Active Volumetric Musculoskeletal Systems

Ye Fan Joshua Litven Dinesh K. Pai

Department of Computer Science, University of British Columbia*

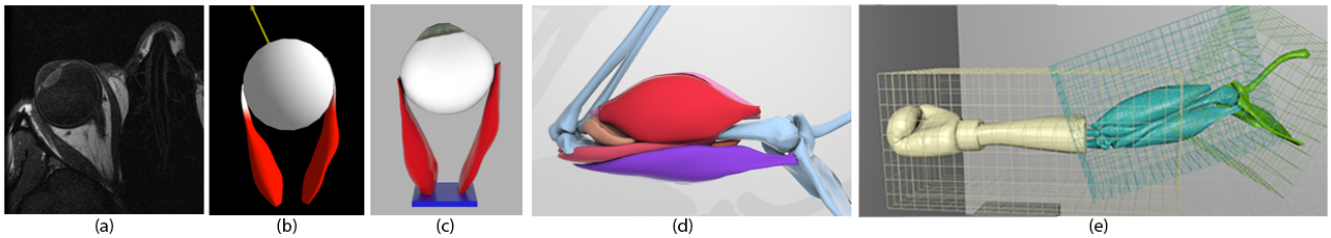


Figure 1: *Our framework spans the entire pipeline for simulating volumetric musculoskeletal systems, starting from data on the active shapes of muscles to physically based simulation of muscles and soft tissues coupled to the skeletal system. (a) MRI[1] of the eye can reveal both passive and active muscle shapes. (b) Reconstructed muscle shapes. (c) Physically based simulation of an individual's eye movement using our system. (d) Upper arm with six muscles and a bone in close sliding contact with each other. Volume preservation produces realistic lateral bulging. (e) Dynamic simulation of arm with soft tissues, a soft glove, and contact with the environment.*

Abstract

We introduce a new framework for simulating the dynamics of musculoskeletal systems, with volumetric muscles in close contact and a novel data-driven muscle activation model. Muscles are simulated using an Eulerian-on-Lagrangian discretization that handles volume preservation, large deformation, and close contact between adjacent tissues. Volume preservation is crucial for accurately capturing the dynamics of muscles and other biological tissues. We show how to couple the dynamics of soft tissues with Lagrangian multibody dynamics simulators, which are widely available. Our physiologically based muscle activation model utilizes knowledge of the active shapes of muscles, which can be easily obtained from medical imaging data or designed to meet artistic needs. We demonstrate results with models derived from MRI data and models designed for artistic effect.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation I.6.8 [Simulation and Modeling]: Types of Simulation—Animation

Keywords: Musculoskeleton, Deformation, Eulerian simulation, Muscle, Activation, Soft Tissues

1 Introduction

Our goal is to simulate biomechanically plausible movement of human "digital doubles" and fictional creatures. Such simulations are of central importance to computer animation. Skilled artists and motion capture techniques can generate high levels of realism, but such animations are very labor intensive to produce and reuse. For this reason, there has been a growing emphasis on physically based musculoskeletal simulation in computer graphics; see Sec. 2 for a review.

However, musculoskeletal simulations are extremely challenging, due to the sheer complexity of such systems. Here are some of the challenges: (1) Most tissues are **highly deformable and volumetric** solids that need 3D discretizations to fully capture significant effects. Some effects, such as the bulging of active muscles, also depend on the fact that these tissues are essentially **incompressible**. In all previous work, volumetric soft tissues are either ignored or simulated using Lagrangian finite element methods (FEM). (2) Even more challenging is the fact that muscles slide relative to each other, in close contact. Detecting and responding to **close internal contact** between 3D meshes can be expensive using standard collision processing methods. Perhaps for this reason, most musculoskeletal simulations in graphics during the last decade have avoided handling such internal sliding contact. It is also important to handle **external contact** with the environment. (3) Muscles are controlled by neural activation, but representing the behavior of **active muscles** remains challenging, despite a century of progress in muscle biomechanics. For example, the widely used "Hill-type" models [Zajac 1989] can have have large errors (about 50%), even in highly controlled laboratory settings [Herzog 2004]. The behavior of active muscles *in vivo* is even less understood, due to motor unit diversity within muscle, and other issues such as motor unit recruitment, cycling, bistability of firing rates, etc. (4) Finally, bones are connected to muscles through tendons, and to other bones through ligamentous joints. Simulation of multi-rigid body systems with bones and low dimensional joint constraints is well understood. The challenge is in simulating the dynamics of the **coupled musculoskeletal system** to move the character, and to model important non-local effects such as how the impact on a boxer's glove affects the motion of the entire body.

We propose a new framework for musculoskeletal simulation in computer animation that addresses all of the above challenges. A key to our approach is **Eulerian** (and Eulerian-on-Lagrangian) discretization of solids; this allows efficient handling of large deformations, collision detection, and simulation of many contacting mus-

*e-mail:{yefan,jlitven,pai}@cs.ubc.ca

[1]MRI data courtesy of Dr. J. L. Demer.

cles on a single, uniform grid. We introduce new methods for volume preserving deformation and for efficiently handling close internal contact. We propose a practical solution to the problem of muscle activation, using a data-driven approach. Using our approach, active shapes of muscles can be estimated from non-invasive medical imaging data, such as MRI scans, or provided by an artist if needed. The shapes are used directly to activate muscles, in a physically based musculoskeletal simulation. We demonstrate the results with a prototype implementation and several test examples. We believe this is the only existing simulation framework that has all the features highlighted in bold font above.

2 Related Work

Simulating musculoskeletal systems, with or without 3D volumetric muscles, has been an active research area in a multitude of disciplines, from biomechanics to computer graphics. Please see Lee, et al. [2012] for a recent survey of this area from a computer graphics perspective. We mention only some examples here. Other articles, related to specific technical aspects of our work such as volume preservation and muscle activation, are described in later sections which provide the appropriate context.

Early work included that of Chen and Zeltzer [1992], who first introduced muscle simulation to character animation. In more recent work, Ng-Thow-Hing and Fiume [2002] introduced B-Spline solids with anisotropic properties and volume preservation. Teran et al. [2003; 2005] built skeletal muscles from the Visible Human Data Set and simulated them using the finite element method (FEM); Lee et al. [2009] simulated the entire human upper body using quasistatic FEM and line-based muscles; Sueda et al. [2008] simulated the dynamics of thin musculotendons in an anatomy-based hand. In addition to the work cited above that used anatomical data bases, there has been work on creating subject specific anatomical models [Schmid et al. 2009; Gilles et al. 2010]. Anatomical data have been used to infer tissue material properties (e.g., [Teran et al. 2005; Lee et al. 2009]) but no previous work, to our knowledge, has used anatomical data of active shapes to activate muscles as we do.

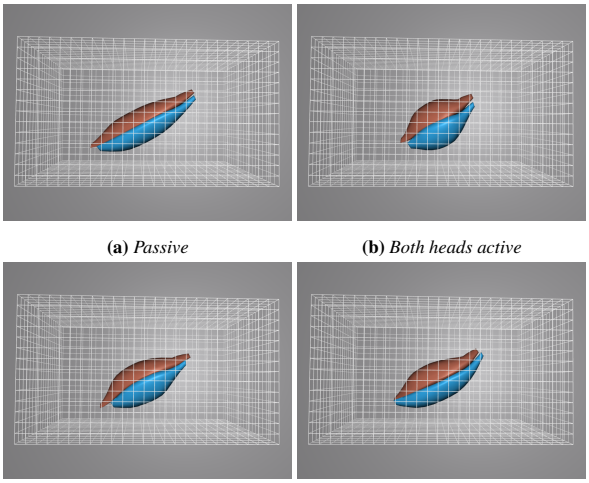
There is a large body of work in musculoskeletal simulation in biomechanics (e.g., [Blemker and Delp 2005]). Zajac [1989] provides an extensive review of muscle and tendon models that are commonly used in biomechanics. High quality open source software is available (e.g., [Delp et al. 2007]). Most of these models use non-volumetric muscles, and do not fully account for muscle mass [Pai 2010]. Such models have been used for controlling computer animations; [Geijtenbeek et al. 2013] is a notable recent example.

All work on volumetric muscle simulation builds upon the broad foundations of computational methods for solid mechanics. The seminal work of Terzopoulos et al. [1987] introduced this field to computer graphics. See [Nealen et al. 2006] for a survey and [Sifakis and Barbic 2012] for a tutorial introduction. All this work is based on Lagrangian discretization of solids. The Eulerian discretizations that we use have been introduced to graphics for simulating solids in 3D [Levin et al. 2011a; Fan et al. 2013], 2D [Li et al. 2013], and 1D [Sueda et al. 2011]. However, none of this work enforced volume preservation, close contact without the use of reduced coordinates, and muscle activation; these features are essential for simulating musculoskeletal systems.

3 Eulerian Tissue Simulation

We use an Eulerian-on-Lagrangian discretization [Fan et al. 2013] of soft tissues in each limb segment, which has many benefits for musculoskeletal simulation. We briefly summarize the Eulerian approach here for completeness; full details can be found in the original articles [Levin et al. 2011a; Fan et al. 2013].

A deformation is map, $\phi : \boldsymbol{X} \mapsto \boldsymbol{x}$, from material space to physical space. Following a common abuse of terminology (or synecdoche), we use $\boldsymbol{X}$, etc., to refer to both a space and a point in that space. The Lagrangian methods almost universally used in solid mechanics discretize material space. In contrast, Eulerian methods discretize physical space, and are widely used in fluids simulation. For Eulerian solids, material coordinates $\boldsymbol{X}$ are advected to keep track of material deformation. That is, motion is tracked using a discrete representation of the inverse mapping ϕ^{-1} (see Figure 3). The map is then used to compute the deformation gradient $\boldsymbol{F}$ and, ultimately, the elastic forces within the solid [Levin et al. 2011a]. One benefit of Eulerian discretization of solids is the ease of handling large deformations, similar to the fluids case.



(a) *Passive* **(b)** *Both heads active* **(c)** *Blue head active* **(d)** *Brown head active*

Figure 2: *Eulerian muscle simulation. A simplified example is shown of the biceps muscle, with two heads in close contact. The grid shown has half the resolution of the simulation grid, for clarity.*

Another major benefit of this approach is the simplicity of collision detection and resolution on the spatial grid (see Fig. 2). All objects are simulated on the same grid, and each object defines an indicator function $I(\phi^{-1}(\boldsymbol{x}))$ on the grid, which records whether $\boldsymbol{x}$ is occupied by the object. This is used to estimate the contact constraint normals $\mathbf{g}$, that are then assembled into the constraint Jacobian matrix $\mathbf{G}$. In the general case of objects in intermittent contact described in [Levin et al. 2011a], after time discretization of the accelerations at the velocity level and using Gauss's principle of least constraint, the dynamics of a deformable object system subject to contact constraints reduces to the solution of a convex quadratic program (QP):

$$\begin{aligned} \mathbf{v}^{t+1} = \arg\min_{\mathbf{v}} & \frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v} - \mathbf{v}^T\mathbf{f}^* \\ \text{s.t. } & \mathbf{G}\mathbf{v} \leq \mathbf{b}, \end{aligned} \tag{1}$$

where $\mathbf{v}$ is the velocity on the grid, $\mathbf{M}$ is a generalized mass matrix, and $\mathbf{f}^*$ is a generalized impulse. The velocities and material coordinates $\boldsymbol{X}$ are then advected on the grid using the velocity field to obtain the new deformed state of the solid.

4 Muscle Activation

When an active muscle contracts, there is an internal reorganization of its mass mediated by molecular motors which cause my-

ofilaments to slide relative to each other. As mentioned in Sec. 1, the actual dynamics of active muscle is complex and not very well understood. Worse, there are significant differences between the properties of different muscles and between individuals. Rather than attempt to model these details, we seek a practical but realistic activation model for use in graphics.

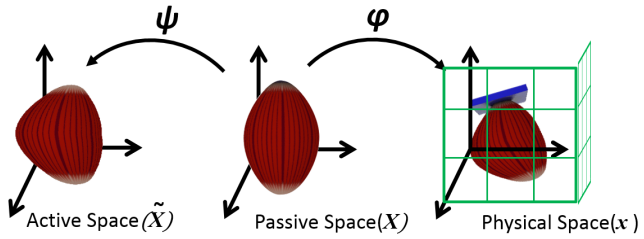


Figure 3: *The mapping between active space, passive space and physical space. Eulerian methods discretize physical space.*

The essential idea of our activation model is to represent the change in material configuration directly. See Figure 3. Suppose the reference activation (normalized to 1) changes the material configuration from $\boldsymbol{X}$ to $\tilde{\boldsymbol{X}}$. The material space without any activation is called the *passive space*; the space after active contraction is called the *active space*. We further define the mapping ψ which transforms points in $\boldsymbol{X}$ to $\tilde{\boldsymbol{X}}$. The chain of mappings from physical space to active space is

$$\tilde{\boldsymbol{X}} = \boldsymbol{\psi}(\boldsymbol{\phi}^{-1}(\boldsymbol{x})). \tag{2}$$

The deformation gradient $\boldsymbol{F} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}$ is the local measure of object deformation. When the muscle is activated, its new elastic deformation gradient $\tilde{\boldsymbol{F}}$ is

$$\tilde{\boldsymbol{F}} = \frac{\partial \boldsymbol{x}}{\partial \tilde{\boldsymbol{X}}} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\psi}(\boldsymbol{\phi}^{-1}(\boldsymbol{x}))} = \left(\frac{\partial \boldsymbol{\psi}(\boldsymbol{\phi}^{-1}(\boldsymbol{x}))}{\partial \boldsymbol{x}}\right)^{-1}. \tag{3}$$

Note that $\boldsymbol{X}$ and $\tilde{\boldsymbol{X}}$ do not need to have the same discretization as the simulation mesh in $\boldsymbol{x}$, as long as the mappings are well defined. Specifically, for a particular point $\boldsymbol{x}$, the method only requires its corresponding position in $\tilde{\boldsymbol{X}}$ be found through the mappings. This allows high resolution discretization for the active configuration space regardless of the simulation mesh.

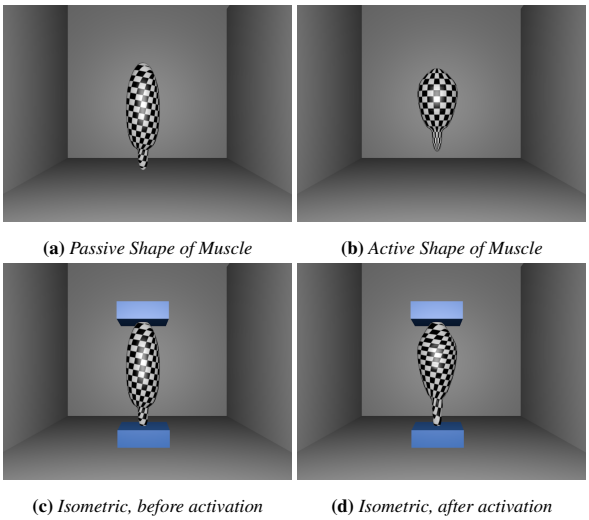
To partially activate a muscle we interpolate $\tilde{\boldsymbol{X}}$ using a one-dimensional generalized activation function $\mathcal{G}$, with the partially activated configuration $\boldsymbol{X} + \mathcal{G}(t, \tilde{\boldsymbol{X}}, \boldsymbol{X})$. Since the muscles in our examples are are shortened with very little rotation, we use linear interpolation in our current implementation, i.e., $\mathcal{G}(t, \tilde{\boldsymbol{X}}, \boldsymbol{X}) = \mathcal{G}(t)(\tilde{\boldsymbol{X}} - \boldsymbol{X})$. $\mathcal{G}(t)$ is a continuous, preferably smooth, function of time that avoids sudden discontinuities. If there is significant rotation during activation, this could be handled by the standard approach of factoring out rotation and interpolating the remainder, similar to the approach used in co-rotational elasticity.

Applying the chain rule to Equation 3 reveals another interesting interpretation of the method.

$$\tilde{\boldsymbol{F}} = \frac{\partial \boldsymbol{x}}{\partial \tilde{\boldsymbol{X}}} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} \frac{\partial \boldsymbol{X}}{\partial \tilde{\boldsymbol{X}}} \tag{4}$$

Equation 4 resembles the multiplicative decomposition law that is widely used in elasto-plastic simulation, and has roots in both computer graphics and muscle biomechanics. In graphics, $\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}$ is the total deformation gradient and $\frac{\partial \boldsymbol{X}}{\partial \tilde{\boldsymbol{X}}}$ is known as the inverse plastic deformation gradient, which evolves over time. Changing the reference shapes for plasticity simulation has been introduced to computer graphics [Bargteil et al. 2007; Wicke et al. 2010; Fan et al. 2013]. Several authors have also used target poses or rest shapes for animating and controlling arbitrary soft bodies [Coros et al. 2012; Schumacher et al. 2012; Liu et al. 2013]. In muscle biomechanics, multiplicative plasticity laws have been used to simulate deformation of active muscle [Nardinocchi and Teresi 2007], which provides some biological support for our model.

The activation model is a gross simplification of how muscles actually work, but has one significant advantage: we can exploit biomedical imaging data (or an artist's knowledge) of real muscle behavior. This advantage is similar to the use of image based rendering, motion capture, and other data-driven methods, in that the realism is baked into the data, even if the model is simple. Muscle shapes can be reconstructed from MRI in some circumstances, such as the presence of surrounding fat or with the use of contrast agents. Ultrasound is also useful for muscle imaging. We have successfully simulated human extraocular muscles and its activation using muscle shapes reconstructed from MRI data [Wei and Pai 2008] (see Figure 10). Modelers are not restricted to biomedical data, and have the option of freely designing desired active shapes. One example, designed by us (not real artists), is shown in Figure 4.



(a) *Passive Shape of Muscle* **(b)** *Active Shape of Muscle*

(c) *Isometric, before activation* **(d)** *Isometric, after activation*

Figure 4: *Muscle activation using user specfied shapes. (a) and (b) show user input data on the passive and active shapes of a muscle. Both ends of the muscle were fixed (c) and activated (d). Even when isometrically activated, muscle mass moves realistically towards the origin of the muscle (the upper end), while elastically stretching to match the endpoint constraints.*

5 Volume Preservation

Muscles, like most biological tissues, are mostly composed of water and are nearly incompressible. Volume preservation in muscles is important for capturing visual effects such as bulging and the classical "squash and stretch". In solid simulation, resistance to volume change is typically approximated by tuning constitutive properties (specifically, Poisson's ratio). This is effective for some virtual soft objects, but as Poisson's ratio approaches 0.5, it produces a numerical difficulty known as the "locking phenomenon". The classical solution is to use a mixed formulation [Hughes 2000] with carefully chosen elements or with a stabilization term [Hughes 2000; Misztal et al. 2012]. Earlier work [Ng-Thow-Hing et al. 2001] simulated muscles by approximating this volume-preserving constraint using spring-like forces based on volume change. Others follow

the quasi-incompressibility method [Simo and Taylor 1991], with finite element methods for muscle simulation [Weiss et al. 1996; Teran et al. 2005; Patterson et al. 2012].

In contrast, we treat volume preservation directly, as a constraint to be satisfied along with the dynamics. This is similar to the treatment in fluid mechanics and to the approach of Irving, et al.[2007] for solids and the strain-limiting methods used for cloth simulation (e.g., [Goldenthal et al. 2007]). Our formulation adds a stabilization term that reduces drift from the constant volume constraint.

The determinant J of the deformation gradient $\boldsymbol{F}$ defines the volume change as

$$dv = JdV \tag{5}$$

For volume-preservation we require that volume in the (active) material domain is equal to the volume in the spatial domain, $dV = dv$. By the definition of J, we require $J = 1$. This looks simple, but is non-linear constraint. In order to simulate the muscles that have large deformations, we can take the time derivative of both sides of $J = 1$, and enforce the constraint $\dot{J} = 0$. The physical interpretation is that the volume change rate at every time step is zero. That is,

$$\dot{J} = J\,\nabla\cdot\boldsymbol{v} = 0 \tag{6}$$

If $J \neq 0$ (i.e., the element is not flattened), the constraint $\dot{J} = 0$ is equivalent to enforcing a divergence-free velocity field. The constrain $\nabla\cdot\boldsymbol{v} = 0$ is commonly solved by a pressure projection step in fluid animation. Unfortunately this constraint doesn't track positional variables, and accumulated numerical drift may cause significant volume change. To combat drift, we can use a stabilization technique that enforces the time integral of $\dot{J}$ be zero:

$$0 = \int_0^{t_{n+1}} \dot{J} = \int_0^{t_n} \dot{J} + \int_{t_n}^{t_{n+1}} \dot{J} = J_n - 1 + \int_{t_n}^{t_{n+1}} J\,\nabla\cdot\boldsymbol{v} \tag{7}$$

where t_n denotes the current time, and we assume $J_0 = 1$. It is straightforward to compute J_n based on $\boldsymbol{X}$. To numerically integrate $J\nabla\cdot\boldsymbol{v}$ over time, from t_n to t_{n+1}, we use a first order method that is explicit in the position variable J and implicit in velocity $\boldsymbol{v}$

$$\int_{t_n}^{t_{n+1}} J\nabla\cdot\boldsymbol{v} = \mathrm{dt} J_n \nabla\cdot\boldsymbol{v}_{n+1} \tag{8}$$

Substituting Equation 8 back to Equation 7, and we get

$$\nabla\cdot\boldsymbol{v}_{n+1} = \frac{1}{\mathrm{dt}}\frac{1-J_n}{J_n} \tag{9}$$

This constraint can be interpreted as a Newton-like step that attempts to zero the error in J in one step. This may be more aggressive than necessary, and lead to large velocity changes when using small time steps. As with Newton's method, we can take a *damped volume preservation* step,

$$\nabla\cdot\boldsymbol{v}_{n+1} = \frac{\gamma}{\mathrm{dt}}\frac{1-J_n}{J_n} \tag{10}$$

with a damping factor $\gamma \in (0, 1]$ that controls how rapidly the volume error is reduced. As can be seen in Sec. 8, our system enforces volume-preservation well, while resolving other constraints simultaneously.

6 Close Contact

A significant challenge is to efficiently represent the interactions between adjacent muscles. Biological tissues are packed closely together in the body, held together by collageneous connective tissues. However, even though most work in computer graphics (e.g., [Lee et al. 2009; Faure et al. 2011]) models the body as soft tissues with complex anisotropic material properties, connective tissues allow muscles to slide relative to each other. There is some force transmission between adjacent synergistic muscles along their lengths, this is small in the physiological range [Maas and Sandercock 2010]. Indeed, many muscles that are thought of as single muscles, such as the biceps and triceps muscles of the arm, should be modeled as multiple muscles in close contact, with relative sliding. Their heads ("-ceps") have different origins and their mechanical actions vary based on joint angles. However, detecting and responding to such close contact over large surface areas is a major challenge using standard contact processing methods originally developed for sparse and intermittent contact between rigid bodies. We address both the challenges of appropriate modeling and

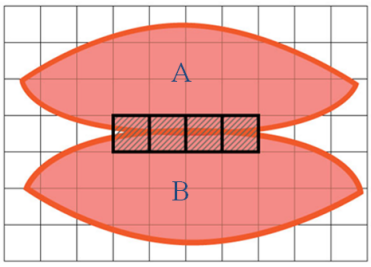


Figure 5: *Two muscles A and B in close contact. Four close contact constraints are formulated in the four shaded cells.*

efficient computation by handling close internal contact differently from intermittent external contacts.

Our contact formulation is similar to that of Levin, et al. [2011a], summarized in Equation 1. Unknowns $\boldsymbol{X}$ and $\dot{\boldsymbol{X}}$ are defined on grid nodes and each close contact constraint is formulated on one grid cell. First, before simulation, we tag pairs of objects as being in close contact. For biological tissues, this prior knowledge is available from anatomy (e.g, the long and short heads of biceps are always in contact, so this pair is a good candidate for close contact). For Eulerian simulation, voxel based collision detection is straightforward (Sec. 3); to get subgrid precision, we subsample every cell to find the ones that both objects occupy. For those cells, one close contact constraint is formulated per cell. In Figure 5, four cells are detected and four constraints are formulated between objects A and B. Specifically, the constraint is written as

$$\boldsymbol{v}^{\mathrm{rel}} \cdot \boldsymbol{g} = 0 \tag{11}$$

where $\boldsymbol{v}^{\mathrm{rel}} = \boldsymbol{v}_\Gamma^A - \boldsymbol{v}_\Gamma^B$, Γ is the contact area, and $\boldsymbol{g}$ is the contact normal. The physical meaning of Equation 11 is that the projection of the two objects' velocities at Γ onto the surface normal direction is the same. The main difference with Eq. 1 is replacing inequalities with equality constraints. In other words, we assume that the objects' relative motion can only be fixed or sliding. In this sense, our method is in the same spirit as [Sueda et al. 2011], where they have assumed that contact always happens at Eulerian nodes, but in our case the constrained nodes are computed dynamically. This assumption brings two big advantages. First, it significantly improves the efficiency of the solver. By assuming all internal contact constraints are active, the contact LCP reduces to solving a linear system. Note that solving a LCP is still necessary for external (intermittent) contact resolution. Second, close contact eliminates the need for connective tissue simulation, and prevents muscles from separating. We use Γ_C to denote the contact surface inside of a cell

Ω_C; $\Gamma_C = \Gamma \cap \Omega_C$. For one constraint, the discretized version of Equation 11 is

$$\int_{\Gamma_C} \left(\Sigma_i \boldsymbol{N}_i \boldsymbol{v}_i^A - \Sigma_i \boldsymbol{N}_i \boldsymbol{v}_i^B \right) \cdot \boldsymbol{g} \ \mathrm{d}\Gamma_C = 0 \tag{12}$$

where $\boldsymbol{N}$ denotes trilinear shape functions in our implementation. Recall that two objects share the same discretization and the contact surface as well. If their shape functions are on the same order, these functions are identical. Therefore, we use $\boldsymbol{N}_i$ for both objects A and B in Equation 12. For the normal $\boldsymbol{g}$, there are multiple ways to evaluate it. One approach follows the method proposed in [Levin et al. 2011a]. Normals are evaluated by taking the gradient of the indicator function. This approach, which is used in our implementation, avoids using any mesh information. Alternatively, the method proposed in [Fan et al. 2013] is also applicable here. After the surface mesh is reconstructed, averaged normal in a cell can be taken as weighted averages of normals computed on surface triangles. These triangles can either be from one of the objects, or both. Although a triangular mesh is needed for normal evaluation, no mesh mesh collision detection is employed here. Fast collision detection is still performed by indicator function comparison.

7 Musculoskeletal System

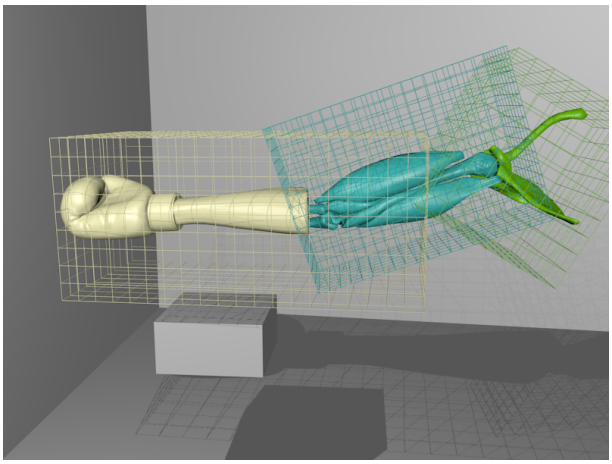


Figure 6: *Muscle, tendon and bone structure with Eulerian simulation grid associated with the skeleton*

We use a modular architecture which separates the musculoskeletal system into a skeleton-tendon subsystem and a soft tissue subsystem. This allows the use of existing multibody simulators with our soft tissue simulator. We enforce weak coupling between the subsystems to achieve this modularization. We detail the systems below.

7.1 Skeletons and Tendons

The skeleton is well approximated as a system of rigid bodies (bones) connected by joints. The dynamics of such systems has been thoroughly studied and understood, in robotics, computer graphics, and biomechanics. Therefore our description is brief. In biomechanical systems, muscles may be connected to distant bones via long stiff tendons that span multiple joints. We follow the standard practice in biomechanics (e.g., [Delp et al. 2007]) of assuming tendons are inextensible, and that their kinematics can be represented using their "moment arms" at each joint. Tendon excursions $\boldsymbol{e}$ are therefore kinematically coupled to the skeleton configuration $\boldsymbol{q}$.

Any existing multibody simulator with the above features could be used in our framework. In our implementation, we use unreduced coordinates for bones, and explicit joint constraints. Using a derivation based on Gauss's principle, as in Sec. 3, we formulate the dynamics as another QP (Eq. 16). We currently use revolute and ball-and-socket joints, though more general joints [Lee and Terzopoulos 2008] could be included easily. Data such as moment arms of each tendon are taken from [Holzbaur et al. 2005].

7.2 Attaching Soft Tissues to Skeleton

We first consider the constraints on the Eulerian soft tissue simulation due to attachments. Typically, the proximal (origin) end of a muscle attaches over a broad area of a bone, and the distal (insertion) end has a more prominent tendon. The brachialis muscle, for example, has a large area of origin on the humerus, and the distal end connects to the forearm via a tendon. For a muscle with attachment boundary Γ either at the origin or at the insertion, we enforce the Eulerian velocity on Γ to be the same as the tendon velocity $\dot{\boldsymbol{e}}$ relative to the reference bone used to model the muscle.

$$\boldsymbol{v}^{n+1}(\boldsymbol{x}) = \dot{\boldsymbol{e}} \quad \boldsymbol{x} \in S, \tag{13}$$

where $S \subseteq \Gamma$ is the attachment area. Its position in physical space is determined by excursion $\boldsymbol{e}$ and rigid body transformation. Writing $\boldsymbol{v}$ in terms of shape functions $\boldsymbol{N}_i$ and integrating Equation 13 over a grid cell Ω_c yields

$$\int_{\Gamma \cap \Omega_c} \sum_i \boldsymbol{N}_i \boldsymbol{v}_i^{n+1} \partial\Gamma = \dot{\boldsymbol{e}} \int_{\Gamma \cap \Omega_c} \partial\Gamma, \tag{14}$$

Note that in the case of muscle-bone attachments, $\dot{\boldsymbol{e}}$ on the attachment area is zero throughout the simulation as the bones are fixed with respect to the simulation grid. A muscle normally has two attachments, one for origin and the other for insertion. We globally assemble these constraints, giving

$$\boldsymbol{G}_A \boldsymbol{v} = \boldsymbol{a}, \tag{15}$$

where $\boldsymbol{G}_A$ is the attachment constraint matrix and $\boldsymbol{a}$ represents the right hand side.

Attachment constraints produce a coupling impulse $\boldsymbol{p}_A$ on bone and tendon. With this impulse included, the skeleton-tendon system's dynamics is described by the QP

$$\begin{aligned} \dot{\boldsymbol{q}}^{\boldsymbol{n+1}} = & \underset{\dot{\boldsymbol{q}}}{\text{argmin}} \quad \frac{1}{2} \dot{\boldsymbol{q}}^T \boldsymbol{M_S} \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}^T \left(\boldsymbol{p}_S + \boldsymbol{p}_A^n \right) \\ & \text{subject to} \quad \boldsymbol{G}_S \dot{\boldsymbol{q}} \leq \boldsymbol{d}, \end{aligned} \tag{16}$$

where $\boldsymbol{M_S}$ is the skeletal inertia tensor, $\boldsymbol{p}_S$ is the external impulse. Note that all quantities defined are globally assembled. The constraints in Equation 16 are written as inequalities for generality, even though most joint constraints will be equalities. Inequality constraints can enforce, for example, joint angle bounds.

Similarly, the muscle QP is

$$\begin{aligned} \boldsymbol{v}^{n+1} = & \underset{\boldsymbol{v}}{\text{argmin}} \quad \frac{1}{2} \boldsymbol{v}^T \boldsymbol{M} \boldsymbol{v} - \boldsymbol{v}^T \boldsymbol{p}_B \\ & \text{subject to} \quad \boldsymbol{G}_V \boldsymbol{v} = \boldsymbol{c}, \boldsymbol{G}_C \boldsymbol{v} \geq \boldsymbol{0}, \boldsymbol{G}_A \boldsymbol{v} = \boldsymbol{a}, \end{aligned} \tag{17}$$

where $\boldsymbol{p}_B$ is the impulse due to advection and all the forces. $\boldsymbol{M}$ is the global mass matrix, $\boldsymbol{G}_V$ is the volume preservation constraint, and $\boldsymbol{G}_C$ are external contact constraints. We simplify the numerics by lumping mass to the nodes so that $\boldsymbol{M}$ is diagonal.

After Equation 16 is solved, we get $\dot{\boldsymbol{q}}^{n+1}$ and subsequently formulate the attachment constraints using the most recent rigid body state, and solve Equation 17. Note that the coupling impulses $\boldsymbol{\lambda}_A^{n+1}$ have been implicitly computed as the Lagrange multipliers associated with the attachment constraints:

$$\boldsymbol{p}_A^{n+1} = \boldsymbol{G}_A^T \boldsymbol{\lambda}_A^{n+1}. \tag{18}$$

Once we know $\boldsymbol{p}_A^{n+1}$, we can advance to t_{n+2} by solving Equation 16 using $\boldsymbol{p}_A^{n+1}$. A high level overview of our solution procedure for a single time step is given in Algorithm 1.

Algorithm 1 Solution Procedure

Initialize $\boldsymbol{\lambda}_A^0 = \mathbf{0}$, $\boldsymbol{q}^0 = \mathbf{0}$, $\boldsymbol{v}^0 = \mathbf{0}$, $t_0 = 0$ and $n = 0$
while $t_n < t_{total}$ **do**
  // Solve for Lagrangian DOFs (skeletal multibody system)
  Compute $\boldsymbol{p}_A^n = \boldsymbol{G}_A^T \boldsymbol{\lambda}_A^n$
  Solve the skeletal QP (Eq. 16) for $\dot{\boldsymbol{q}}^{n+1}$
  Update $\boldsymbol{q}^{n+1} = \boldsymbol{q}^n + dt\dot{\boldsymbol{q}}^{n+1}$
  // Solve for Eulerian DOFs (muscles, soft tissues)
  Advect $\boldsymbol{v}^{\boldsymbol{n}}$ and compute $\boldsymbol{p}_B$
  Formulate the constraints in Equation 17 using $\boldsymbol{q}^{n+1}$
  Solve muscle QP (Eqs. 17,19) for $\boldsymbol{v}^{n+1}$ and $\boldsymbol{\lambda}_A^{n+1}$
  $\boldsymbol{X}^{n+1}$ = advect($\boldsymbol{X}^n$)
  $t_{n+1} = t_n + dt$
  $n = n + 1$
end while

7.3 Solver and Implementation

Each time step requires the solution of the skeletal and muscle QPs, given by Equation 16 and Equation 17, respectively. Both are solved via a primal-dual active-set method [Ito and Kunisch 2008]. Each iteration of the active-set method requires the solution of the following symmetric, indefinite linear system or KKT system. For the skeletal QP, the size is small and the solution is computed via a direct solver. The KKT system of the muscle QP is

$$\begin{pmatrix} \boldsymbol{M} & \boldsymbol{G}_V^T & \tilde{\boldsymbol{G}}_C^T & \boldsymbol{G}_A^T \\ \boldsymbol{G}_V & -\epsilon \boldsymbol{L} & & \\ \tilde{\boldsymbol{G}}_C & & & \\ \boldsymbol{G}_A & & & \end{pmatrix} \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{\lambda}_V \\ \boldsymbol{\lambda}_C \\ \boldsymbol{\lambda}_A \end{pmatrix} = \begin{pmatrix} \boldsymbol{p}_B \\ \boldsymbol{c} \\ \mathbf{0} \\ \boldsymbol{a} \end{pmatrix} \tag{19}$$

where $\tilde{\boldsymbol{G}}_C$ consists of rows from the contact Jacobian $\boldsymbol{G}_C$ corresponding to *active* constraints. The QP solver usually converges within five iterations since we don't have a large number of external objects in contact. To solve the resulting linear system Equation 19, any off the shelf solves can be used. In our implementation, we use a sparse direct solver based on LU factorization. Table 1 gives the running time for the full arm simulations. Over the simulations, the number of the constraints such as volume-preservation, attachment and close contact does not change much. The above statistics are representative for most of the time steps. Our implementation uses both the GPU and CPU. The solver and assembly part are done on the CPU while the rest are on GPU. Subsampling- takes a large amount of time. Evaluating the mass matrix, divergence operator and enforcing close contact all require subsampling. Currently we do not prune empty cells far away, or fully filled cells deeply inside of an object. If subsampling can be done only where needed, i.e., in partially filled cells, this will significantly improve the performance.

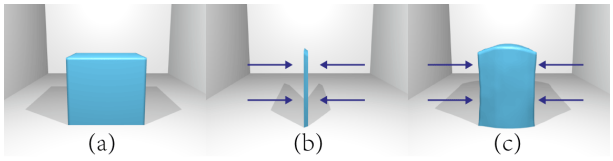


Figure 7: *A soft elastic cube is placed in a force field, with huge horizontal forces that compress the cube. The forces are depicted by arrows. (a) the original shape of the cube. (b) a compressible cube is squeezed to a sheet under the force field. (c) incompressible solid simulation using our method. The cube barely loses any volume.*

8 Results

8.1 Volume Preservation

To validate the effectiveness of the volume-preserving constraint with the positional stabilization, we crushed a soft cube using different methods. The cube was placed in a large horizontal force field. The force magnitude was uniform over the whole space, but the direction differed, see Figure 7. We have run three simulations

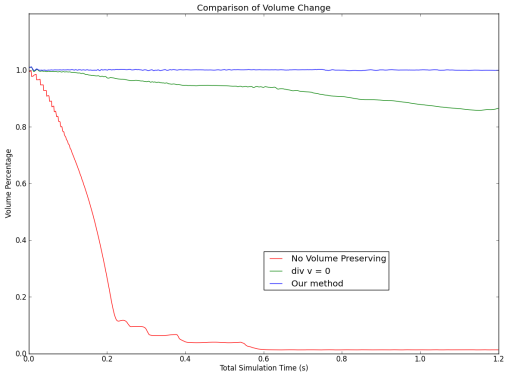


Figure 8: *The figure shows the volume change for three simulations. y-axis: object volume as a fraction; x-axis: total simulation time in seconds. The red line is for the non-volume-preserving solid while the green and blue lines are for volume preserving solids. The green one is based on the $\nabla \cdot \boldsymbol{v} = 0$ condition and the blue one is based on our method.*

for the same scenario: non-volume-preservation, velocity volume-preservation only and velocity volume-preservation with positional stabilization. The total volume change over time is shown in Figure 8 respectively. The three lines correspond to the three simulations under investigation. The non-volume-preserving object (red line) loses volume from the very beginning and is quickly smashed to a sheet. In contrast, volume-preserving solids (green and blue lines) are more resistant against volume change under the external force field. The $\nabla \cdot \boldsymbol{v} = 0$ formulation (green line) loses volume over time. Our method with the proposed positional stabilization (the blue line) exhibits almost constant volume. The maximum volume change does not exceed 0.2% of the original volume. This result shows that our method enforces volume preservation well and prevents numerical drift from accumulation.

Figure 9 demonstrates that contact and volume-preservation constraints are simultaneously resolved. We simulated two bunnies dropping to the ground at the same high speed, with one volume-preserving (left) and the other one purely elastic (right). Figure 9

Dim (upper arm)	Dim (forearm)	DOFs	Subsampling	Assembly	Muscle solver	Multibody solver	total (ms)
$7 \times 55 \times 43 \times 30$	$2 \times 42 \times 20 \times 21$	44394	1815	588	799	70	3526
$6 \times 55 \times 43 \times 30$	$2 \times 42 \times 20 \times 21$	20615	1486	456	118	66	2316

Table 1: *Statistics for a typical time step of the simulations in milliseconds. The examples are run on i7 2.67 hz CPU and GTX 660 graphics card. First row: isometric activation example with soft tissues on the upper arm. Second row: full arm dynamics example.*

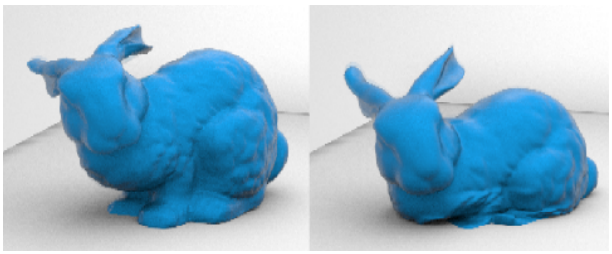


Figure 9: *Comparison of volume-preserving and non-volume-preserving bunnies dropping to the ground at the same speed. Left: volume-preserving. Right: purely elastic.*

captures the moment when the two bunnies touch the ground and undergo large deformation. It is clear by inspection that the non-volume preserving bunny has experienced a significant volume loss near the contact area, compared to the volume preserving one.

8.2 Eye muscles: from MRI to Movement

As shown in Figure 10, we simulated the extraocular muscles and the eyeball using our musculoskeletal simulator. The left pictures in Figure 10 are the raw MRI images. When looking towards the nose (bottom figure), the Medial Rectus (MR) muscle is highly active and the Lateral Rectus (LR) muscle is very relaxed. When looking outward (top figure), it's the reverse. Thus it is possible to get muscle shapes for individual muscles in different states of activation. The 3D shapes of the muscles can be reconstructed from the MRI data, for different gaze directions [Wei and Pai 2008; Wei et al. 2009]. As a first approximation, we took the most relaxed muscle shape as the passive shape of the muscle and the most active muscle shape as the active shape. Using our simulator we then simulate eye movements using these shapes for activation.

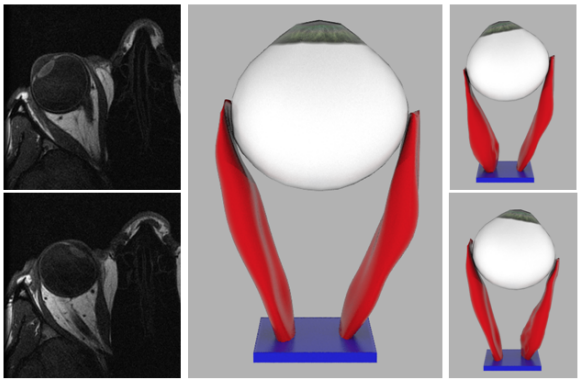
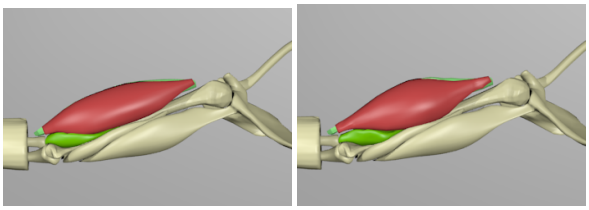


Figure 10: *Left: the MRI data of a human eye (courtesy of Dr. J. L. Demer); middle:the passive configuration of the eye muscles; right: simulations of the activated muscles in physical space*

The results, seen in the video, are very promising. The muscle shapes (seen on the right of the figure) deform realistically during movement. The rotation of the globe slightly undershoots the target. This may be due to the several approximations made: among the six extraocular muscles, only the two most useful for horizontal movement were modeled, and we didn't model other soft tissues in the eye. Other limitations are discussed in Sec. 9. Despite these limitations, it's remarkable that one can go from medical imaging data directly to realistic musculoskeltal simulation.

8.3 Isometric Activation

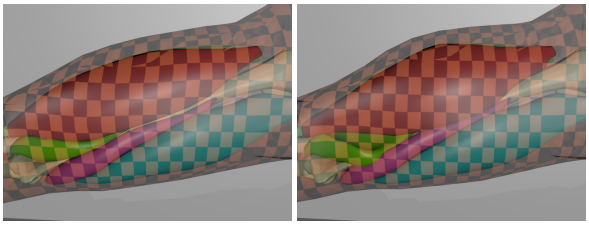


(a) *Passive* **(b)** *biceps activated*

Figure 11: *Activation of biceps and brachialis, with joints fixed. Even though there is no skeletal motion, the muscle changes shape as it should*

Many muscle models used for visual effects are driven by skeleton motion specified by an animator, but active muscles change shape even when there is no change in total length, as shown in Fig. 4. Figure 11 also illustrates this phenomenon. We fix all the bone DOFs, and then fully activate the biceps and the brachialis. Muscle deformation and activation in our method do not rely on skeleton configurations.

8.4 General Soft Tissue Simulation



(a) *Passive* **(b)** *biceps activated*

Figure 12: *Muscle wrapped in a soft tissue layer, representing fat and skin.*

Our method naturally extends to simulating tissues other than muscles. Tissues such as fat have different material properties than the muscles, but they are also volume preserving and are in close contact with muscles. We can simulate tissues the same way as we simulate muscles and add necessary constraints to them. In the example, we wrap all the muscles on the humerus using an incompressible soft tissue. That tissue is hollow inside, but has close contact constraints with all the neighbouring muscles and bones. The biceps are isometrically activated like the ones in Figure 11. Due to

incompressibility, the soft tissues above the biceps bulge appropriately. As a side effect, this unified tissue simulator outputs dynamic skin as a direct result of the simulation, and needs no additional skinning effort.

8.5 Boxer's Arm

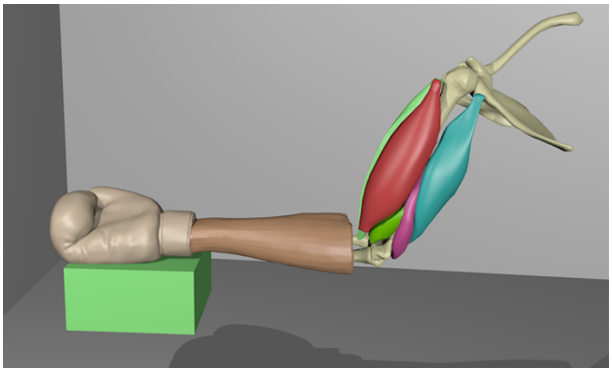


Figure 13: *The whole arm moves downwards and hits a rigid obstacle and bounces back.*

Figure 13 shows a simulation of an arm hitting a rigid body. We first activate biceps and brachialis. This makes the arm flex. After five seconds, those activated muscles are deactivated, meanwhile, all the triceps are activated. This should make the whole arm move towards the rigid obstacle at a high speed. As can be seen in the video (and Figure 13), our simulator is capable of resolving external contact while maintaining internal constraints. When contact occurs, we see the collision between the glove and the rigid box is correctly resolved locally. This contact prevents the arm from further moving downwards and causes all the muscles to deform due to inertial forces.

9 Limitations

As expected from any attempt to model a complex biological system, our work has many limitations. Perhaps the foremost is the necessity of making approximations both in the biomechanical model and in the simulation algorithms.

Our muscle activation model is clearly a major simplification, especially for partial activation, but as mentioned in Sec. 1, the alternatives have their own problems. A muscle's behavior is influenced by its fiber architecture, which is not modeled explicitly in our approach. We chose not model the architecture because the required data are extremely difficult to obtain; most previous work in this area (e.g., [Agur et al. 2003]) are not subject-specific and required painstaking cadaver dissections. More recent work could produce subject specific architecture *in vivo* using MRI (e.g., [Levin et al. 2011b]), but this work is at an early stage. To estimate active muscle shape from imaging data we currently simply use a scaled version of the *in vivo* active muscle shape, which is not stress-free. This limitation could be addressed by using the simulator to iteratively refine the stress-free active muscle shape, solving an inverse problem. Despite these limitations, the activation model works remarkably well, perhaps because the realism is baked into data (either from imaging or from an artist). It could be very useful for graphics, in the same way that texture mapping and motion capture have been; it may be better to have a bad model that can exploit good data, rather than the other way around.

Our simulations also have many limitations and inefficiencies that could be improved. We currently have one Eulerian-on-Lagrangian grid attached to each moving bone to simulate the soft tissues near each bone, similar to Chimera grids [English et al. 2013]. The transitions between grids could be removed to produce a single deformable grid using free-form deformation of the grid at the joints (similar to lattice-based Lagrangian deformation methods, e.g., [Patterson et al. 2012]); this fits easily into the Eulerian-on-Lagrangian framework of [Fan et al. 2013]. The simulation grids are also currently large and could be more tightly fitted and adapted. There are some small jiggling artifacts visible in our video results. This is partly due to observing soft tissue dynamics in slow motion, and partly due to the limitations of the numerical methods used in our current implementation. Specifically, the oscillations are probably due to explicit time integration of active elastic forces. Employing an implicit time integration scheme could address this issue. Finally, we do not address the problem of controlling the musculoskeletal system. Our focus has been on a better model of the "plant." A controller should be added on top of our system to produce interesting movements.

10 Conclusion

We have introduced a new comprehensive framework for musculoskeletal simulation. The framework addresses several limitations of previous musculoskeletal simulators, notably the ability to robustly model large numbers of volumetric muscles, sliding relative to each other in close contact. We described a stabilized volume preservation algorithm for Eulerian solids. Volume preservation is essential for simulating tissues and for producing squash-and-stretch effects. We introduced a new and practical muscle activation model that can exploit medical imaging data to not only capture anatomy but also this important physiological aspect of muscle function. With the increasing availablity and affordability of medical imaging data, we believe this approach will grow in importance for computer graphics.

Acknowledgements

This work was funded in part by grants from NSERC, CFI, and the Canada Research Chairs Program. We would like to thank Dr. Qi Wei for the 3D models of the orbit and Dr. Joseph L. Demer for the MRI data (supported by National Eye Institute grant EY08313) described in Section 8.2.

References

AGUR, A. M., NG-THOW-HING, V., BALL, K. A., FIUME, E., AND MCKEE, N. H. 2003. Documentation and three-dimensional modelling of human soleus muscle architecture. *Clinical Anatomy 16*, 4, 285–293.

BARGTEIL, A. W., WOJTAN, C., HODGINS, J. K., AND TURK, G. 2007. A Finite Element Method for Animating Large Viscoplastic Flow. *ACM Trans. Graph. 26*, 3 (July), 16:1–16:8.

BLEMKER, S. S., AND DELP, S. L. 2005. Three-dimensional representation of complex muscle architectures and geometries. *Annals of Biomedical Engineering 33*, 5 (May), 661–673.

CHEN, D. T., AND ZELTZER, D. 1992. Pump it up: Computer animation of a biomechanically based model of muscle using the finite element method. SIGGRAPH '92, 89–98.

COROS, S., MARTIN, S., THOMASZEWSKI, B., SCHUMACHER, C., SUMNER, R., AND GROSS, M. 2012. Deformable objects alive! *ACM Transactions on Graphics (TOG) 31*, 4, 69.

DELP, S. L., ET AL. 2007. OpenSim: open-source software to create and analyze dynamic simulations of movement. *IEEE Trans Biomed Eng 54*, 1940–1950.

ENGLISH, R. E., QIU, L., YU, Y., AND FEDKIW, R. 2013. Chimera grids for water simulation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 85–94.

FAN, Y., LITVEN, J., LEVIN, D. I. W., AND PAI, D. K. 2013. Eulerian-on-lagrangian simulation. *ACM Trans. Graph. 32*, 3 (July), 22:1–22:9.

FAURE, F., GILLES, B., BOUSQUET, G., AND PAI, D. K. 2011. Sparse meshless models of complex deformable solids. In *ACM Transactions on Graphics (TOG)*, vol. 30, ACM, 73.

GEIJTENBEEK, T., VAN DE PANNE, M., AND VAN DER STAPPEN, A. F. 2013. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics (TOG) 32*, 6, 206.

GILLES, B., REVERET, L., AND PAI, D. 2010. Creating and animating subject-specific anatomical models. In *Computer Graphics Forum*, vol. 29, Wiley Online Library, 2340–2351.

GOLDENTHAL, R., HARMON, D., FATTAL, R., BERCOVIER, M., AND GRINSPUN, E. 2007. Efficient simulation of inextensible cloth. *ACM Transactions on Graphics (TOG) 26*, 3, 49.

HERZOG, W. 2004. History dependence of skeletal muscle force production: implications for movement control. *Hum Mov Sci 23* (Nov), 591–604.

HOLZBAUR, K., MURRAY, W., AND DELP, S. 2005. A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control. *Annals of biomedical engineering 33*, 6, 829–840.

HUGHES, T. 2000. *The finite element method: linear static and dynamic finite element analysis*. Dover Publications.

IRVING, G., SCHROEDER, C., AND FEDKIW, R. 2007. Volume conserving finite element simulations of deformable models. *ACM Transactions on Graphics (TOG) 26*, 3, 13.

ITO, K., AND KUNISCH, K. 2008. Lagrange Multiplier Approach to Variational Problems and Applications.

LEE, S.-H., AND TERZOPOULOS, D. 2008. Spline joints for multibody dynamics. In *ACM Transactions on Graphics (TOG)*, vol. 27, ACM, 22.

LEE, S., SIFAKIS, E., AND TERZOPOULOS, D. 2009. Comprehensive biomechanical modeling and simulation of the upper body. *ACM Transactions on Graphics (TOG) 28*, 4, 99.

LEE, D., GLUECK, M., KHAN, A., FIUME, E., AND JACKSON, K. 2012. Modeling and simulation of skeletal muscle for computer graphics: A survey. *Foundations and Trends in Computer Graphics and Vision 7*, 4, 229–276.

LEVIN, D. I. W., LITVEN, J., JONES, G. L., SUEDA, S., AND PAI, D. K. 2011. Eulerian solid simulation with contact. *ACM Transactions on Graphics (TOG) 30*, 4, 36.

LEVIN, D. I. W., GILLES, B., MÄDLER, B., AND PAI, D. K. 2011. Extracting skeletal muscle fiber fields from noisy diffusion tensor data. *Medical Image Analysis 15*, 3, 340–353.

LI, D., SUEDA, S., NEOG, D. R., AND PAI, D. K. 2013. Thin skin elastodynamics. *ACM Transactions on Graphics (TOG) 32*, 4, 49.

LIU, L., YIN, K., WANG, B., AND GUO, B. 2013. Simulation and control of skeleton-driven soft body characters. *ACM Transactions on Graphics (TOG) 32*, 6, 215.

MAAS, H., AND SANDERCOCK, T. G. 2010. Force transmission between synergistic skeletal muscles through connective tissue linkages. *J. Biomed. Biotechnol. 2010*, 575–672.

MISZTAL, M., ERLEBEN, K., BARGTEIL, A., FURSUND, J., CHRISTENSEN, B., BÆRENTZEN, J., AND BRIDSON, R. 2012. Multiphase flow of immiscible fluids on unstructured moving meshes. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 97–106.

NARDINOCCHI, P., AND TERESI, L. 2007. On the active response of soft living tissues. *Journal of Elasticity 88*, 1, 27–39.

NEALEN, A., MÜLLER, M., KEISER, R., BOXERMAN, E., AND CARLSON, M. 2006. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, vol. 25, 809–836.

NG-THOW-HING, V., AND FIUME, E. 2002. Application-specific muscle representations. In *Graphics Interface*, vol. 2, Citeseer, 107–16.

NG-THOW-HING, V., AGUR, A., AND MCKEE, N. 2001. A muscle model that captures external shape, internal fibre architecture, and permits simulation of active contraction with volume preservation. In *INTERNATIONAL SYMPOSIUM ON COMPUTER METHODS IN BIOMECHANICS & BIOMEDICAL ENGINEERING (5.: 2001: Rome). Anais. Rome*.

PAI, D. K. 2010. Muscle mass in musculoskeletal models. *J. Biomechanics 43*, 11 (August), 2093–2098. DOI: 10.1016/j.jbiomech.2010.04.004.

PATTERSON, T., MITCHELL, N., AND SIFAKIS, E. 2012. Simulation of complex nonlinear elastic bodies using lattice deformers. *ACM Transactions on Graphics (TOG) 31*, 6, 197.

SCHMID, J., SANDHOLM, A., CHUNG, F., THALMANN, D., DELINGETTE, H., AND MAGNENAT-THALMANN, N. 2009. Musculoskeletal simulation model generation from mri data sets and motion capture data. In *Recent Advances in the 3D Physiological Human*. Springer, 3–19.

SCHUMACHER, C., THOMASZEWSKI, B., COROS, S., MARTIN, S., SUMNER, R., AND GROSS, M. 2012. Efficient simulation of example-based materials. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, 1–8.

SIFAKIS, E., AND BARBIC, J. 2012. Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Course Notes*.

SIMO, J. C., AND TAYLOR, R. L. 1991. Quasi-incompressible finite elasticity in principal stretches. continuum basis and numerical algorithms. *Computer Methods in Applied Mechanics and Engineering 85*, 3, 273–310.

SUEDA, S., KAUFMAN, A., AND PAI, D. K. 2008. Musculotendon simulation for hand animation. *ACM Transactions on Graphics (TOG) 27*, 3, 83.

SUEDA, S., JONES, G. L., LEVIN, D. I., AND PAI, D. K. 2011. Large-scale dynamic simulation of highly constrained strands. In *ACM Transactions on Graphics (TOG)*, vol. 30, ACM, 39.

TERAN, J., BLEMKER, S., HING, V., AND FEDKIW, R. 2003. Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 68–74.

TERAN, J., SIFAKIS, E., BLEMKER, S., NG-THOW-HING, V., LAU, C., AND FEDKIW, R. 2005. Creating and simulating skeletal muscle from the visible human data set. *Visualization and Computer Graphics, IEEE Transactions on 11*, 3, 317–328.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, vol. 21, ACM, 205–214.

WEI, Q., AND PAI, D. 2008. Physically consistent registration of extraocular muscle models from MRI. In *Proc. IEEE Engineering in Medicine and Biology Society (EMBS) Annual Conference*, 2237–2241.

WEI, Q., SUEDA, S., MILLER, J. M., DEMER, J. L., AND PAI, D. K. 2009. Template-based reconstruction of human extraocular muscles from magnetic resonance images. In *ISBI'09. IEEE International Symposium on Biomedical Imaging*, IEEE, 105–108.

WEISS, J. A., MAKER, B. N., AND GOVINDJEE, S. 1996. Finite element implementation of incompressible, transversely isotropic hyperelasticity. *Computer methods in applied mechanics and engineering 135*, 1, 107–128.

WICKE, M., RITCHIE, D., KLINGNER, B. M., BURKE, S., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2010. Dynamic local remeshing for elastoplastic simulation. *ACM Trans. Graph. 29* (July), 49:1–49:11.

ZAJAC, F. E. 1989. Muscle and tendon: Properties, models, scaling, and application to biomechanics and motor control. *CRC Critical Reviews of Biomedical Engineering 17*, 4, 359–411.