Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2014)
Vladlen Koltun and Eftychios Sifakis (Editors)

Steklov-Poincaré Skinning

Ming Gao University of Wisconsin-Madison
Nathan Mitchell University of Wisconsin-Madison
Eftychios Sifakis University of Wisconsin-Madison

Abstract

We introduce a novel and efficient simulation technique for generating physics-based skinning animations of skeleton-driven characters with full support for collision handling. Although physics-based approaches may use a volumetric (e.g. tetrahedral) flesh model, operations such as rendering, collision processing and user manipulation directly involve only the surface of this mesh. Motivated by this fact we define an elastic model of the skin surface which, while directly using only the surface degrees of freedom, exhibits a mechanical response that captures the full volumetric flesh behavior. We achieve this unusual result by combining three fundamental contributions: First, we present a material model which offers a plausible approximation to corotational elasticity at significantly reduced cost, by computing local rotations via procedural skinning rather than deriving them from the mesh deformation; the result is a force model which is affine on vertex positions, with coefficients dependent on the skeletal pose (but not on the deformation). Second, we use this force model to derive a direct mapping between surface vertex positions and resulting equilibrium forces on the same boundary vertices, which is a discrete version of the Steklov-Poincaré operator of the volumetric elastic model. This mapping is conveniently shown to also be affine (with pose-dependent coefficients), but with a dense stiffness matrix which renders direct numerical solution impractical. However, as a third and final step we show how a modified Newton iteration and a skinning-inspired preconditioner can solve the boundary problem with a competitive runtime cost. We assess the efficacy of our solution in simulations of high resolution human flesh models, with full external and self-collision processing.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1. Introduction

Skinning tools for articulated characters are indispensable components of a production pipeline, and artists today have access to a number of different techniques, whether they stress real-time performance [KCZO08], physical realism [MZS*11] or interactive collision modeling [VBG*13]. We present a different, and somewhat unconventional physics-based skinning alternative which can simulate *large* models of flesh-covered articulated characters with collision processing, at interactive or near-interactive rates. Compared to full-fledged finite element-based skinning techniques, our method makes just two simplifying assumptions, which are well tolerated in a majority of practical scenarios: First, we initially concentrate on quasistatic flesh simulation and forgo inertial or ballistic effects. Second, we employ a specific constitutive law which provides a highly plausible approximation to the familiar corotated elasticity model, but sidesteps the algebraic nonlinearity of the latter and its effect on solver convergence. Beyond that, our approach fully supports physics-based collision handling, both in cases of self-collision as well as flesh collisions with external objects.

Our key conceptual innovation stems from a simple observation: Physics-based skinning techniques will often utilize volumetric discretizations (e.g. tetrahedral/hexahedral meshes or embedded grid-based models) to capture behaviors such as volume conservation, pinching and bulging. However, even though the entirety of this volume is involved in determining the mechanics of deformation, in many practical aspects the flesh behavior is "experienced" via its boundary surface alone, i.e. the skin surface layer. It is exactly that layer that is used for rendering, manipulation by the user/artist, or even collision detection and processing. In light of this, it would be fortuitous if the deformable flesh could be modeled solely as an elastic *surface*, while minimizing (or eliminating, if possible) the computational effort spent on the interior of the flesh which is not directly rendered or manipulated. Simplified modeling approaches could achieve this goal, albeit with significant compromises in physical accuracy. For example, one could use soft springs to attach an elastic cloth model on the surface of a procedurally skinned flesh volume. Of course, in such a scenario the skin layer would not be able to couple its influence back to

DOI: 10.2312/sca.20141132

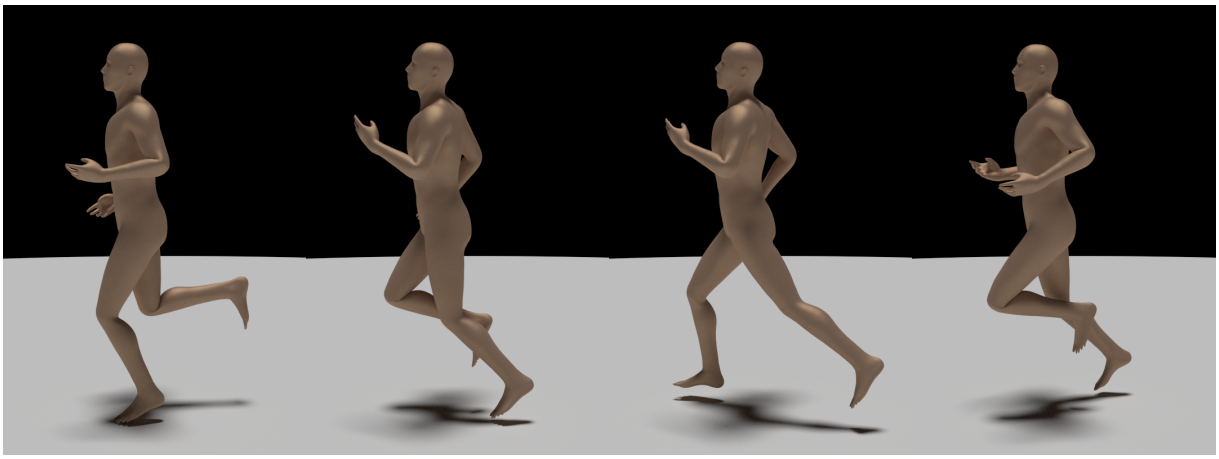

Figure 1: *Skinning simulation of a high-resolution human character, in a running sequence, with self-collision handling*

the volume, nor could it faithfully resolve volumetric behaviors such as collision-induced compression and bulging. In contrast, our approach replicates *exactly* what a quasistatic volumetric simulation would have produced, even though it only uses boundary vertices as degrees of freedom. From a technical standpoint, our volumetrically-reactive elastic surface model is a discretization of the construct known as the *Steklov-Poincaré operator* in the study of boundary-value PDE problems. This operator converts a given boundary condition (e.g. Dirichlet) to the values of a different type (e.g. Neumann) that would yield the same solution assuming any source terms remain unchanged. In our case, the PDE in question is that of 3D elasticity, and the mapping is from Dirichlet to traction boundary conditions or, discretely, from boundary displacements to boundary elastic forces.

At first glance, it may seem that our motivation for pursuing a surface-only elastic skin model has to do with reducing the number of degrees of freedom. Although there is truth in this statement, the cost-benefit analysis is more complex as the governing equations of our surface model are, numerically, more complex than the standard volumetric discretization. Our design decisions were even more influenced by the practical impact that collisions have on the efficiency of physics-based skinning techniques. When simulating high resolution nonlinear flesh models it is well documented (see, e.g. [MZS*11]) that collision support has a profound impact on runtime cost. Collision *detection* itself is typically a small fraction of the overall cost, for high-resolution models. The true cost stems from the collision-induced nonlinearity and its impact on the convergence of the Newton-Raphson iteration for the combined governing equations. When performing such iterations, we pay a volumetrically-scaling runtime cost to resolve what is primarily a defect localized on the surface (or fraction thereof). Using a surface-based elastic response helps defray the cost of such iterations, by avoiding misplaced emphasis on the interior of the flesh volume as contact and collision patterns are iterated to convergence.

Our specific algorithmic contributions are:

- We introduce the material model of *ex-rotated* (extrinsically rotated) elasticity, which uses a procedural skinning technique to approximate co-rotated elasticity with an affine force model with pose-dependent coefficients.
- We craft an equivalent formulation of the quasistatic equilibrium equations for a volumetric model which, conceptually, uses only surface degrees of freedom in its formulation. We demonstrate equivalence with the volumetric approach and compatibility with collision processing.
- We present a preconditioning technique and a modified Newton iteration that dramatically accelerate the surface-based elasticity formulation and circumvent the need for dense linear algebra in its implementation. We demonstrate the efficacy of our preconditioned treatment in simulations of high-resolution body motion with collisions.

2. Related work

Procedural skinning of a skeleton driven character was introduced in the form of Linear Blend Skinning [MTLT88]. Despite its efficiency, this technique has well known practical limitations which a number of authors have worked to alleviate [LCF00, LCF00, SRC01, MG03, WP02, KJP02]. These techniques offer significant quality improvements with competitive performance, but cannot avoid nonphysical deformations, especially in configurations involving collisions. Dual quaternion skinning [KCZO08] has been a very popular procedural approach, and further improvements such as the use of a high-quality weighting scheme [JBPS11] or precomputing weights to best approximate nonlinear model behavior [KS12] have been developed. Allowing nonrigid bone transformations has also been investigated [JS11].

For applications that require faithful resolution of nonlinear material behavior, finite element discretizations can capture intricate details of flesh deformations [LST09, SB12]. For some of the simplest nonlinear material models, such

as corotated elasticity [CPSS10], accelerated solutions have been demonstrated which optionally support collision handling [MZS*11] at near-interactive simulation times. Recent efforts have also focused on plausible contact modeling without simulation of nonlinear material deformation [VBG*13]. Our work has a strong link to finite element based tetrahedral flesh approaches [TSB*05, TSIF05] but employs a simplified approximation to corotational elasticity in the interest of performance. A number of authors have exploited the fact that skin deformation of skeleton-driven characters is dominated by the kinematics of the driving rig or skeleton: Capell and colleagues [CGC*02, CBC*05] adapted elasticity tensors to the local frame of reference of skeletal components. Hahn et al. [HTC*13] generated secondary skin dynamics based on the rig degrees of freedom. Kim and James [KJ12] combined locally-rotated nonlinear subspace models to simulate detailed deformations of complex models. Our preconditioning strategy uses local rotation information to approximate the stiffness matrix of our specific elastic model; Hecht et al. [HLSO12] describe a much more general preconditioning strategy based on delayed rotational updates. Finally, our concept of a surface-centric elastic model parallels the Schur Complement Method [QV99] for iterative substructuring, the Boundary Element Method leveraged by James and Pai [JP99] for real-time simulation of deformable models, and the process of static condensation for finite elements which has been utilized in the context of surgical simulation [BNC96].

3. Outline and background

In section 4 we propose a constitutive law specifically tuned for simulating flesh deformation driven by an articulated skeleton, which we label *ex-rotated elasticity*. This model leverages information from a procedural skinning method, and ultimately yields an affine force-position relationship:

$$\mathbf{f}(\mathbf{x};\mathbf{q}) = \mathbf{K}(\mathbf{q})\mathbf{x} + \mathbf{g}(\mathbf{q})$$

where the coefficients in $\mathbf{K}$ and $\mathbf{g}$ are only dependent on the skeletal pose $\mathbf{q}$, and not on the flesh deformation $\mathbf{x}$. We demonstrate that the behavior of this model is very similar to that of co-rotated elasticity, even when collisions are present. In section 5 we show that we can further derive a purely affine relationship that links boundary vertex positions ($\mathbf{x}_b$) directly to resulting boundary forces ($\mathbf{f}_b$), assuming that the interior remains at equilibrium at all times:

$$\mathbf{f}_b(\mathbf{x}_b;\mathbf{q}) = \mathbf{K}_b(\mathbf{q})\mathbf{x}_b + \mathbf{g}_b(\mathbf{q}).$$

We show that due to the reduced dimensionality of this formulation, iterative solvers converge in notably fewer iterations than its volumetric counterpart. However, a naive use of this system would be computationally impractical, since $\mathbf{K}_b$ is both (a) dense and (b) very expensive to compute. Section 5.1 alleviates these drawbacks by designing a very effective preconditioner, and replacing $\mathbf{K}_b$ with an excellent approximation that only requires sparse algebra. In the remainder of this section, we review certain theory and algorithmic concepts that our method builds on.

Tetrahedral FEM discretization of elasticity. We define discrete elastic forces on tetrahedral meshes using the standard finite element methodology, assuming linear elements and constant-strain tetrahedra. We refer the reader to the tutorial by Sifakis and Barbić [SB12] for the detailed derivation, while we restate just the final algorithm in this section.

The deformation of the flesh volume is encoded in the deformation function $\vec{x} = \vec{\phi}(\vec{X})$ which maps any material point with reference coordinates $\vec{X}$ to its deformed spatial location $\vec{x}$. Using a tetrahedral mesh to capture the flesh geometry, we sample the deformation map through the undeformed ($\vec{X}_i$) and deformed ($\vec{x}_i$) coordinates of every mesh vertex v_i, and barycentrically interpolate $\vec{\phi}(\vec{X})$ in each tetrahedron to yield a piecewise linear deformation field. The Jacobian $\mathbf{F} := \partial\vec{\phi}/\partial\vec{X}$ is called the *deformation gradient* matrix, and is piecewise constant as a consequence of the piecewise linear interpolation of $\vec{\phi}$. In a given tetrahedron with undeformed vertex positions $\vec{X}_0,\vec{X}_1,\vec{X}_2,\vec{X}_3$ and deformed coordinates $\vec{x}_0,\vec{x}_1,\vec{x}_2,\vec{x}_3$ the deformation gradient can be shown to equal $\mathbf{F} = \mathbf{D}_s\mathbf{D}_m^{-1}$, where $\mathbf{D}_s = [\vec{x}_1-\vec{x}_0|\vec{x}_2-\vec{x}_0|\vec{x}_3-\vec{x}_0]$ and $\mathbf{D}_m = [\vec{X}_1-\vec{X}_0|\vec{X}_2-\vec{X}_0|\vec{X}_3-\vec{X}_0]$ are called the deformed and undeformed *shape matrices*, respectively.

We use a hyperelastic material model, defined via an energy density function $\Psi(\mathbf{F})$ measuring the stored potential energy per unit volume as a function of the deformation gradient. The matrix $\mathbf{P} = \partial\Psi(\mathbf{F})/\partial\mathbf{F}$ is the *first Piola-Kirchhoff stress tensor*. A hyperelastic material model can be defined via a formula for either $\Psi(\mathbf{F})$ or $\mathbf{P}(\mathbf{F})$ since any of the two can be derived from the other. Given an algebraic expression for $\mathbf{P}(\mathbf{F})$ we can readily compute the elastic forces on three of the four tetrahedron vertices using the matrix equation

$$[\vec{f}_1|\vec{f}_2|\vec{f}_3] = -\tfrac{1}{6}\det(\mathbf{D}_m)\cdot\mathbf{P}(\mathbf{F})\cdot\mathbf{D}_m^{-T} \qquad (1)$$

while the last vertex receives a force $\vec{f}_0 = -\vec{f}_1-\vec{f}_2-\vec{f}_3$ to ensure balance of internal forces. The elastic forces on all vertices of the tetrahedral simulation mesh are computed by accumulating the contribution of each individual element.

Penalty-based collisions. Similar to other physics-based skinning techniques (see, e.g. [MZS*11]) we detect collision events on a discrete point set of surface *collision proxies* and respond to such instances using penalty forces. In our work, we simply use all surface vertices in the flesh mesh as collision proxies, although any set of points sampled from the surface of the flesh model can be used instead. Collisions with external kinematic bodies are detected using a level-set representation of the latter, allowing computation of collision depth and normal at $O(1)$ time. For self-collision detection we use, in principle, the same technique employed by Teran et al [TSIF05] and McAdams et al [MZS*11] by first checking each proxy against an axis-aligned bounding box hierarchy defined over the tetrahedra of the flesh volume. Once a collision is detected, the offending interior location

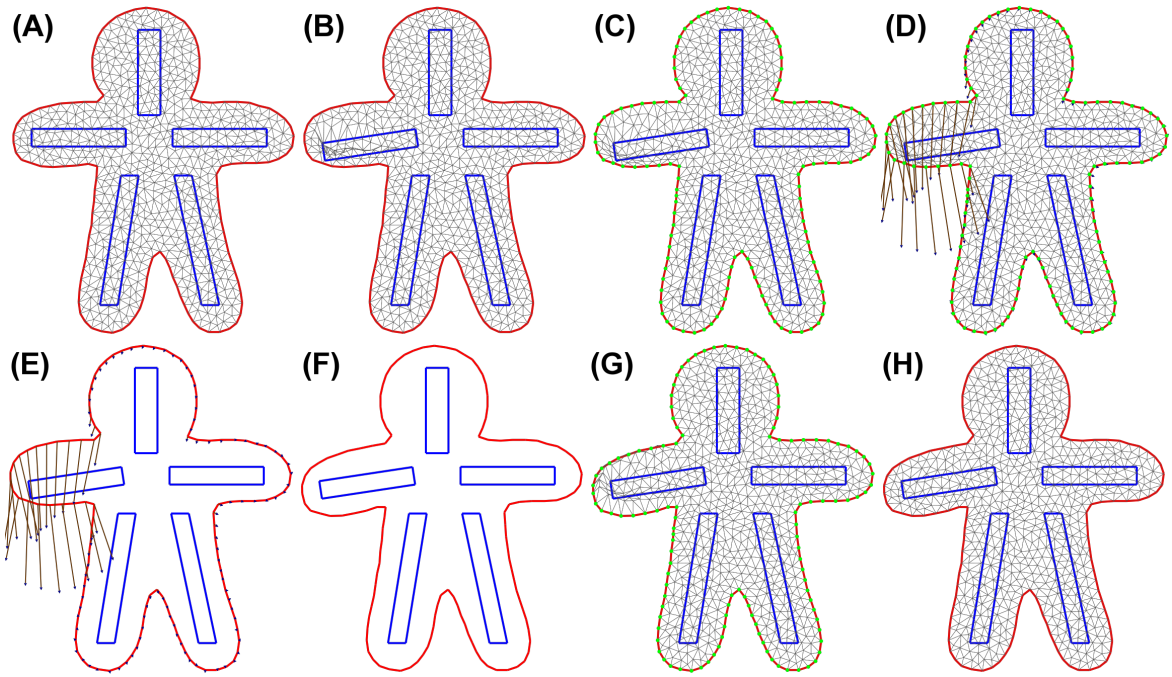

Figure 2: *Pictorial illustration of our surface-based simulation timeline: (a) Starting pose, boundary outlined in red. (b) A bone is articulated, causing its associated constrained vertices to move. (c) Keeping the boundary fixed, we relax all interior vertices to equilibrium. Convergence is fast due to the extensive constraints on bones and surface. (d) Elastic forces are computed on the boundary. (e) Disregarding the interior, we advance the surface-based model to equilibrium using the previous forces as external stimuli. (f) The surface model is converged. (g) Interior vertices have not been updated, as they did not participate in the last step. (h) If desired, interior vertices can be relaxed one last time. The result is the same as full volumetric simulation.*

is mapped to its undeformed coordinates, where a level-set representation of the flesh volume is used to project it to the skin surface. Finally, that surface location is mapped back to the deformed configuration, providing an approximate collision target that the proxy should be attracted to. Ultimately, every colliding proxy $\vec{x}_i$ receives a penalty force

$$\vec{f}_i^{\,\text{col}} = -\mathbf{k}(\vec{x}_i - \vec{x}_i^{\,\text{target}}). \tag{2}$$

Finally, if the proxy in question was party to a self-collision, the collision target will also receive the opposite penalty force $\vec{f}_i^{\,\text{target}} = -\vec{f}_i^{\,\text{col}}$ which is barycentrically distributed to its embedding (surface) vertices.

Quasistatic simulation. Our skinning algorithm operates in the context of quasistatic simulation; as a consequence, each skeletal pose is mapped directly to an equilibrium shape of the flesh volume. For simplicity, we assume that the vertices of the simulation mesh include a set of *constrained* nodes $\mathbf{x}_c = \mathbf{x}_c(\mathbf{q})$ whose coordinates are kinematically determined by the pose of the articulated skeleton ($\mathbf{q}$), while the remaining unconstrained vertices are partitioned into *interior* ($\mathbf{x}_i$) and *boundary* nodes ($\mathbf{x}_b$). The flesh shape resulting from a pose $\mathbf{q}$ is determined by requiring that all unconstrained nodes (interior and boundary) are at force equilibrium:

$$\mathbf{f}_b(\mathbf{x}_b, \mathbf{x}_i, \mathbf{x}_c(\mathbf{q})) = \mathbf{0} \;\text{ and }\; \mathbf{f}_i(\mathbf{x}_b, \mathbf{x}_i, \mathbf{x}_c(\mathbf{q})) = \mathbf{0} \tag{3}$$

In Section 4 we discuss a formulation which yields elastic forces that are affine functions of $\mathbf{x}_i$ and $\mathbf{x}_b$. Thus, in the absence of collisions equation (3) is just a linear system of equations. In the presence of collisions, however, the boundary forces $\mathbf{f}_b$ (note: *only* the boundary forces, not the interior $\mathbf{f}_i$) incorporate nonlinear collision-related terms. Although equation (2) seems linear, it is in fact a nonlinear term since its contribution is activated or deactivated depending on whether $\vec{x}_i$ is colliding, while additionally the collision target $\vec{x}_i^{\,\text{target}}$ is a function of the proxy location $\vec{x}_i$ itself.

In cases where the system of equation (3) is nonlinear (whether this is due to a nonlinear material or collisions), a Newton-Raphson procedure is employed for iterative solution. In each iteration, we perform collision detection and, if a proxy is found to be colliding, we instance (for the duration of the Newton iteration) a zero-restlength spring connecting the collision proxy with its target, which is either an explicit point in space (for external body collision) or an embedded surface location (for self collision). Thus, in each Newton step the total forces in our flesh model are given by the sum of elastic material forces plus any (temporarily created) collision-induced springs; since we use *linear* zero-restlength springs for collision response, no further linearization is needed in the Newton-Raphson procedure (beyond adding and removing springs after each detection step).

Figure 3: *Arm bending simulation using standard corotated elasticity (left), vs. our ex-rotated elasticity model (right).*

4. The ex-rotated elasticity model

We propose a new material model for skeleton-driven flesh deformation, which we call *extrinsically rotated elasticity*, or *ex-rotated* elasticity for short. Our objective is to mimic as closely as possible the nonlinear behavior of corotational elasticity (see figure 3). However, our design goal is to have the forces of our material model be *affine* functions of the vertex positions ($\mathbf{x}$), but with coefficients that are dependent on the pose ($\mathbf{q}$) of the articulated skeleton, as follows:

$$\mathbf{f}(\mathbf{x};\mathbf{q}) = \mathbf{K}(\mathbf{q})\mathbf{x} + \mathbf{g}(\mathbf{q}). \tag{4}$$

Note that such a linear relation would be trivial to achieve if we used linear elasticity as the flesh material model – in this case $\mathbf{K}$ and $\mathbf{g}$ would actually be pose-independent constants. Of course, linear elasticity lacks rotational invariance and suffers from artifacts under large deformation. Our rationale is that, by making $\mathbf{K}(\mathbf{q})$ and $\mathbf{g}(\mathbf{q})$ pose-dependent we have the opportunity to mitigate such artifacts while retaining the linearity (with respect to the shape $\mathbf{x}$) of equation (4). Specifically, our new model defines the energy density as

$$\Psi(\mathbf{F};\mathbf{q}) = \mu \left\| \mathbf{F} - \hat{\mathbf{R}}(\mathbf{q}) \right\|_F^2 + \frac{\lambda}{2}\mathrm{tr}^2\left[\hat{\mathbf{R}}(\mathbf{q})^T\mathbf{F} - \mathbf{I}\right] \tag{5}$$

while, for comparison, standard corotational elasticity uses

$$\Psi_{\mathrm{COR}}(\mathbf{F}) = \mu \left\| \mathbf{F} - \mathbf{R}(\mathbf{F}) \right\|_F^2 + \frac{\lambda}{2}\mathrm{tr}^2\left[\mathbf{R}(\mathbf{F})^T\mathbf{F} - \mathbf{I}\right].$$

In both equations, Ψ is the energy density, $\mathbf{F}$ is the deformation gradient, and μ, λ are the Lamé coefficients. In the corotated model, $\mathbf{R}(\mathbf{F})$ is the rotational component of the polar decomposition $\mathbf{F} = \mathbf{RS}$ of the deformation gradient, thus ultimately a function of the deformed shape. As a consequence, $\Psi_{\mathrm{COR}}(\mathbf{F})$ is not a pure quadratic, and elastic forces will be inherently nonlinear. In contrast, our proposed energy in equation (5) replaces $\mathbf{R}(\mathbf{F})$ with a pose-dependent approximation $\hat{\mathbf{R}}(\mathbf{q})$, which is computed via procedural skinning.

More specifically, we compute a collection of scalar fields corresponding to blending weights, one for each independently moving skeletal component, using the Bounded Biharmonic Weights (BBW) method [JBPS11]. Discretely, the BBW method produces weights on every mesh vertex; we average those to tetrahedron centers and normalize across all bones to make sure they form a partition of unity. We then compute a procedural pose-dependent (yet shape-independent) skinning transform at each tetrahedron center using Dual-Quaternion skinning [KCZO08]. We ultimately retain only the rotational component of this transform as $\hat{\mathbf{R}}(\mathbf{q})$ and use exactly this value in equation (5) to compute elastic forces on each tetrahedron. Note that since we only care about the rotational component, the result is essentially that of Spherical Linear Interpolation (with Bounded Biharmonic Weights). Our approach is inspired by the work of Capell et al [CGC*02], although we use a separate rotation per tetrahedron and our skeletal transform is built directly in to the energy model, while their work partitions the body mesh into bone-associated regions and applies macroscopic rotations on locally computed discretizations of elasticity.

We observe that the energy of equation (5) is now purely quadratic on $\mathbf{F}$ (and by extension on vertex positions). The associated first Piola-Kirchhoff stress tensor $\mathbf{P} = \partial\Psi/\partial\mathbf{F}$ is

$$\mathbf{P}(\mathbf{F};\mathbf{q}) = 2\mu\left[\mathbf{F} - \hat{\mathbf{R}}(\mathbf{q})\right] + \lambda\mathrm{tr}\left[\hat{\mathbf{R}}(\mathbf{q})^T\mathbf{F} - \mathbf{I}\right]\hat{\mathbf{R}}(\mathbf{q}) \tag{6}$$

Since equation (6) is affine on $\mathbf{F}$, substituting into equation (1) yields purely affine nodal forces, as encapsulated in our initially desired formulation (4). Note that, when using equation (6) to evaluate discrete forces, we take $\hat{\mathbf{R}}(\mathbf{q})$ at the centroid of each tetrahedron, and treat this (similar to the treatment of $\mathbf{F}$) as an element-wise constant matrix. From a computational standpoint, an explicit representation of $\mathbf{g}(\mathbf{q})$ is not directly utilized (it suffices to be able to evaluate $\mathbf{f}(\mathbf{x};\mathbf{q})$ via equations 1 and 6), but if desired it could be computed simply as $\mathbf{g}(\mathbf{q}) = \mathbf{f}(\mathbf{0};\mathbf{q})$. Constructing the stiffness matrix $\mathbf{K}(\mathbf{q})$ is more relevant to us; consider the "canonical" energy

$$\Psi_0(\mathbf{F}) = \mu\|\mathbf{F} - \mathbf{I}\|_F^2 + \frac{\lambda}{2}\mathrm{tr}^2(\mathbf{F} - \mathbf{I})$$

resulting from fixing the rotation $\hat{\mathbf{R}}$ to the identity matrix. Then, for tetrahedron T_e with vertices $\mathbf{x}_e = (\vec{x}_1^{\,e}, \vec{x}_2^{\,e}, \vec{x}_3^{\,e}, \vec{x}_4^{\,e})$ we can define the associated (canonical) stiffness matrix as

$$\mathbf{K}_0^e = -\frac{\partial^2\Psi_0(\mathbf{F}_e)}{\partial\mathbf{x}_e^2} \cdot \mathrm{Volume}(T_e)$$

This would be the contribution of T_e to the stiffness matrix, if the rotation at its centroid was $\hat{\mathbf{R}}^e = \mathbf{I}$. In our model, this elemental stiffness is conjugated by the block diagonal matrix $\mathbf{R}_*^e(\mathbf{q}) = \mathrm{diag}\left[\hat{\mathbf{R}}^e(\mathbf{q}), \hat{\mathbf{R}}^e(\mathbf{q}), \hat{\mathbf{R}}^e(\mathbf{q}), \hat{\mathbf{R}}^e(\mathbf{q})\right]$ to yield

$$\mathbf{K}^e(\mathbf{q}) = \mathbf{R}_*^e(\mathbf{q}) \cdot \mathbf{K}_0^e \cdot (\mathbf{R}_*^e(\mathbf{q}))^T$$

Since $\mathbf{K}_0^e$ is symmetric negative definite (it is the negated Hessian of a convex quadratic), its orthogonal conjugation $\mathbf{K}^e(\mathbf{q})$ and the global stiffness matrix assembled from all elemental contributions will also be symmetric negative definite. As a consequence, Conjugate Gradients can be used to solve the equilibrium equations (3) without the need for an indefiniteness treatment as in McAdams et al [MZS*11].

In our experiments, the ex-rotated model produced visual results that were strikingly similar to standard (nonlinear) corotated elasticity, as seen in Figure 3. This was the case, even in scenarios involving collision. Upon closer inspection, small differences can be seen, but typically well below the threshold of perceptible non-physicality. Of course, we can envision scenarios that discrepancies would be more visible: areas of loose skin, that are not well attached to the underlying skeleton, or parts of the skin where folds and wrinkles tend to form might accentuate visible differences. For the range and styles of motions we examined, however, we found the ex-rotated approximation to be surprisingly plausible. Finally, we note that the use of the procedural rotation $\hat{\mathbf{R}}(\mathbf{q})$ in our energy density *does not* force the simulation to match this local rotation exactly – the final rotation is determined by the quasistatic solution of the ex-rotated forces.

5. A volumetrically reactive surface-only model

The ex-rotated model allowed us, for a given skeletal pose, to express the elastic forces as a purely affine function of vertex positions. We claim that an even more aggressive optimization is now made possible: we will show that, if the interior vertices $\mathbf{x}_i$ are assumed to be at quasistatic equilibrium at all times, we can derive an *affine* mapping directly from boundary vertex positions $\mathbf{x}_b$ to the forces $\mathbf{f}_b^*$ exerted on them by the deformed volume. We will initially use the symbol $\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})$ to distinguish boundary forces when the interior is equilibrated vs. the notation $\mathbf{f}_b(\mathbf{x}_b,\mathbf{x}_i,\mathbf{x}_c(\mathbf{q}))$ (no asterisk) when we allow the interior nodes to be at arbitrary (non-equilibrium) positions, as previously in equation (3).

Conceptually, the function $\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})$ encapsulates the result of the following process:

- We place the boundary vertices at user-specified locations $\mathbf{x}_b$, treating them as Dirichlet boundary conditions.
- Given the user-specified boundary values $\mathbf{x}_b$ and the pose-determined constrained vertices $\mathbf{x}_c(\mathbf{q})$, we compute the equilibrium position of all interior vertices, which we denote by $\mathbf{x}_i^*(\mathbf{x}_b;\mathbf{q})$. This is accomplished by solving (for $\mathbf{x}_i^*$) the *interior* equilibrium system

$$\mathbf{f}_i(\mathbf{x}_b,\mathbf{x}_i^*,\mathbf{x}_c(\mathbf{q}))=\mathbf{0} \tag{7}$$

  treating $\mathbf{x}_b$ and $\mathbf{x}_c(\mathbf{q})$ as constants. Equation (7) is purely affine, even in the presence of collisions, since collision penalty forces only contribute to boundary forces $\mathbf{f}_b$. Thus, equation (7) has a unique solution for $\mathbf{x}_i^*(\mathbf{x}_b;\mathbf{q})$.
- We substitute $\mathbf{x}_i^*(\mathbf{x}_b;\mathbf{q})$ into the equation (3) for $\mathbf{f}_b$, obtaining the final forces that the boundary feels from the equilibrated interior. The final result is denoted as $\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})$

$$\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})=\mathbf{f}_b(\mathbf{x}_b,\mathbf{x}_i^*(\mathbf{x}_b;\mathbf{q}),\mathbf{x}_c(\mathbf{q})) \tag{8}$$

The important consequence of this manipulation is that we can replace the volumetric equilibrium equations (3) with the surface-only variant

$$\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})=\mathbf{0}$$

This last expression has abstracted away the interior degrees of freedom, but is fully equivalent with the volumetric equilibrium conditions. Of course, if the equilibrium locations of the interior nodes are also needed for any reason, they can be readily computed by solving equation (7).

The interesting fact we will prove is that $\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})$, for all its perceived complexity, is merely an affine function of $\mathbf{x}_b$. We start by differentiating equation (7) with respect to $\mathbf{x}_b$:

$$\frac{\partial}{\partial \mathbf{x}_b}\left\{\mathbf{f}_i(\mathbf{x}_b,\mathbf{x}_i^*,\mathbf{x}_c(\mathbf{q}))\right\}=\mathbf{0}\Rightarrow$$

$$\Rightarrow \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_b}+\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i}\cdot\frac{\partial \mathbf{x}_i^*}{\partial \mathbf{x}_b}=\mathbf{0}\Rightarrow \frac{\partial \mathbf{x}_i^*}{\partial \mathbf{x}_b}=-(\mathbf{K}_{ii})^{-1}\mathbf{K}_{ib} \tag{9}$$

where we use the notation $\mathbf{K}_{ii}:=\partial\mathbf{f}_i/\partial\mathbf{x}_i$, $\mathbf{K}_{ib}:=\partial\mathbf{f}_i/\partial\mathbf{x}_b$ (similarly for $\mathbf{K}_{bi}$ and $\mathbf{K}_{bb}$, to be used later) for the blocks of the stiffness matrix corresponding to the partitioning of the unconstrained vertices into interior and boundary. Note that the chain rule applied in equation (9) would technically dictate that the partial derivative $\partial\mathbf{f}_i/\partial\mathbf{x}_i$ should be evaluated at $\mathbf{x}_i^*$. However, $\partial\mathbf{f}_i/\partial\mathbf{x}_i=\mathbf{K}_{ii}$ is constant, so an explicit specification of the evaluation point is not necessary. Next, we differentiate equation (8) with respect to $\mathbf{x}_b$:

$$\frac{\partial}{\partial \mathbf{x}_b}\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})=\frac{\partial}{\partial \mathbf{x}_b}\left\{\mathbf{f}_b(\mathbf{x}_b,\mathbf{x}_i^*(\mathbf{x}_b;\mathbf{q}),\mathbf{x}_c(\mathbf{q}))\right\}=$$

$$=\frac{\partial \mathbf{f}_b}{\partial \mathbf{x}_b}+\frac{\partial \mathbf{f}_b}{\partial \mathbf{x}_i}\cdot\frac{\partial \mathbf{x}_i^*}{\partial \mathbf{x}_b}=\mathbf{K}_{bb}-\mathbf{K}_{bi}(\mathbf{K}_{ii})^{-1}\mathbf{K}_{ib} \tag{10}$$

So far we have not *assumed* that $\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})$ is an affine function of $\mathbf{x}_b$; but equation (10) effectively proves that fact, since the derivative with respect to $\mathbf{x}_b$ is constant (it is only a function of $\mathbf{q}$). We can, thus, write

$$\mathbf{f}_b^*(\mathbf{x}_b;\mathbf{q})=\mathbf{K}_b(\mathbf{q})\mathbf{x}_b+\mathbf{g}_b(\mathbf{q}), \tag{11}$$

$$\text{where}\quad \mathbf{K}_b(\mathbf{q})=\mathbf{K}_{bb}(\mathbf{q})-\mathbf{K}_{bi}(\mathbf{q})\left[\mathbf{K}_{ii}(\mathbf{q})\right]^{-1}\mathbf{K}_{ib}(\mathbf{q}) \tag{12}$$

is easily recognized as the Schur complement of $\mathbf{K}_{bb}$ in the stiffness matrix $\mathbf{K}(\mathbf{q})$. Thus, $\mathbf{K}_b(\mathbf{q})$ is symmetric since $\mathbf{K}_{bi}=\mathbf{K}_{ib}^T$, and inherits the negative definiteness of $\mathbf{K}(\mathbf{q})$. The exact algebraic expression of $\mathbf{g}_b(\mathbf{q})$ is of less relevance; the typical mode of utilization of equation (11) would be as follows: Given an initial guess for the equilibrium boundary shape $\mathbf{x}_b^{(0)}$, we seek a correction $\delta\mathbf{x}_b$ such that

$$\mathbf{f}_b^*(\mathbf{x}_b^{(0)}+\delta\mathbf{x}_b;\mathbf{q})=\mathbf{0}\Rightarrow \mathbf{f}_b^*(\mathbf{x}_b^{(0)};\mathbf{q})+\mathbf{K}_b(\mathbf{q})\delta\mathbf{x}_b=\mathbf{0}\Rightarrow$$

$$\Rightarrow -\mathbf{K}_b(\mathbf{q})\delta\mathbf{x}_b=\mathbf{f}_b^*(\mathbf{x}_b^{(0)};\mathbf{q}) \tag{13}$$

The right hand side of equation (13) can be computed via equation (8), and the coefficient matrix $-\mathbf{K}_b(\mathbf{q})$ is symmetric positive definite. As a consequence, the Conjugate Gradients algorithm can be used to solve for the boundary update.

The formulation we presented is related to a familiar concept in domain decomposition and iterative substructuring approaches. The continuous analogue of our surface-based force model is known as the *Steklov-Poincaré* operator [QV99] and is defined as a differential operator that maps one type of boundary conditions for an elliptic PDE to another. In our case, we map Dirichlet boundary condition to the equivalent traction conditions (or, discretely, boundary forces), via $\mathbf{f}_b^*(\mathbf{x}_b)$. In our skinning application, the tangible benefit of using the surface-based elasticity formulation is that equation (8) has a significantly lower dimensionality than its volumetric counterpart, and thus Conjugate Gradients (or other iterative solvers) converge much faster, as illustrated in Figure 4. A related process in computational mechanics is *static condensation*, which has been leveraged in the visual computing literature for surgery simulation [BNC96]. An additional alternative could also be to construct a surface-based discretization using the Boundary Element Method, which was used as the basis of the ARTDEFO system by James and Pai [JP99].

5.1. Preconditioning the boundary equations

One may observe that there are serious practicality considerations about utilizing equation (13) directly, to determine the boundary deformation: the coefficient matrix involves an expensive inversion, and the right hand side requires a solution for the interior equilibrium locations $\mathbf{x}_i^*$. These concerns will be addressed in Section 6 but, before we turn to such linear algebra-related considerations, let us investigate the option of preconditioning the boundary system (13) to further accelerate the convergence of Conjugate Gradients.

A good preconditioner would provide an approximation to the coefficient matrix $\mathbf{K}_b(\mathbf{q})$. If this matrix was not pose-dependent, we could consider the possibility of using a sparse approximate factorization, computed in advance, as the preconditioner. Of course this is not an option, because the coefficient matrix would constantly change as the skeleton moves. A more promising option has to do with the relation $\mathbf{K}_b(\mathbf{q})$ has with its *reference* value $\mathbf{K}_b(\mathbf{0})$, evaluated around the reference skeletal pose (denoted $\mathbf{q}=\mathbf{0}$). In fact, the two matrices would be intimately related, if the skeleton had been rigidly rotated with a *global*, constant rotation $\mathbf{R}^0$. Due to the way $\mathbf{K}(\mathbf{q})$ is constructed, in such a scenario we have the exact equality $\mathbf{K}(\mathbf{q}) = \mathbf{R}_*^0 \cdot \mathbf{K}(\mathbf{0}) \cdot [\mathbf{R}_*^0]^T$, where the block diagonal matrix $\mathbf{R}_*^0 = \text{diag}(\mathbf{R}^0, \mathbf{R}^0, \ldots, \mathbf{R}^0)$ contains the global rotation $\mathbf{R}^0$ in each of its 3×3 diagonal blocks (one block for each mesh vertex). From equation (12) we can confirm that this similarity property would carry over to the Schur complement as $\mathbf{K}_b(\mathbf{q}) = \mathbf{R}_*^0 \cdot \mathbf{K}_b(\mathbf{0}) \cdot [\mathbf{R}_*^0]^T$. Note that we re-used the notation $\mathbf{R}_*^0$ in this last expression to denote

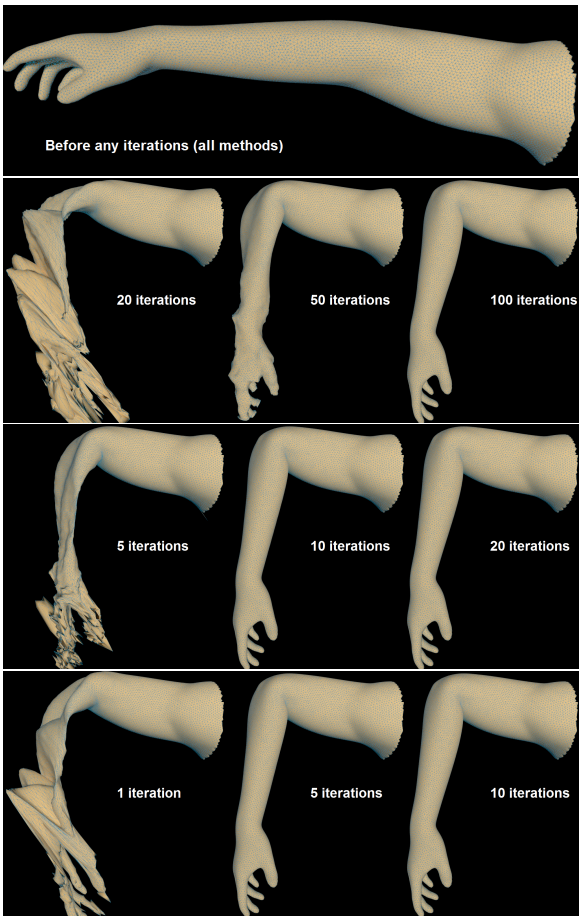

Figure 4: *Conjugate Gradients is used to solve for the shape of an arm, after the elbow joint has been sharply bent to 90 degrees. Convergence speed is compared for the volumetric ex-rotated formulation (2nd row), our surface-based formulation without a preconditioner (3rd row), and the surface formulation with our proposed preconditioner (4th row).*

a block diagonal matrix with as many blocks as *boundary* vertices. We will tolerate this overloaded notation as long as the context disambiguates the proper size.

These observations suggest the following idea: Let us construct the block diagonal matrix

$$\mathbf{R}_*(\mathbf{q}) = \text{diag}\left(\hat{\mathbf{R}}_1(\mathbf{q}), \hat{\mathbf{R}}_2(\mathbf{q}), \ldots, \hat{\mathbf{R}}_m(\mathbf{q})\right)$$

where $\hat{\mathbf{R}}_i(\mathbf{q})$ is the rotation produced by procedural skinning in at the location of the i-th boundary vertex. Subsequently, consider approximating $\mathbf{K}_b(\mathbf{q})$ with its "skinned" version:

$$\hat{\mathbf{K}}_b(\mathbf{q}) = \mathbf{R}_*(\mathbf{q}) \cdot \mathbf{K}_b(\mathbf{0}) \cdot [\mathbf{R}_*(\mathbf{q})]^T \qquad (14)$$

We claim that $\hat{\mathbf{K}}_b(\mathbf{q})$ can be used as a highly effective preconditioner for $\mathbf{K}_b(\mathbf{q})$. We have already established that the two matrices are identical when $\mathbf{q}$ is a global rotation. In the general case where the skeleton has been articulated in a

non-rigid fashion, the two matrices will be different (remember that $\mathbf{K}(\mathbf{q})$ is built by conjugating the canonical stiffness matrix element-by-element, not vertex-by-vertex), but still very close, numerically. Qualitatively, the two operators are almost identical "away" from joints, and develop discrepancies in areas where the skinning transformation blends the influence of several bones. In addition, since:

$$[\hat{\mathbf{K}}_b(\mathbf{q})]^{-1} = \mathbf{R}_*(\mathbf{q}) \cdot [\mathbf{K}_b(\mathbf{0})]^{-1} \cdot [\mathbf{R}_*(\mathbf{q})]^T \qquad (15)$$

a precomputed factorization of $\mathbf{K}_b(\mathbf{0})$ could be leveraged to quickly invert $\hat{\mathbf{K}}_b(\mathbf{q})$; we elaborate on this in Section 6.

Our experiments indicated a very favorable preconditioning performance for $\hat{\mathbf{K}}_b(\mathbf{q})$, both reducing the necessary iteration count by close to an order of magnitude as well as improving the visual quality of the very first few PCG results (prior to convergence). For single-joint models, as few as 3-4 PCG iterations were often enough. Figure 4 includes visual samples from the convergence sequence of our preconditioned scheme, compared to using either of the unpreconditioned volumetric or surface-based alternatives.

5.2. Incorporating collisions

In the presence of collision events, the boundary-specific force includes terms originating from penalty forces in addition to internal elastic forces. Very little needs to change in our quasistatic solution process; the most essential change is that equation (13) needs to include collision terms, and we also need to solve this equation with a full Newton-Raphson iteration, due to the nonlinearity of collision forces. Each Newton iteration is given by:

$$-\left[\mathbf{K}_b(\mathbf{q}) + \mathbf{K}_b^{\text{coll}}(\mathbf{x}_b^{(k)})\right] \delta\mathbf{x}_b^{(k)} = \mathbf{f}_b^*(\mathbf{x}_b^{(k)}; \mathbf{q}) + \mathbf{f}_b^{\text{coll}}(\mathbf{x}_b^{(k)})$$

where $\mathbf{f}_b^{\text{coll}}(\mathbf{x}_b^{(k)})$ is the collision spring force and $\mathbf{K}_b^{\text{coll}}(\mathbf{x}_b^{(k)})$ its associated stiffness, for those collision proxies found to be colliding in configuration $\mathbf{x}^{(k)}$. We have successfully used the same preconditioner $\hat{\mathbf{K}}_b(\mathbf{q})$ without adapting it for the extra penalty terms, with very satisfactory performance. In practice, we observe that areas prone to self-collision (e.g. joints) often coincide with areas where $\hat{\mathbf{K}}_b(\mathbf{q})$ was less effective of a preconditioner to begin with (without collisions).

6. Implementation and practical optimizations

The theoretical results of the previous sections suggest that the linearized Newton-Raphson system can be solved via Conjugate Gradients, at a low iteration count. However, there are three significant practical concerns, related to the implementation of this algorithm: (a) the matrix $\mathbf{K}_b(\mathbf{q})$ is dense, and difficult to compute as it requires a matrix inversion, (b) the right hand side is not readily available, and (c) the preconditioner $\hat{\mathbf{K}}_b(\mathbf{q})$ is dense, and not in a readily invertible form. In this section we provide practical remedies to those concerns, to make the surface-based formulation viable and competitive from a performance standpoint.

6.1. Modified Newton iteration

In order to facilitate efficient iterative solution, one option is to replace the coefficient matrix of equation (13) with a more numerically favorable approximation. This is similar to altering the Newton-Raphson matrix as other authors have previously done [MZS*11, TSIF05] in the interest of safeguarding (or restoring) positive definiteness. The resulting *Modified Newton* iteration will still converge to the same solution if the approximation is not too invasive. For example, a sufficient condition would be that $\tilde{\mathbf{K}}_b(\mathbf{q})$ and $\mathbf{K}_b(\mathbf{q})$ satisfy the spectral criterion $\rho[\mathbf{I} - [\tilde{\mathbf{K}}_b(\mathbf{q})]^{-1}\mathbf{K}_b(\mathbf{q})] < 1$. Of course, this would *require* a Newton-style iteration, even when the problem would have been otherwise linear. However, from a practical standpoint, a Newton-style iteration would be mandated anyways by the presence of nonlinear collision terms.

The approximation we leverage in our implementation starts by approximating $\mathbf{K}_{ii}$ with its "skinned" reference value $\mathbf{K}_{ii}(\mathbf{q}) \approx \mathbf{R}_*(\mathbf{q})\mathbf{K}_{ii}(\mathbf{0})[\mathbf{R}_*(\mathbf{q})]^T$. In addition, we replace the reference matrix $\mathbf{K}_{ii}(\mathbf{0})$ (at the reference skeletal pose) with its incomplete Cholesky factorization $\mathbf{K}_{ii}(\mathbf{0}) \approx \mathbf{L}_i\mathbf{L}_i^T$. Finally, our approximation $\tilde{\mathbf{K}}_b(\mathbf{q})$ becomes:

$$\tilde{\mathbf{K}}_b(\mathbf{q}) = \mathbf{K}_{bb}(\mathbf{q}) - \mathbf{K}_{bi}(\mathbf{q})\mathbf{R}_*(\mathbf{q})\mathbf{L}_i^{-T}\mathbf{L}_i^{-1}[\mathbf{R}_*(\mathbf{q})]^T\mathbf{K}_{ib}(\mathbf{q})$$

Note that, although this matrix is also dense, when used as the system matrix of Conjugate Gradients it never needs to be explicitly built; instead, we simply need to have a process by which the matrix-vector product $\tilde{\mathbf{K}}_b(\mathbf{q}) \cdot \mathbf{w}$ between the matrix and a vector $\mathbf{w}$ provided by the Krylov solver can be evaluated. This can be done with just sparse algebra:

- Multiplications by factors $\mathbf{K}_{bb}, \mathbf{K}_{bi}, \mathbf{K}_{ib}$ can be performed efficiently, as those are sparse sub-blocks of $\mathbf{K}(\mathbf{q})$, which is constructed at linear cost (relative to total vertices) as detailed in Section 4.
- Multiplications by $\mathbf{R}_*(\mathbf{q})$ and its transpose are linear-cost vertex-by-vertex rotations.
- Multiplication by $\mathbf{L}_i^{-1}$ is performed in linear time (in the number of nonzero entires of $\mathbf{L}_i$, and equivalently $\mathbf{K}_{ii}$) via forward substitution. Similarly, multiplication by $\mathbf{L}_i^{-T}$ is performed in linear time via backward substitution.

As a consequence, each matrix-vector product $\tilde{\mathbf{K}}_b(\mathbf{q}) \cdot \mathbf{w}$ can be computed at a cost proportional to the sparsity of the volumetric stiffness $\mathbf{K}(\mathbf{q})$. Although we would have hoped for a per-iteration cost that scales only with the boundary, the significant reduction in the number of iterations that Conjugate Gradients requires for convergence still makes our method computationally beneficial, especially in scenarios with a significant number of active collisions. Further quantitative details on this are provided in the examples section. In practice, we have noticed a very modest impact on the number of additional required iterations due to the Modified Newton procedure itself, compared to the extra Newton iterations that are needed, regardless, due to the nonlinearity of collision forces.

6.2. Right-hand side update

The right-hand side $\mathbf{f}_b^*(\mathbf{x}_b^{(k)};\mathbf{q}) = \mathbf{f}_b(\mathbf{x}_b^{(k)}, \mathbf{x}_i^*(\mathbf{x}_b^{(k)};\mathbf{q}), \mathbf{x}_c(\mathbf{q}))$ in the Newton system involves the nontrivial quantity $\mathbf{x}_i^*(\mathbf{x}_b^{(k)};\mathbf{q})$, i.e. the equilibrium interior positions subject to given boundary and bone-constrained vertices. This can be found by solving the *linear* interior equilibrium equation (7) with Conjugate Gradients. We have found that simply using PCG for this system with the incomplete Cholesky preconditioner $\tilde{\mathbf{K}}_{ii} = \mathbf{R}_*(\mathbf{q})\mathbf{L}_i^{-T}\mathbf{L}_i^{-1}[\mathbf{R}_*(\mathbf{q})]^T$ converges extremely rapidly (typically in no more than 4-6 iterations). This is due to the fact that this is an extremely well supported elasticity problem; we are solving for an interior layer which is tightly packed between a fixed boundary and a fixed skeletal layer. In addition, this interior solve involves no collisions. All these factors contribute to very good preconditioning performance by incomplete Cholesky (notably improved, per our observations, compared to the same preconditioner on a volumetric model with un-constrained surface).

6.3. Practical Preconditioner

In accordance with equation (15), our objective is to generate an approximation to $\mathbf{K}_b(\mathbf{0})$ that can be efficiently inverted. It is useful at this point to revisit the intuitive interpretation of $\mathbf{K}_b$; this matrix captures the *boundary response* of the ex-rotated flesh model. The i-th column of this matrix encodes the forces that will manifest on boundary nodes, if we impart a unit displacement on the i-th boundary degree of freedom, while keeping the rest of the boundary fixed. Given the extensive attachments of the flesh to the underlying skeleton, we would expect these boundary forces to be strongly concentrated around the location of the disturbance that triggered them (the i-th boundary variable). Thus, although $\mathbf{K}_b$ is algebraically dense, it is *practically* concentrated on a sparse set of entries which correspond to pairs of boundary vertices with relatively small mesh distance. In our implementation, we generate a specific sparsity pattern by only allowing nonzero entries on pairs of degrees of freedom that have a maximum graph distance of 4-8 mesh edges and approximate $\mathbf{K}_b$ with a sparse matrix $\tilde{\mathbf{K}}_b$ that only contains nonzero elements at the specified sparse locations.

Although we can tolerate the time expense of constructing $\tilde{\mathbf{K}}_b$ as a one-time preprocessing cost, we have to be conscious about space requirements. To aid in this construction we use the following expression for the *exact* $\mathbf{K}_b(\mathbf{0})$:

$$\begin{aligned}\mathbf{K}_b(\mathbf{0}) &= \mathbf{K}_{bb}(\mathbf{0}) - \mathbf{K}_{bi}(\mathbf{0})\mathbf{L}_i^{-T}\mathbf{L}_i^{-1}\mathbf{K}_{ib}(\mathbf{0}) \\ &= \mathbf{K}_{bb}(\mathbf{0}) - \mathbf{M}\mathbf{M}^T = \mathbf{K}_{bb}(\mathbf{0}) - \textstyle\sum_{i=1}^n m_i m_i^T . \quad (16)\end{aligned}$$

In this expression, $\mathbf{K}_{ii}(\mathbf{0}) = \mathbf{L}_i\mathbf{L}_i^T$ is the *exact* Cholesky factorization of $\mathbf{K}_{ii}$ (we reorder variables to make it as sparse as possible), and $\mathbf{M} = \mathbf{K}_{bi}(\mathbf{0})\mathbf{L}_i^{-T}$. The i-th column of $\mathbf{M}$ is $m_i = \mathbf{M}e_i = \mathbf{K}_{bi}(\mathbf{0})\mathbf{L}_i^{-T}e_i$ and can be computed via backward substitution and a sparse matrix-vector multiply. We build the sparse $\tilde{\mathbf{K}}_b$ by incorporating the contribution of one m_i at a time, without ever building a dense matrix (instead, we only update entries in the allowable sparsity pattern). The absolute value of any discarded entries which would normally be contributed to the values K_{ij}, K_{ji} falling outside of the allowable sparsity pattern, are instead subtracted from the respective diagonal entries $\tilde{K}_{ii}$ and $\tilde{K}_{jj}$ to preserve negative definiteness (this is equivalent to annihilating the off-diagonal contribution by adding a negative definite rank-one matrix to $\tilde{\mathbf{K}}_b$). Once $\tilde{\mathbf{K}}_b$ has been constructed, we compute an incomplete Cholesky factorization (with the same or a reduced sparsity pattern) to use as our final preconditioner.

7. Examples

In addition to the 2D models used for academic demonstrations in figure 2 and in the supplemental video, we benchmark our method in three different anatomic models: (a) An arm model meshed adaptively to refine aggressively near the surface. The model includes 220K tetrahedra, 12.5K boundary vertices and 26K interior vertices. (b) An arm model with less aggressive adaptivity (used in figures 3,4) with 259K tetrahedra, 10.2K boundary vertices and 36.3K interior vertices. (c) A full-body model (figure 1) with 864K tetrahedra, 32.8K boundary vertices and 95.5K interior vertices. The following table lists the runtime cost of (i) the procedural skinning update, (ii) the stiffness matrix construction, (iii) various components of the interior solve (section 6.2), (iv) components of the boundary system solve. The runtime per frame includes the collision detection cost. All times are on a 4-core Intel Xeon E3-1270 v3 CPU @ 3.5Ghz.

	Skinning Update	Stiffness matrix	Interior Solve (1 CG iteration)	Interior Solve (full CG solve)	Boundary Solve (1 CG iteration)	Boundary Solve (1 Newton step)	Average cost per frame
Arm - 220K tetrahedra (highly adaptive)	0.013s	0.128s	0.0087s	0.054s	0.0176s	0.095s	0.280s
Arm - 259K tetrahedra (moderately adaptive)	0.018s	0.135s	0.0108s	0.097s	0.0238s	0.120s	0.463s
Body - 864K tetrahedra	0.141s	0.680s	0.0348s	0.579s	0.0573s	0.864s	2.282s

Figure 4 demonstrated the convergence improvements of our scheme, relative to the un-preconditioned volumetric and surface-based approaches. For completeness, we also compare with a *preconditioned* volumetric alternative: we use $\tilde{\mathbf{K}} = \mathbf{R}_*(\mathbf{q})\mathbf{L}^{-T}\mathbf{L}^{-1}[\mathbf{R}_*(\mathbf{q})]^T$ as the preconditioner, where $\mathbf{K}(\mathbf{0}) \approx \mathbf{L}\mathbf{L}^T$ is an incomplete Cholesky factorization of the reference pose stiffness matrix. The following table details our results on the 259K arm model (b) described above.

Frame Range	Bending Angle	Collision Pairs (avg)	Suface-based (PCG)				Volumetric (PCG)		
			Interior Solve	1 CG iteration (boundary)	Boundary Solve (avg)	Frame Time (average)	1 CG iteration (volumetric)	Newton Solve (avg)	Frame Time (average)
0-15	0°-45°	0	0.093s	0.019s	0.152s	0.349s	0.013s	0.16s	0.293s
15-30	45°-90°	0	0.096s	0.02s	0.177s	0.38s	0.014s	0.174s	0.306s
30-35	90°-105°	64	0.097s	0.02s	0.198s	0.397s	0.016s	0.269s	0.407s
35-40	105°-120°	239	0.099s	0.023s	0.425s	0.619s	0.019s	1.12s	1.25s
40-45	120°-135°	530	0.098s	0.026s	0.48s	0.673s	0.02s	1.97s	2.11s
45-50	135°-150°	1094	0.097s	0.028s	0.512s	0.691s	0.022s	2.63s	2.77s

These results clearly indicate that our method is best suited to simulation of *large* meshes, in *collision-heavy* scenarios. Without collisions, the volumetric approach benefits heavily from the ex-rotated model, since a good preconditioner can be obtained, and it only requires a single Newton iteration (although PCG might converge slower than in our method).

8. Limitations and future work

The most evident among present limitations of our approach is the strong affinity to linear materials, and the adoption of a quasistatic-only simulation model. Although it would be possible to form the boundary-response system for the *linearized* equations of a nonlinear material, we would not have the same opportunities for precomputation of an effective preconditioner. In addition, our approach omits the optimization opportunities afforded by grid-based embedded simulation methods (e.g. [MZS*11]), as it presumes that boundary vertices are full (not embedded) degrees of freedom. As future work we will explore the possibility of extending out technique to *implicit* time integration schemes, where the linearity of the ex-rotated model might expose opportunities for a boundary-only evolution. Finally, a very exciting possibility would be to investigate combinations between our surface-based elasticity and model reduction schemes, using surface-only modes in the context of a reduced simulation.

Acknowledgments We thank Ladislav Kavan for numerous insightful discussions, and Rajsekhar Setaluri for assisting with the video. This research was supported in part by NSF IIS-1253598, NSF CNS-1218432, and US Army TARDEC.

References

[BNC96] Bro-Nielsen M., Cotin S.: Real-time volumetric deformable models for surgery simulation using finite elements and condensation. In *Computer graphics forum* (1996), vol. 15, pp. 57–66. 3, 7

[CBC*05] Capell S., Burkhart M., Curless B., Duchamp T., Popović Z.: Physically based rigging for deformable characters. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2005), pp. 301–310. 3

[CGC*02] Capell S., Green S., Curless B., Duchamp T., Popović Z.: Interactive skeleton-driven dynamic deformations. *ACM Trans. Graph. 21*, 3 (July 2002), 586–593. 3, 5

[CPSS10] Chao I., Pinkall U., Sanan P., Schröder P.: A simple geometric model for elastic deformations. *ACM Trans. Graph. 29*, 4 (July 2010), 38:1–38:6. 3

[HLSO12] Hecht F., Lee Y. J., Shewchuk J. R., O'Brien J. F.: Updated sparse cholesky factors for corotational elastodynamics. *ACM Trans.on Graph. 31*, 5 (2012), 123. 3

[HTC*13] Hahn F., Thomaszewski B., Coros S., Sumner R. W., Gross M.: Efficient simulation of secondary motion in rig-space. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2013), pp. 165–171. 3

[JBPS11] Jacobson A., Baran I., Popović J., Sorkine O.: Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph. 30*, 4 (July 2011), 78:1–78:8. 2, 5

[JP99] James D., Pai D.: ArtDefo: accurate real time deformable objects. In *Proceedings of SIGGRAPH 99* (1999), pp. 65–72. 3, 7

[JS11] Jacobson A., Sorkine O.: Stretchable and twistable bones for skeletal shape deformation. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 165:1–165:8. 2

[KCZO08] Kavan L., Collins S., Zara J., O'Sullivan C.: Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graph. 27*, 4 (Nov. 2008), 105:1–105:23. 1, 2, 5

[KJ12] Kim T., James D.: Physics-based character skinning using multidomain subspace deformations. *IEEE Trans. Visualization and Computer Graphics 18*, 8 (2012), 1228–1240. 3

[KJP02] Kry P. G., James D. L., Pai D. K.: Eigenskin: Real time large deformation character skinning in hardware. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2002), pp. 153–159. 2

[KS12] Kavan L., Sorkine O.: Elasticity-inspired deformers for character articulation. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 196:1–196:8. 2

[LCF00] Lewis J. P., Cordner M., Fong N.: Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *SIGGRAPH 2000 proceedings* (2000), pp. 165–172. 2

[LST09] Lee S.-H., Sifakis E., Terzopoulos D.: Comprehensive biomechanical modeling and simulation of the upper body. *ACM Trans. Graph. 28*, 4 (Sept. 2009), 99:1–99:17. 2

[MG03] Mohr A., Gleicher M.: Building efficient, accurate character skins from examples. *ACM Trans. Graph. 22*, 3 (July 2003), 562–568. 2

[MTLT88] Magnenat-Thalmann N., Laperrière R., Thalmann D.: Joint-dependent local deformations for hand animation and object grasping. In *Proceedings on Graphics Interface* (1988), pp. 26–33. 2

[MZS*11] McAdams A., Zhu Y., Selle A., Empey M., Tamstorf R., Teran J., Sifakis E.: Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (July 2011), 37:1–37:12. 1, 2, 3, 6, 8, 10

[QV99] Quarteroni A., Valli A.: *Domain decomposition methods for partial differential equations*, vol. 10. Clarendon Press, 1999. 3, 7

[SB12] Sifakis E., Barbič J.: FEM simulation of 3d deformable solids: A practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses* (2012), pp. 20:1–20:50. 2, 3

[SRC01] Sloan P.-P., Rose C., Cohen M.: Shape by example. In *Proceedings of the Symposium on Interactive 3D Graphics* (2001), pp. 135–143. 2

[TSB*05] Teran J., Sifakis E., Blemker S. S., Ng-Thow-Hing V., Lau C., Fedkiw R.: Creating and simulating skeletal muscle from the visible human data set. *IEEE Trans. Visualization and Computer Graphics 11*, 3 (2005), 317–328. 3

[TSIF05] Teran J., Sifakis E., Irving G., Fedkiw R.: Robust quasistatic finite elements and flesh simulation. *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2005), 181–190. 3, 8

[VBG*13] Vaillant R., Barthe L., Guennebaud G., Cani M.-P., Rohmer D., Wyvill B., Gourmel O., Paulin M.: Implicit skinning: Real-time skin deformation with contact modeling. *ACM Trans. Graph. 32*, 4 (July 2013), 125:1–12. 1, 3

[WP02] Wang X. C., Phillips C.: Multi-weight enveloping: Least-squares approximation techniques for skin animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2002), SCA '02, pp. 129–138. 2