line formed by the constrained vertices restricted to be orthogonal to that line.

3.1.3 Three-Dimensional Strain Limiting

Limiting the strain in a tetrahedron is done in roughly the same fashion as for triangles. Because a tetrahedron has four vertices, there are now three edge vectors, and the matrices $\mathbf{D_x}$, $\mathbf{D_r}$ and $\mathbf{F}$ will all be 3×3. SVD will produce three singular values, corresponding to three principal directions. After clamping the singular values, four vertices are updated from matrix products, and shifted to achieve linear momentum conservation.

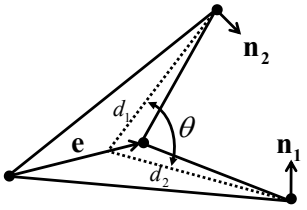

Figure 3: *The dihedral angle between two neighboring triangles.*

3.1.4 Bend Limiting

One problem that a surface embedded in three dimensions does not usually encounter but that can occur for tetrahedra is element inversion. Singular value decomposition guarantees that all the s_i are nonnegative and they are correct in representing the compression ratio most of the time. However, when the element is inverted due to over-compression, they are no longer correct because either one or three of s_i should be negative. Here we adopt the same strategy used by Irving and colleagues [2004]: if a tetrahedron becomes inverted (i.e., $\det \mathbf{F} < 0$), we simply negate the smallest singular value and keep $\mathbf{U}$ and $\mathbf{V}$ as rotations.

3.1.5 Discussion

Because this method is independent of the underlying coordinate system, the deformation gradient can be computed in any convenient coordinates that need not be the same for different elements. In practice, we simply use a two-dimensional local coordinate system for each triangle, whose first edge vector defines the $\hat{x}$ axis.

Compared with the component-based Cauchy strain-limiting method [Thomaszewski et al., 2009], our method is automatically rotationally invariant and coordinate independent. Therefore, it does not require a global surface parametrization and is easy to apply to curved meshes.

Furthermore, our method allows isotropic material behavior which cannot be achieved using component limits. Consider a rectangular piece of cloth that is parameterized so that the $\hat{x}$ and $\hat{y}$ axes are respectively aligned with the horizontal and vertical directions, and where we wish to have a material that can be stretched by at most twice its original length. For horizontal stretching, this limit will be reached when Cauchy's strain, $\mathbf{C}$, is $\mathrm{diag}(1, 0)$, implying that a suitable limit for C_x is 1.0. However a $2x$ stretch at 45° produces $\mathbf{C}$ with all four entries equal to 0.5, contradictorily implying that the limit for C_x should be 0.5. Using strain-component limits for an isotropic material is equivalent to attempting to constrain a point to be inside a sphere using independent limits on the point's $\hat{x}$, $\hat{y}$, and $\hat{z}$ coordinates.

Additionally, the computational cost of our method is less than the previous approach based on the components of the Cauchy strain tensor which required solving a 6×6 linear system for each triangular element, and would presumably require solving a 12×12 system for tetrahedral elements. Our method does require computing the SVD of a 2×2 or 3×3 system for each element, however these can be computed efficiently and will often already be available if a corotational or invertible-element scheme is in use. Each update is straightforwardly calculated as matrix products. We can also avoid many of the SVD operations because the singular values of $\mathbf{F}$ are square roots of the eigenvalues of $\mathbf{F}^\mathsf{T}\mathbf{F}$, which are the roots of a quadratic or cubic equation. Therefore, a triangle or tetrahedron can be skipped without further processing if no root exists within the limiting range. In our experience this test allows 40% to 50% of elements to be skipped in each enforcement step.

When simulating thin shells using bending energies based on dihedral angles across edges [Grinspun et al., 2003; Bridson et al., 2003], we can also include constraints to prevent the dihedral angle between two triangles from becoming too large or too small. To enforce the bending angle limit, we first calculate the dihedral angle $\theta = \pi - \mathrm{atan2}((\mathbf{n_1} \times \mathbf{n_2}) \cdot \mathbf{e}, \mathbf{n_1} \cdot \mathbf{n_2})$, where $\mathbf{e}$ is the normalized edge vector, and $\mathbf{n_1}$ and $\mathbf{n_2}$ are the two unit-length neighboring triangle normals as in Figure 3. We then set:

$$\theta^* = \mathrm{clamp}\,(\theta, \theta_0 - (d_1 + d_2)\theta_{\min}, \theta_0 + (d_1 + d_2)\theta_{\max}) \qquad (5)$$

where θ_0 is the resting angle, $\theta_{\min}$ and $\theta_{\max}$ are the mesh-independent bending angle limiting coefficients (degrees per unit length). To enforce the angle constraint while preserving angular momentum, we respectively rotate the two triangles along the axis $\mathbf{e}$ by amounts:

$$\begin{aligned} \theta_1^* &= m_2 d_2^2\,(\theta^* - \theta) \Big/ \left(m_1 d_1^2 + m_2 d_2^2\right) \\ \theta_2^* &= m_1 d_1^2\,(\theta - \theta^*) \Big/ \left(m_1 d_1^2 + m_2 d_2^2\right). \end{aligned} \qquad (6)$$

This rotation would shift the center of mass of the triangle pair. The shift could be corrected for by applying a compensating translation, but we have found that doing so may cause jittering and prevent convergence. The problem arises because angle constraint violations tend to occur in clustered groups where the surface is bending, and the average three-to-one ratio of edges to vertices combined with each edge constraint impacting four vertices results in many instances where multiple bending constraints try to move a single vertex in conflicting directions. To solve this problem, we calculate the compensating translation separately for each triangle. We first allow each bending angle constraint to only update the two vertices opposite that edge. We calculate all three vertex updates in the triangle according to three opposing edges using Equation 6, and then remove the center of mass translation of each triangle by a constant offset using Equation 4. In this way, each bending angle constraint only affects the two wing vertices directly, and the other two vertices along the edge are indirectly affected by the average influence of the surrounding neighborhood. Our experience shows that enforcing linear momentum conservation in this way eliminates the jittering and convergence problems.

3.2 Global Enforcement

The strain limit for a single triangle or the bending angle limit between two neighboring triangles can be reached instantly in one step using the methods described in Sections 3.1 and 3.1.4. However, enforcing limits over an entire mesh results in a system of nonlinear constraints that must be solved iteratively using a variation of either Jacobi's method or the Gauss-Seidel method. Similar to Jacobi's method for solving a linear system, the nonlinear version simultaneously calculates displacements to satisfy each individual constraint in isolation, and then simultaneously updates the position for each vertex using the weighted average of the displacements for that vertex. In contrast, the nonlinear Gauss-Seidel method immediately updates vertex positions as the displacements are computed for each constraint, and it is affected by traversal order. Thomaszewski and his colleagues [2009] provide a detailed discussion comparing both methods. We have found that a randomized Gauss-Seidel method works well.

We process boundary conditions simultaneously with strain limiting so that enforcing one set of constraints does not cause the other to be violated.

Solid Object Collision — When a vertex is moved we check if it has penetrated a solid object. If it has, the vertex will be projected back to the solid surface and its velocity adjusted according to an

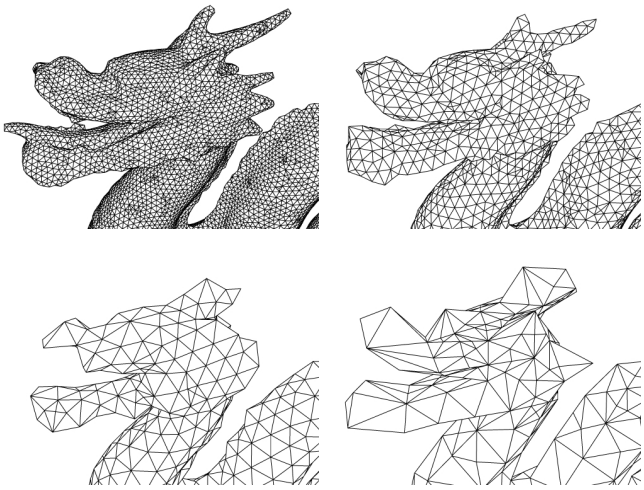

Figure 4: *Partial surfaces of a tetrahedral mesh hierarchy for a solid dragon. From fine to coarse, the meshes contain 74.1K, 16.2K, 4.2K and 2.3K tetrahedra.*

elastic collision model. For simplicity, we do not detect edge-solid collisions.

Self Collision — To reduce the cost of self-collision detection we only check for them once every 64 iterations. We use a continuous collision detection algorithm that tests for both vertex-triangle and edge-edge collisions. Each collision test is formulated as a cubic function and treated as a hard constraint [Bridson et al., 2003].

Friction — When collisions are detected, we use the penetration depth to generate dynamic friction forces on each penetrating vertex:

$$\mathbf{f_f}(\mathbf{x}) = -\mu d(\mathbf{x})\mathbf{v}(\mathbf{x})/|\mathbf{v}(\mathbf{x})| \tag{7}$$

in which $\mathbf{f_f}(\mathbf{x})$ is the friction force at $\mathbf{x}$, μ is the friction coefficient, d is the penetration depth and $\mathbf{v}$ is the relative tangential velocity at $\mathbf{x}$. However, if the magnitude of $\mathbf{v}(\mathbf{x})$ is already less than $\mu d(\mathbf{x})$, we simply set the relative tangential velocity to zero as an effect of static friction.

4 A Multi-Resolution Approach

Although the nonlinear iterative solver described above works well for small systems of up to a couple thousand elements, larger systems experience the same slow convergence behavior that occurs with the linear version of these methods. To deal with this problem we describe a multi-resolution approach to enforce constraints in large meshes.

4.1 Constructing the Mesh Hierarchy

We construct the triangle mesh for each hierarchical level directly from the base mesh. Given the desired number of triangles at each level, we uniformly re-triangulate the mesh and then apply edge flipping optimization using a Delaunay metric. If uniform sampling causes too much detail loss in coarse meshes, we switch to QSlim [Garland and Heckbert, 1997] instead, and then flip edges. To build a tetrahedral mesh hierarchy, we first create a surface mesh hierarchy, use TetGen [Si, 2010] to tessellate the interior with tetrahedra, and then improve the result using Stellar [Klingner and Shewchuk, 2007]. A mesh hierarchy constructed in this way is shown in Figure 4.

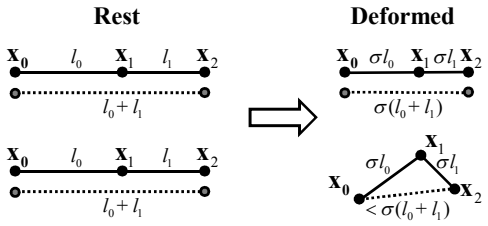

Figure 5: *The solid and dashed lines respectively represent elements in a fine and coarse mesh. Bending in the fine mesh appears as compression at coarser levels, requiring a looser compressive limit for the coarse mesh. Similarly, a bent rest configuration for the fine mesh will require looser limits on the coarse mesh for expansion.*

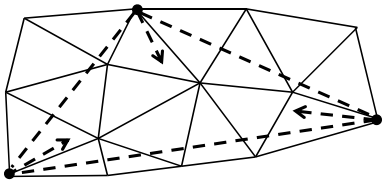

Figure 6: *The solid lines show a neighborhood in the base mesh H_0. The dashed lines show a triangle T in H_i that covers a portion of this neighborhood. To find the strain bounds for T, the algorithm deforms T while enforcing existing limits on the covered portion of the base mesh.*

To define downsampling and upsampling functions between two levels of the mesh hierarchy, H_i and H_j, we find for each vertex in H_j its closest element in H_i and the corresponding barycentric coordinates. A scalar/vector field defined over H_i can then be upsampled to H_j using linear interpolation. Downsampling uses these same weights for averaging to obtain values on the coarse mesh.

4.2 Generating Limiting Parameters

When simulating a solid body using a tetrahedral mesh in three dimensions, isotropic strain-limiting parameters are invariant to mesh triangulation. If each element can be stretched or compressed by at most $x\%$, a larger element as the combination of small elements can be stretched or compressed by $x\%$ at most as well. Therefore, we simply use the same strain-limiting parameters to simulate tetrahedral meshes at all hierarchical levels.

Unfortunately, limiting parameters for thin shells are not invariant to mesh triangulation because fine-scale out-of-plain buckling appears as compression at coarse scales. A one-dimensional illustration of this phenomenon appears in Figure 5. Without carefully choosing limiting parameters at each level, the result can be overly stiff or loose, as Muller [2008] demonstrated. Here we propose a training algorithm to automatically generate limiting parameters from the fine base mesh for each hierarchical level. Let H_0 be the original base mesh and H_i be the constructed coarser mesh at level i. Our goal is to find a set of tight limiting parameters over H_i such that they are consistent with limiting parameters over H_0. By consistency, we mean any deformation satisfying limiting parameters in H_0 should satisfy parameters in H_i as well.

Example	Mesh Type	# of Elements	# of Levels	Iterations / level	Time / frame
Bunny	Tri.	58.8K	4	256	16.56s
Cloth	Tri.	12.8K	4	128	1.80s
Shirt	Tri.	40.0K	4	256	10.56s
Duck	Tri.	4.1K	2	128	0.38s
Dragon	Tri.	37.4K	4	200	8.28s
Dragon	Tet.	74.1K	4	256	16.67s
Tori	Tet.	51.9K	3	256	13.11s

Table 1: *Timing tests were run as a single process on a Dell T7500 workstation with a dual quad core 2.26 GHz processor. The same number of iterations was used at each level of a given hierarchy. The time step used in all examples is 1/30 second. The average timing for each frame is counted without collision detection. Collision detection typically takes 0.5 to 6 seconds, depending on the mesh resolution and the amount of interpenetration.*

Given a base mesh at level H_0 (shown as the solid lines in Figure 6) and all its known limits, the question is how much a triangle T in H_i (the dashed triangle in Figure 6) is allowed to deform without violating limits in H_0. To find out this, we first slightly shrink T in H_i as three arrows show, update the three corresponding vertices in H_0, then enforce all the limits in H_0, and send updated vertex positions back to T in H_i. After several iterations (100 to 400), T will converge to its minimum shape when all the limits in H_0 are still satisfied. In practice, we only enforce those limits in H_0 that are close to T for efficiency. The upper bound (how much T can expand) can be found in the same way by gradually expanding T. Bending angle limits are also calculated by increasing or decreasing the dihedral angle while enforcing nearby limits in H_0.

Instead of calculating the limits at each level directly from H_0, an indirect method is to calculate the limits at H_i using limits at H_{i-1} in a hierarchical fashion. Compared with the direct method, the indirect method is more efficient because a triangle T in H_i covers fewer elements in H_{i-1} than in H_0. However, limiting parameters generated indirectly may not be tight enough, because strain limits at H_{i-1} cannot represent all constraints in H_0. As a result, they affect the convergence rate when being used to enforce limits.

This process of learning limits for each element and edge in the hierarchy (excluding the fine base mesh) can be time consuming, but the process only needs to be done once for each mesh and setting of its limit parameters. For our examples the direct method took between 12 to 20 hours to calculate, and the indirect method 6 to 12 hours.

4.3 A Multi-Resolution Solver

Using the downsampling and upsampling functions defined in Section 4.1, our multi-resolution limiting system works like other multi-resolution systems as Figure 2 shows. Velocities are first downsampled to the top hierarchical level, constrained with strain limits and bending angle limits, upsampled to the next level, constrained again, until velocities pass through all the levels. What makes the system slightly more complicated are the boundary conditions. We will discuss how to incorporate typical boundary conditions into this system next.

When processing a coarse mesh at some hierarchical level, boundary conditions can be formulated either directly from the coarse mesh or indirectly from the base mesh. Ideally they should be obtained by upsampling from the base mesh because the final result will be represented in the base mesh. In practice, we select different ways for boundary conditions, according to their computational cost. Collisions with the environment are formulated directly from the base mesh as they are easy to calculate. This allows the limiting method to capture interactions between small features and the environment even at a coarse level. One such example is the collision between the dragon horn and the cylinder roller in Figure 11. On the other hand, self collisions are expensive to compute in the base mesh and their accuracy is less important as long as no self penetration is observed. So we detect them using coarse meshes directly and they are only tested once every 64 iterations as described in Section 3.2. Finally, friction is applied to each vertex's velocity using Equation 7 once a collision is detected. Figure 10 shows that all three types of boundary conditions are stably handled with a large time step.

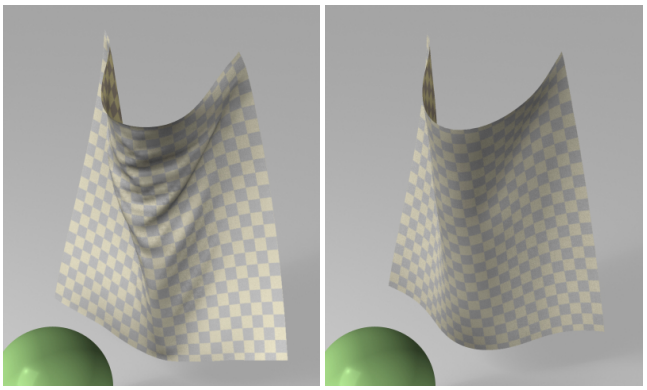

Figure 7: *A pair of cloth sheets showing the effect of bending limits on wrinkle formation. Loose bending angle limits* $[-60^{\circ/\mathrm{cm}}, 60^{\circ/\mathrm{cm}}]$ *(left) allow small wrinkles, while tighter limits* $[-30^{\circ/\mathrm{cm}}, 30^{\circ/\mathrm{cm}}]$ *(right) prevent small wrinkles from forming.*

5 Results and Discussion

We implemented a triangle-based simulator for thin shells and a tetrahedral simulator for solid bodies. Both systems use linear-basis finite elements with nonlinear Green's strain and a linear constitutive law [O'Brien and Hodgins, 1999]. For triangulated surfaces, bending forces are calculated using the method described by Bridson and his colleagues [2003]. Our strain-limiting process works with this basic finite element simulation as a velocity filter.

5.1 Examples

We tested a number of different examples, with animations shown in the accompanying video. Timing results are reported in Table 1. The maximum number of iterations per level is listed in Table 1 and was chosen so that the average residual strain of violating triangles converged to between 0.1% and 1%, and the average residual angle of violating edges to between $0.5^{\circ/\mathrm{cm}}$ and $1.2^{\circ/\mathrm{cm}}$. The iterations terminated when a small convergence threshold or the maximum iteration number was reached.

The bunny example in Figure 1 demonstrates behaviors on a thin shell using different limiting parameters. Similarly, the cloth example shown in Figure 7 illustrates that loose bending angle limits can maintain small wrinkles over the cloth, while tighter limits prevent small wrinkles from forming. The shirt example in Figure 8 shows cloth on a virtual character and demonstrates highly detailed wrinkles.

Figure 8: *A simulated shirt on a virtual character with strain limits* $[-1\%, 1\%]$ *and bending angle limits* $[-60^{\circ/\text{cm}}, 60^{\circ/\text{cm}}]$.

The duck example shown in Figure 9 is a hollow triangulated mesh. The figure compares results from our multi-resolution strain-limiting method with non-hierarchical strain limiting and a standard finite-element result using stiff material parameters. The result from the hierarchical method matches the non-hierarchical result but is substantially cheaper to compute.

The dragon example in Figure 11 demonstrates the system performance on large meshes, and also the dynamic difference between a thin shell and a solid body. The tori example in Figure 10 is used to show that boundary conditions work well with a large time step (1/30 second) even when the tori are moving rapidly.

The computational cost is mostly determined by the average number of elements violating the limits and the number of iterations. Therefore, the timing varies in each frame over the whole sequence, depending on how over-constrained the scene is. The timings listed in Table 1 are averages taken over just the active part of the simulation — for example, when the roller is in contact with the dragon head in Figure 11. The average timings for the whole sequences are actually smaller.

5.2 Comparisons

Figure 9 shows a hollow duck mesh falling off a ramp, simulated in four different ways. Our multi-resolution strain limiting method (Figure 9.a) simulates each (30Hz) frame in 0.375 seconds. Without using the multi-resolution approach, the basic strain-limiting method (Figure 9.c) requires many more iterations in order to achieve similar stiff behaviors — using the same amount of computational time (number of iterations) cannot reproduce the correct stiff behavior (Figure 9.b). This makes the basic strain-limiting method even slower than a basic FEM simulator with 100 sub-steps per time step (Figure 9d).

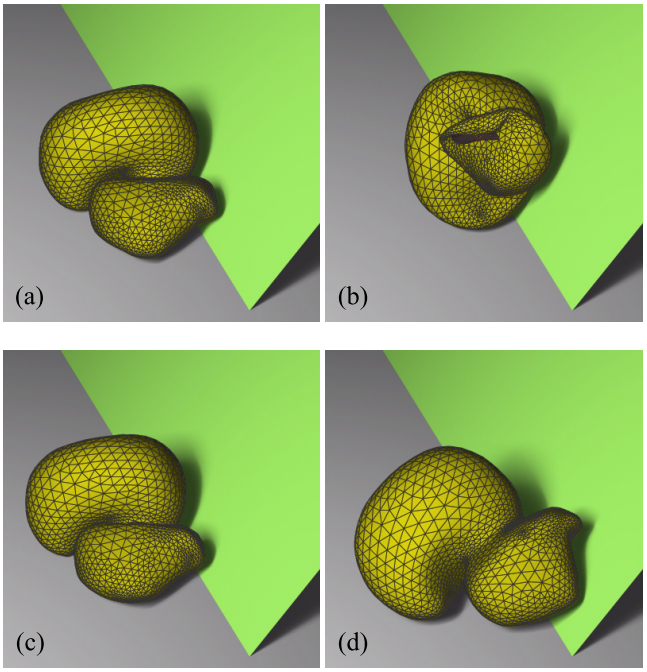

Figure 9: *A hollow duck simulated in four different ways (with no self collision detection). Our proposed multi-resolution strain-limiting method (a) simulates each (30Hz) frame with 128 iterations in 0.375 seconds. Also using 128 iterations, the non-hierarchical strain-limiting method (b) simulates each frame in 0.359 seconds, but the result loses its stiff behavior. By using 2048 iterations, the basic strain-limiting method (c) slowly (5.875 seconds) generates stiff results again. Finally, a basic finite-element simulator (d) produces a similar result at 0.734 seconds per frame by using 100 substeps in each frame. The limit parameters are* $[-5\%, 5\%]$ *and* $[-3^{\circ/\text{cm}}, 3^{\circ/\text{cm}}]$.

To compare the performance of our multi-resolution system with other constraint-based methods we implemented a one-dimensional chain for testing purposes using MATLAB. Results from these tests are shown in Figure 12. The chain contained 128 edge constraints, which approximates one-dimensional scaling of a two-dimensional mesh with $129 \times 129 = 16641$ vertices. We compared different techniques for projecting the chain back to its resting length by setting strain limits to zero. Figure 12 (top) plots the computational cost of each method at different termination thresholds. We tested the multi-resolution method with either 4, 8 or 16 vertices at the very top level of the mesh hierarchy. Figure 12 shows that although an ideal multi-resolution method performs the best in one dimension, with 16 vertices at the top level it still performs slightly better than the (fast projection) constrained Lagrangian method proposed in [Goldenthal et al., 2007], especially if the threshold is loose. Another advantage of the multi-resolution method is its scalability to high-resolution meshes, as shown in Figure 12 (bottom). Both two- and three-dimensional cases are more complicated than one-dimension, so Figure 12 only gives a rough picture of how the system may behave with different global enforcement methods. It should also be noted that although we used the Gauss-Seidel method in our current multi-resolution system, it is not the only choice for global enforcement. At this time, we are not clear how to use the constrained Lagrangian method for strain limiting and bending angle limiting, because the matrix system formulated from constraint gradients will be much less sparse. But once an efficient

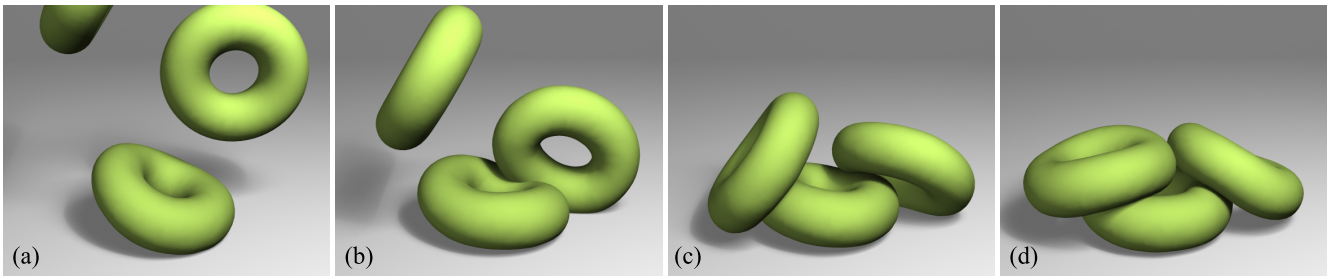

Figure 10: *A set of three solid tori being dropped onto each other. Limit parameters are* $[-5\%, 5\%]$ *strain.*

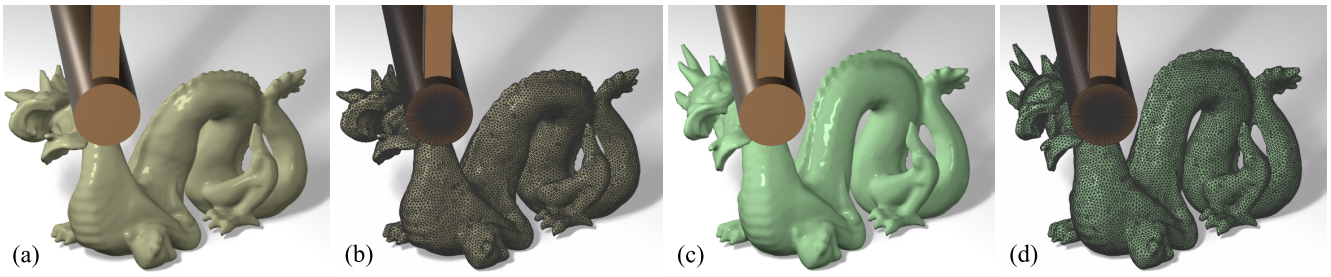

Figure 11: *A hollow dragon mesh (a) behaves differently from a solid dragon mesh (c) when pressed by a cylindrical roller. The underlying meshes are shown in (b) and (d). Limit parameters are* $[-5\%, 5\%]$ *strain for both hollow and solid, and* $[-3^{\circ/\mathrm{cm}}, 3^{\circ/\mathrm{cm}}]$ *bending for hollow.*

solution is found in the future, the constrained Lagrangian method can be incorporated into our system as well to further improve the performance.

Figure 14 compares the limiting behavior applied to a square sheet of cloth that starts in a randomly perturbed shape (Figure 14.a), without (top) and with (bottom) our multi-resolution approach. The color pattern on the larger images visualizes the residual strain over the cloth using the HSV color model: the hue channel encodes the maximum principal strain direction, the value channel shows the magnitude of the maximum principal residual strain past the limit, and the saturation channel encodes the ratio of the two principle strains. The images and residual plot (lower left) show that the multi-resolution approach converges significantly faster than the fine-only approach. The reason can be seen in the color plots from how the non-hierarchical method quickly removes high-frequency errors but then becomes very inefficient at eliminating low-frequency errors. The multi-resolution approach operates at different scales and does not experience that difficulty. The multi-resolution method reaches an average of 1% residual strain per triangle in 0.510 seconds after 128 iterations at each level (512 iterations total). In contrast, the fine-only approach needs at least 2000 iterations (7.680 seconds) to achieve the same error. Figure 14 compares both when restricted to a total of 512 iterations, and the multi-resolution approach with smaller error is still four times faster than the fine-only approach, because the computational cost at the coarse level (0.005 seconds for the first 128 iterations) is much smaller than the cost at the finest level (0.446 seconds for the first 128 iterations).

5.3 Discussion and Limitations

Because strain limiting prevents large deformations, our experience shows that even relatively stiff constitutive models can be used in a finite-element simulation without instability issues. However, using excessively large or small stiffness coefficients can still cause artifacts in simulations. Large coefficients may cause unnatural jittering on the surface, while small coefficients may cause over-damping artifacts due to the energy loss.

When the system is over constrained by boundary conditions, there may exist no solution satisfying all of the limiting constraints. Instead, the system will halt when the iteration limit is reached, ideally with an evenly distributed, minimal error. For instance, the tetrahedral dragon mesh cannot satisfy the strain limit $[-5\%, 5\%]$ without violating the boundary conditions (its bottom is attached to the ground while the roller squeezes its body). The actual strain bound reached in this case is around -60%.

As mentioned previously, one potential artifact that may occur from the strain-limiting method is surface jittering. Aside from the four described reasons (conflicts in bending angle constraints, bad mesh quality, large stiffness coefficients, and inverted elements), another possible cause is the stiffness discontinuity in biphasic materials implied by the strain-limiting model. We have not encountered difficulties from this problem, however if the issue arose a potential solution would be to replace the Gauss-Seidel iterations with nonlinear successive over-relaxation or the constrained Lagrangian method.

In a multi-resolution system, the limiting work required at each level is determined by the amount of remaining deformations not covered by higher levels. Because the deformation field is usually smooth, these remaining deformations are small in most cases and the multi-resolution structure improves the convergence speed of the whole system. However, it can be less effective in some special cases, such as the twisting example shown in Figure 13.

6 Conclusion and Future Work

We have described a fast, isotropic, multi-resolution strain-limiting approach to simulate stiff, incompliant materials for thin shell and

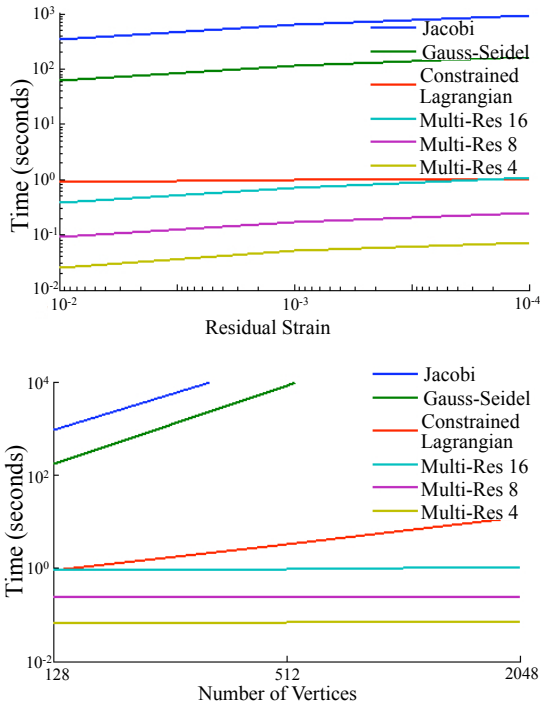

Figure 12: *Performance of several constraint enforcement techniques for a one-dimensional chain example.*

solid elastic body simulations. The strain limit is efficiently enforced in a coordinate-invariant fashion based on elements' principal strains. We also provide a bending angle limiting method to resist out-of-plane bending. The system includes a novel multi-resolution algorithm that consistently applies tight constraints at each hierarchical level, so the system can quickly converge to match desired limits. In the future, we would like to improve the performance of the multi-resolution algorithm by using the constrained Lagrangian method to solve strain and bending angle limiting problems, by finding better solutions to twisting and other deformations that cannot be well handled in a multi-resolution fashion, by accelerating the system using parallel processing, and by experimenting with other multi-grid schemes to obtain faster convergence.

Acknowledgments

We thank the members of the Berkeley Graphics Group for their support and helpful suggestions. This work was supported in part by NSF Awards IIS-0924968, IIS-0915462, ONR YIP Award N00014-10-1-0032, California Discovery Grant COM09S-156646, UC Lab Fees Research Program Grant 09-LR-01-118889-OBRJ, and by generous support and equipment donations from Intel, NVIDIA, Pixar, Adobe, and Autodesk.

References

Bridson, R., Marino, S., and Fedkiw, R. 2003. Simulation of clothing with folds and wrinkles. In *Proc. of SCA 2003*, 28–36.

Chentanez, N., Alterovitz, R., Ritchie, D., Cho, L., Hauser, K. K., Goldberg, K., Shewchuk, J. R., and O'Brien, J. F. 2009. Interactive simulation of surgical needle insertion and steering. In *Proc. of ACM SIGGRAPH 2009*.

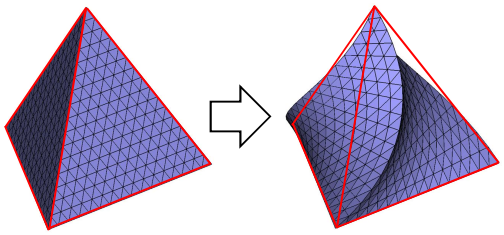

Figure 13: *A twisting case. The coarse tetrahedron (red) remains undeformed, while the base mesh is deformed under twisting motion. The multi-resolution approach is less effective in this case.*

Choi, M.-H., Hong, M., and Welch, S. 2004. Modeling and simulation of sharp creases. In *ACM SIGGRAPH 2004 Sketches*, 95.

Desbrun, M., Schröder, P., and Barr, A. 1999. Interactive animation of structured deformable objects. In *Proc. of Graphics interface 1999*, 1–8.

English, E., and Bridson, R. 2008. Animating developable surfaces using nonconforming elements. In *Proc. of ACM SIGGRAPH 2008*, vol. 27, 1–5.

Etzmuß, O., Keckeisen, M., and Straßer, W. 2003. A fast finite element solution for cloth modelling. In *Proc. of Pacific Graphics 2003*, 244.

Forsyth, C. J., and Simpson, J. 2008. Everything changes once you hang: Flesh hook suspension. *Deviant Behavior 29*, 4 (May), 367–387.

Garg, A., Grinspun, E., Wardetzky, M., and Zorin, D. 2007. Cubic shells. In *Proc. of SCA 2007*, 91–98.

Garland, M., and Heckbert, P. S. 1997. Surface simplification using quadric error metrics. In *Proc. of ACM SIGGRAPH 1997*, 209–216.

Gibson, S. F. F., and Mirtich, B. 1997. A survey of deformable modeling in computer graphics. Tech. rep., Mitsubishi Electric Research Laboratories.

Goldenthal, R., Harmon, D., Fattal, R., Bercovier, M., and Grinspun, E. 2007. Efficient Simulation of Inextensible Cloth. In *Proc. of ACM SIGGRAPH 2007*, vol. 26, 49.

Goodfellow, G. E. 1887. Notes on the impenetrability of silk to bullets. *Southern California Practitioner*.

Grinspun, E., Hirani, A. N., Desbrun, M., and Schröder, P. 2003. Discrete shells. In *Proc. of SCA 2003*, 62–67.

Irving, G., Teran, J., and Fedkiw, R. 2004. Invertible finite elements for robust simulation of large deformation. In *Proc. of SCA 2004*, 131–140.

Irving, G., Schroeder, C., and Fedkiw, R. 2007. Volume conserving finite element simulations of deformable models. In *Proc. of ACM SIGGRAPH 2007*, ACM, New York, NY, USA, vol. 26, 13.

Klingner, B. M., and Shewchuk, J. R. 2007. Agressive tetrahedral mesh improvement. In *Proceedings of the 16th International Meshing Roundtable*, 3–23.

Müller, M., and Gross, M. 2004. Interactive virtual materials. In *Proceedings of Graphics Interface 2004*, 239–246.

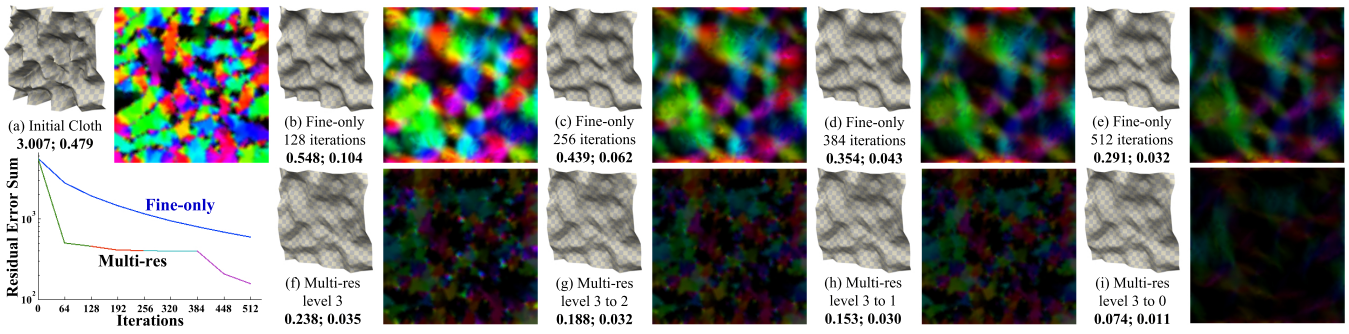

Figure 14: *A cloth square is initially perturbed with noise and these plots compare the convergence and smoothing behavior for our strain-limiting scheme without (top) and with (bottom) hierarchical enforcement. The small gray images show the sheet's geometric configuration and colored square plots strain over the sheet with hue indicating strain direction and intensity indicating strain magnitude. The two bold numbers under each rendered image are the maximum and the average residual strain. The plot in the lower left shows the overall convergence of the residual strain sum during 512 iterations.*

Müller, M. 2008. Hierarchical position based dynamics. In *Proc. of Virtual Reality Interactions and Physical Simulations*.

Nealen, A., Müller, M., Keiser, R., Boxerman, E., and Carlson, M. 2005. Physically based deformable models in computer graphics. Tech. rep., Eurogrphics 2005 state of the art report.

O'Brien, J. F., and Hodgins, J. K. 1999. Graphical modeling and animation of brittle fracture. In *Proc. of ACM SIGGRAPH 1999*, 137–146.

Otaduy, M. A., Germann, D., Redon, S., and Gross, M. 2007. Adaptive deformations with fast tight bounds. In *Proc. of SCA 2007*.

Provot, X. 1995. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface '95*, 147–154.

Si, H., 2010. A quality tetrahedral mesh generator and a 3d delaunay triangulator. http://tetgen.berlios.de/.

Teran, J., Blemker, S., Hing, V. N. T., and Fedkiw, R. 2003. Finite volume methods for the simulation of skeletal muscle. In *Proc. of SCA 2003*, 68–74.

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. 1987. Elastically deformable models. In *Proceedings of ACM SIGGRAPH 1987*, vol. 21, 205–214.

Thomaszewski, B., Pabst, S., and Strasser, W. 2009. Continuum-based strain limiting. In *Proc. of Eurographics 2009*, vol. 28, 569–576.

Tsiknis, K. D., 2006. Better cloth through unbiased strain limiting and physics-aware subdivision. Master thesis, The University of British Columbia.

Volino, P., Magnenat-Thalmann, N., and Faure, F. 2009. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph. 28*, 4, 1–16.