Dexterous Manipulation and Control with Volumetric Muscles

SEUNGHWAN LEE, RI YU, and JUNGNAM PARK, Seoul National University
MRIDUL AANJANEYA, Rutgers University
EFTYCHIOS SIFAKIS, University of Wisconsin-Madison
JEHEE LEE, Seoul National University

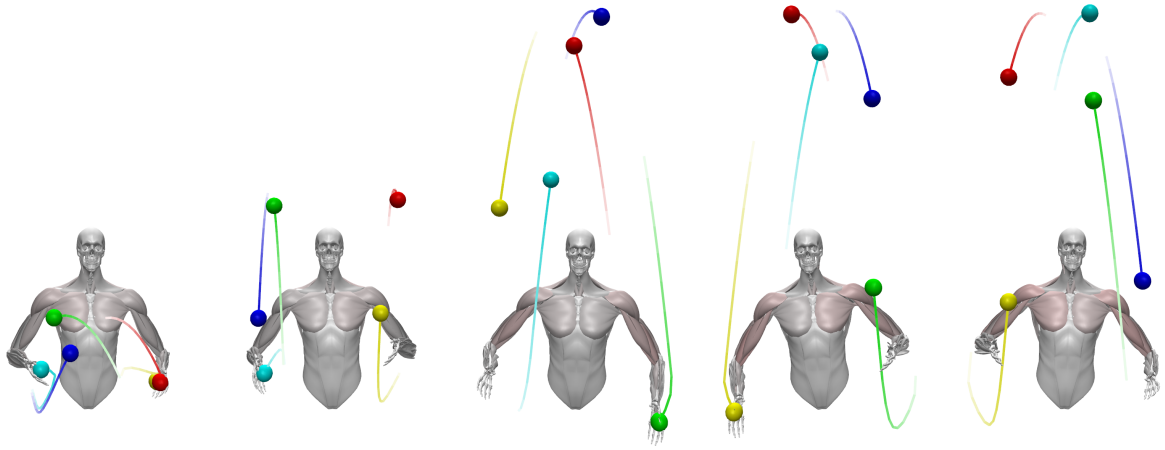


Fig. 1. Demonstration of motion control in an anatomical simulation of a juggling task, actuated by volumetric muscles. Active muscles are shown in pink.

We propose a framework for simulation and control of the human musculoskeletal system, capable of reproducing realistic animations of dexterous activities with high-level coordination. We present the first controllable system in this class that incorporates volumetric muscle actuators, tightly coupled with the motion controller, in enhancement of line-segment approximations that prior art is overwhelmingly restricted to. The theoretical framework put forth by our methodology computes all the necessary Jacobians for control, even with the drastically increased dimensionality of the state descriptors associated with three-dimensional, volumetric muscles. The direct coupling of volumetric actuators in the controller allows us to model muscular deficiencies that manifest in shape and geometry, in ways that cannot be captured with line-segment approximations. Our controller is coupled with a trajectory optimization framework, and its efficacy is demonstrated in complex motion tasks such as juggling, and weightlifting sequences with variable anatomic parameters and interaction constraints.

Authors' addresses: Seunghwan Lee; Ri Yu; Jungnam Park, Department of Computer Science and Engineering, Seoul National University, lsw9021@mrl.snu.ac.kr, yuri@mrl.snu.ac.kr, jungnam04@mrl.snu.ac.kr; Mridul Aanjaneya, Department of Computer Science, Rutgers University, mridul.aanjaneya@rutgers.edu; Eftychios Sifakis, Computer Science Department, University of Wisconsin-Madison, sifakis@cs.wisc.edu; Jehee Lee, Department of Computer Science and Engineering, Seoul National University, jehee@mrl.snu.ac.kr.

0730-0301/2018/8-ART57
https://doi.org/10.1145/3197517.3201330

CCS Concepts: • **Computing methodologies** → **Physical simulation**; **Motion processing**;

Additional Key Words and Phrases: Motion control, Volumetric Muscles, Musculoskeletal simulation, Trajectory Optimization

ACM Reference Format:
Seunghwan Lee, Ri Yu, Jungnam Park, Mridul Aanjaneya, Eftychios Sifakis, and Jehee Lee. 2018. Dexterous Manipulation and Control with Volumetric Muscles . *ACM Trans. Graph.* 37, 4, Article 57 (August 2018), 13 pages. https://doi.org/10.1145/3197517.3201330

1 INTRODUCTION

Complex movement of a human body emerges from the geometrical structure of the musculoskeletal system and its mechanical characteristics. The skeleton supports the body, and the muscles contract to induce motion of the attached bones. The human brain coordinates the activations of the muscles in harmonious synergy, to create intended bulk motion or maintain kinematic balance. In Computer Graphics, there is a strong research trail of methods for reproducing natural human motion by incorporating the true mechanics of the musculoskeletal system. However, common modeling assumptions such as the near-ubiquitous adoption of line-segment primitives for the actuation of such systems raise a number of important questions: First, it is not well understood what impact such simplifying assumptions may have on the accuracy and biomechanical fidelity of the simulations thus produced. Second, it is unclear if we could forego such simplification by just accepting an increase in the computational cost, or whether current control formulations would be challenged to accommodate more complex, volumetric actuators.

Finally, it is reasonable to question whether the incorporation of volumetric muscle primitives with control techniques can cope with the complexity of regenerating highly coordinated dexterous skills.

The Hill-type muscle model [Thelen et al. 2003; Zajac 1989] has been broadly adopted to encode the nonlinear contraction dynamics of muscle in fields such as ergonomics, computer graphics, and robotics. However, almost every prior attempt at *tightly coupled simulation and control* of the musculoskeletal system has resorted to simplifying muscles into sequences of line segments (some of which are capable of active contraction), consciously neglecting the geometrical structure of volumetric muscles (e.g. cross-sectional geometry and active fiber field distribution). In addition, various physical parameters such as pennation angle, maximum force, and way-points in the Hill-type muscle model have to be tuned according to the actual muscles, in order to empirically match the physiological behavior of observed human motion.

Prior work has progressed towards tightly integrated volumetric musculature simulation and control, but has stopped just short of achieving this goal. Lee et al. [2009] demonstrated a volumetric musculoskeletal simulation, but derived bone kinematics and muscle activations from a separate line-segment simulation and controller. In the work of Si et al. [2014], a volumetric simulation is primarily used for visualization and to mediate force transfer from an aquatic environment to the skeleton, while the controller still uses line-segment muscles. The work of Fan et al. [2014] demonstrates the impressive accomplishment of stable articulation of the skeleton based on the action of volumetric primitives; nevertheless, control is consciously left outside the scope of this work. Volumetric muscles have been used in tight integration with control in the realm of facial animation [Sifakis et al. 2005], but without directly articulating the mandible, nor including any other skeletal bones. We claim that this paper presents the first work in musculoskeletal simulation in the domain of computer graphics that successfully incorporates volumetric muscles in tight integration with a motion controller.

The Finite Element Method (FEM) has been used in a large segment of prior work on anatomical simulation, and it affords a broad spectrum of constitutive models to convey the mechanics of biomaterials. In our work, however, we use a Corotated Elasticity formulation for the background isotropic elasticity of muscle tissue (in line with prior approaches by Teran et al. [2005b] and Saito et al. [2015]), which allows us to employ Projective Dynamics [Bouaziz et al. 2014] for improved robustness and performance. In addition, we contribute a novel formulation by which the active Hill-type muscle force can be accurately added to the Projective Dynamics framework, while retaining the robustness and efficiency of the method. We demonstrate how Jacobians of muscle forces, required for optimization-based motion controllers, can be analytically computed via a quasistatic assumption of the muscle deformation. The simulation framework is combined with a two-level trajectory optimization approach, motivated by the formulation of Lee et al. [2014] and adapted to the intricacies of our volumetric actuators.

2 RELATED WORK

Muscle-based anatomy modeling. Muscle-based anatomical simulation has been popular both in academic research and production [Damsgaard et al. 2006; Delp et al. 2007]. A thorough survey of muscle-oriented techniques in animation tasks has also been explored [Cruz Ruiz et al. 2017]. Over the past two decades, Computer Graphics research has pursued higher biomechanical accuracy in anatomy modeling, using Finite Element Method simulations [Teran et al. 2003] and models derived from medical imaging [Teran et al. 2005a]. The emphasis on visual plausibility instead of absolute biophysical accuracy has allowed Graphics research to be more aggressive in the pursuit of full-body soft tissue simulations, especially when volumetric simulation is involved, compared to the inclination of biomechanics literature towards localized parts of anatomy (e.g. thigh mechanics [Stelletta et al. 2017]). Simplified line-segment muscle models have allowed anatomical modeling to address inverse problems and motion control scenarios, on which volumetric simulation is often layered as a visual embellishment [Lee et al. 2009; Si et al. 2014].

In the domain of facial animation, volumetric descriptions of facial muscles is used to infer activations that move the face surface in accordance with a motion-captured target [Sifakis et al. 2005]. Ichim et al. [2017] optimize shape and contractile properties of facial tissues, starting from a template, as to reproduce input blendshape-based animations, yielding facial models that can be actuated into new animation sequences. the concept of blendshapes to material properties is extended, allowing simulated facial animations to reproduce motion effects attributable to temporally variable constitutive properties [Kozlov et al. 2017]. A coupled Eulerian-on-Lagrangian formulation is utilized in elastic tendon strands with a control strategy to simulate hand articulation [Sachdeva et al. 2015]. The Eulerian-On-Lagrangian framework has been applied to the human upper extremity [Fan et al. 2014], demonstrating the ability of the framework to handle muscle contact, and coupling to an articulated skeleton. Separate from issues related to simulation methods, the authoring of simulation-ready subject-specific models is a great challenge, even if a small number of geometric templates exist (which however do not provide strict specifications of material properties, and are not subject-specific). Skeletal geometry and kinematics are jointly inferred from a temporal deformation of only the outer skin surface [Zhu et al. 2015]. The way of generating a spectrum of human body types with various degrees of muscle growth is proposed, evolving from a standard template using physical processes [Saito et al. 2015]. Surface scan data from various subjects and body poses is also used to reconstruct personalized anatomical muscle models [Kadleček et al. 2016].

Simulation of nonlinear volumetric solids. Simulation performance, which has been a significant obstacle in early efforts, has been improved by virtue of stable implicit time integration schemes [Teran et al. 2005b], accelerated deformers based on regular lattices [Patterson et al. 2012], improved contact handling mechanisms [Mitchell et al. 2015a], and improved interactivity with Projective Dynamics schemes [Bouaziz et al. 2014]. Lattice-based deformers make simulator accommodate nontrivial topological features (e.g. surgical incisions), while safeguarding the regularity of the underlying data structures and the benefits of parallelism [Mitchell et al. 2015b]. Hierarchies of embedded discretizations are proposed to selectively infuse dynamic degrees of freedom where artists suggest additional

deformation detail is desired [Malgat et al. 2015]. Data-driven frameworks enable creating dynamic deformations of flesh and tissue on articulated characters [Kim et al. 2017; Pons-Moll et al. 2015]. A layered dynamic simulation of flesh is used, superimposed on a kinematic skeleton, to synthesize detailed skin deformation [Murai et al. 2017]. One-way or two-way coupled simulation of skeleton and flesh enables interactive, skeleton-driven animation of musculoskeleton [Capell et al. 2002; Liu et al. 2013]. Real-time flesh simulation is also available reducing its high dimensionality in rig space [Xu and Barbič 2016]. Multi-body simulation systems have been developed to deal with rigid bodies, soft bodies and their coupling in a uniform manner [Faure et al. 2012; Lloyd et al. 2012].

Motion control. Controllable musculoskeletal models have overwhelmingly relied on line-segment approximations of Hill-type models to actuate an articulated character skeleton. Taking the efficacy of such a system, human bipedal locomotion is demonstrated, producing optimized control patterns in a variety of gait patterns and styles [Lee et al. 2014]. Objectives involved in the crafting of realistic locomotion controllers include the optimization of effort [Lee et al. 2015; Wang et al. 2012]. Geijtenbeek et al. [2013] proceed to further optimize muscle geometry and routing (still within the realm of simplified actuators) to produce motions for a range of non-human bipedal creatures. Hand manipulation presents a unique set of challenges, especially when grasping constraints are needed for stability of the crafted activity [Liu 2008]. Sueda et al. [2008] introduced strand dynamics and sliding/surface constraints for hand simulation and control with emphasis on tendon dynamics. Balancing the head with neck muscles has been studied, coordinating their activations by feed forward and feed back rules [Lee and Terzopoulos 2006]. Muscle activation levels in tongue can be estimated by solving quadratic programming [Stavness et al. 2010]. Approaches based on deep learning have also been leveraged to craft control strategies in bipedal locomotion [Peng et al. 2017] and flapping flight [Won et al. 2017]. In addition to musculoskeletal systems, where the action of deformable simulation components chiefly manifests in skeletal articulation, control formulations have been applied to scenarios where the volumetric deformation itself is the primary output being optimized [Coros et al. 2012], or where locomotion is caused by deformation rather than skeletal articulation [Tan et al. 2012].

3 HUMAN MODEL

Our model focuses on the upper body musculoskeletal system. We use an articulated skeleton with 19 degrees of freedom, and 130 motor units (i.e. independently activated contractile regions within muscles), as illustrated in Figure 2(left). The skeleton is actuated as a result of muscle excitation, which is respectively governed by nonlinear internal dynamics. We simulate volumetric muscle volumes, discretized into individual tetrahedral meshes, with constitutive properties following a Hill-type model [Zajac 1989]. Sections 3.1 and 3.2 detail the volumetric muscle model discretization and simulation, Section 3.3 describes our modeling of the skeletal system, while coupling between the two is detailed in Section 3.4.

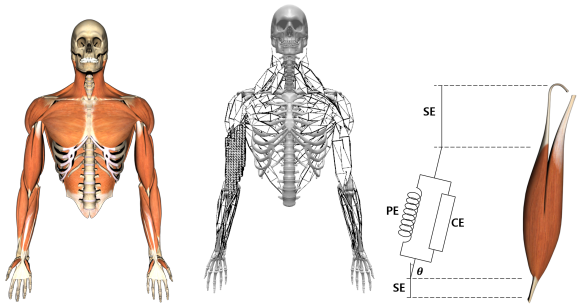


Fig. 2. (Left) The simulated muscles and skeletal components. (Middle) Low-resolution and high-resolution simulation meshes. (Right) Idealized Hill-type actuator.

3.1 FEM simulation

We assume tetrahedral meshes of muscles provided as input. Since we individually simulate each muscle volume (which might however contain several independently controllable contractile regions), consider any individual one of these meshes with k discrete vertices. The state of this muscle model is represented in the nodal positions $\mathbf{x}_n \in \mathbb{R}^{3k}$ and velocities $\mathbf{v}_n \in \mathbb{R}^{3k}$, where the subscript indicates a time instance t_n in the dynamic evolution of this model. Integrating the equations of motion in accordance with a Backward Euler scheme, provides the following update rule:

$$\begin{aligned} \mathbf{x}_{n+1} &= \mathbf{x}_n + h\mathbf{v}_{n+1} \\ \mathbf{v}_{n+1} &= \mathbf{v}_n + h\mathbf{M}^{-1}(\mathbf{f}_{\text{int}}(\mathbf{x}_{n+1}) + \mathbf{f}_{\text{ext}}) \end{aligned} \tag{1}$$

where h denotes the length of the time step, $\mathbf{M} \in \mathbb{R}^{3k\times 3k}$ is the mass matrix, and $\mathbf{f}_{\text{int}}(\mathbf{x}) = -\nabla_{\mathbf{x}} E(\mathbf{x})$ are internal forces computed from an elastic strain energy $E(\mathbf{x})$, and $\mathbf{f}_{\text{ext}}$ is the external force. We note that, in principle, the internal forces could also be dependent on the nodal velocities ($\mathbf{v}$) as well; this could be the case if explicit damping was incorporated into our methodology, or when material properties are allowed to vary as a function of velocity. We do not include explicit damping in our simulation, but there are velocity-dependent material properties stemming from the force-velocity relationship of Hill-type muscles as discussed in Section 3.2. In order to simplify our evolution, we use the velocity computed at the end of the previous time step ($\mathbf{v}_n$) whenever this appears in the expression of equation (1), essentially rendering it a constant for the purposes of the evolution from time t_n to t_{n+1}; hence, for simplicity we retain the notation $\mathbf{f}_{\text{int}}(\mathbf{x}_{n+1})$ for the forces in the update equation, indicating that only the positions $\mathbf{x}_{n+1}$ are to be solved for.

In general, the Backward Euler update is solved by using a Taylor expansion to approximate the nonlinear force as $\mathbf{f}(\mathbf{x}_{n+1}) \simeq \mathbf{f}(\mathbf{x}_n) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_n} (\mathbf{x}_{n+1} - \mathbf{x}_n)$. This approximation yields the following linear system, which we alternate between solving and updating the linearization, until convergence:

$$\left[\mathbf{M} - h^2 \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right] \mathbf{v}_{n+1} = \mathbf{M}\mathbf{v}_n + h(\mathbf{f}_{\text{int}}(\mathbf{x}_n) + \mathbf{f}_{\text{ext}}) \tag{2}$$

In order to reduce the computation cost and improve the robustness of the Newton approach, we adopt the Projective Dynamics formulation of [Bouaziz et al. 2014], who solve an equivalent problem by alternating an efficient and parallelizable *local* update rule, with a *global* purely quadratic problem, which can be accelerated by pre-factorizing its Hessian as a one-time cost. The Projective Dynamics formulation is centered on the premise that certain elastic strain energies can be written as the sum $E(\mathbf{x}) = \sum_i E_i(\mathbf{x})$, where each term $E_i(\mathbf{x})$ (conceptually associated with an individual *constraint*) has the form:

$$\mathbf{E}_i(\mathbf{x}) = \min_{\mathbf{p}_i \in C_i} \frac{k}{2} \|\mathbf{A}_i \mathbf{x} - \mathbf{p}_i\|^2 \tag{3}$$

Here, C_i is a manifold associated with each individual constraint, and $\mathbf{p}_i$ is the "projection" of the quantity $\mathbf{A}_i\mathbf{x}$ onto this manifold, which makes this energy match the expression of the original strain energy, once the solution has been reached. The definition of $\mathbf{A}_i$ and the process for computing the projection $\mathbf{p}_i$ is problem-dependent; for details we refer to the original work of [Bouaziz et al. 2014]. The local step amounts to updating all $\mathbf{p}_i$ values by projecting $\mathbf{A}_i\mathbf{x}$ onto each respective C_i, and in the global step we solve the pure quadratic system in equation (3) by assuming all $\mathbf{p}_i$ to be held constant to their computed projections. For our problem, this global step amounts to the solution of the equation:

$$(\mathbf{M} + h^2\mathbf{L})\mathbf{x}_{n+1} = \mathbf{M}(\mathbf{x}_n + h\mathbf{v}_n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}) + \mathbf{J}\mathbf{d} \tag{4}$$

where $\mathbf{L} = \sum_i \mathbf{A}_i^T\mathbf{A}_i$, $\mathbf{J} = \sum_i \mathbf{A}_i^T\mathbf{S}_i$, $\mathbf{d} = \sum_i \mathbf{S}_i^T\mathbf{p}_i$, and $\mathbf{S}_i$ are selector matrices, such that $\mathbf{p}_i = \mathbf{S}_i\mathbf{d}$.

Since the matrix of this system is constant we use Cholesky decomposition to factorize it as a pre-processing step. Furthermore we adopted a quasistatic assumption which is fast and robust, by taking the limit of this equation as $h \to \infty$. Although this simplification eschews dynamic effects (e.g., jiggling), we found it to have minimal impact on our controller, partially due to the fact that muscles are attached to the bones tightly, reducing the effect of the inertial motion on simulation. Under the quasistatic hypothesis, equation (4) simplifies to:

$$\mathbf{L}\mathbf{x}_{n+1} = \mathbf{f}_{\text{ext}} + \mathbf{J}\mathbf{d} \tag{5}$$

3.2 Muscle Model

In contraction dynamics of Hill-type muscles, the muscles are divided into three parts according to their role, as seen in Figure 2(right). Those three parts are: the Passive Element (PE) modeling the background elasticity of the muscle, the Contractile Element (CE) generating the force when the muscle excites, and the Serial Element (SE) modeling the tendon which transfers the muscle-generated force to the skeleton. We discuss how each component is accounted for in our simulation framework.

Passive Element. We use a simple Corotational Elastic energy for modeling the background isotropic elasticity of each muscle:

$$\Psi_{\text{PE}}(\mathbf{F}) = \frac{1}{2}\mu\|\mathbf{F} - \mathbf{R}\|^2 \tag{6}$$

where μ is the Young's modulus , $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$ is the deformation gradient for the element, and $\mathbf{R} = \mathbf{U}\mathbf{V}^T$. This is the exact constitutive model that the Projective Dynamics formulation of Bouaziz et al. [2014] is centered around. Equation (6) provides the energy for each tetrahedral element, and each such tetrahedron gives rise to one *constraint* C_i in the Projective Dynamics formulation, where $\mathbf{A}_i$ is the linear operator that maps nodal positions $\mathbf{x}$ to the deformation gradient $\mathbf{F}_i$ of the i-th element, and the projection operation maps $\mathbf{F}_i = \mathbf{U}\Sigma\mathbf{V}^T$ to its rotational component $\mathbf{p}_i = \mathbf{R} = \mathbf{U}\mathbf{V}^T$. Incorporating volume preservation into the formulation is mathematically involved. We refer to their paper [Bouaziz et al. 2014] for volume preservation and details on the relevant algebra.

Contractile Element. Physiologically, when the excitation signal is delivered to the muscle, actin and myosin fibers pull each other and contract the muscle. In accordance with the approach in prior Finite Element muscle modeling approaches [Lee et al. 2009; Teran et al. 2005a], we model the anisotropic action of the contractile element by an additive contribution to the strain energy $\Psi_m(l)$, which is taken to be dependent only on the *fiber stretch factor* $l = \|\mathbf{F}\mathbf{d}\|$, where $\mathbf{F}$ is the deformation gradient and $\mathbf{d}$ is a unit vector in the direction of the muscle fiber. An explicit expression for $\Psi_m(l)$ is typically never referenced or pursued, as only the gradient and Hessian of this energy will ever be used in an FEM simulation. The *derivative* of this quantity, however, is the fiber tension which is given by the well-known Hill-curve $\partial\Psi_m/\partial l := f_{hill}(l, \dot{l}, a)$ and this expression is the one actually used in simulation, where a is the level of muscle activation. An application of the chain rule provides the following formula for the Piola-Kirchhoff stress:

$$\mathbf{P}_m := \frac{\partial\Psi_m}{\partial\mathbf{F}} = \frac{\partial\Psi_m}{\partial l} \cdot \frac{\partial l}{\partial\mathbf{F}} = f_{hill}(l, \dot{l}, a) \cdot \frac{1}{l}\mathbf{F}\mathbf{d}\mathbf{d}^T \tag{7}$$

from which nodal forces can be readily computed (Sifakis and Barbic [2012] provide the relevant details for tetrahedral meshes).

It is well understood that the Projective Dynamics formulation of [Bouaziz et al. 2014] supports a specific and somewhat narrow scope of materials within its core formulation, due to the requirement that the energy being minimized must be expressible in the form of equation (3). This narrow scope motivated the later approach of Liu et al. [2017] that provided the opportunity to accommodate a broader gamut of materials, with some modest compromises in the robustness and efficiency of the prior formulation of Bouaziz et al. [2014] (e.g., the need to incorporate a line search for stability).

We introduce a novel approach to incorporate the Hill-type muscle force into the exact formulation mandated by the Projective Dynamics framework of Bouaziz et al. [2014]. We do so by defining the following energy associated with each muscle, in the exact fashion of equation (3):

$$\Psi_{\text{CE}}(\mathbf{F}) = \frac{1}{2}k\|\mathbf{F}\mathbf{d} - \mathbf{p}(l, \dot{l}, a)\|^2 \tag{8}$$

where k is a stiffness coefficient. In the local step of Projective Dynamics, the vector $\mathbf{p}(l, \dot{l}, a)$ is chosen in the subspace of vectors parallel to $\mathbf{F}\mathbf{d}$, with the specific scale factor in the expression below:

$$\mathbf{p}(l, \dot{l}, a) = \left[1 - \frac{f_{hill}(l, \dot{l}, a)}{k \cdot l}\right] \mathbf{F}\mathbf{d} \tag{9}$$

In the provided Appendix, we demonstrate that with the selection of this special value of $\mathbf{p}$, the Piola-Kirchhoff stress $\partial\Psi_{CE}/\partial\mathbf{F}$ computed by this expression matches exactly the value of $\mathbf{P}_m$ in equation (7)

corresponding to the standard active muscle stress of prior FEM approaches; thus, at equilibrium this Projective Dynamics formulation reproduces exactly the Hill-type muscle force that prior FEM muscle simulation formulations employed. With $\mathbf{p} = \mathbf{p}(l, \dot{l}, a)$ taken to be constant (in the global step), the expression $\Psi_{CE}(\mathbf{x})$ becomes a pure quadratic ($\mathbf{F}$ is a linear function of nodal positions $\mathbf{x}$, as mentioned in the treatment of the corotated energy), and the Projective Dynamics formulation becomes fully applicable. We ultimately compute nodal forces via the Piola-Kirchhoff stress in equation (7), and distribute them to the forces of each simulated tetrahedron as described by Sifakis and Barbic [2012]. Note that the choice of the coefficient k in equation (8) is arbitrary; however, for adequately large values of k, the provided expression for $\mathbf{p}(l, \dot{l}, a)$ most closely approximates a true Euclidean projection in the vector subspace spanned by $\mathbf{Fd}$. We found that a value of $k = 10^7$ (compare with $\mu = 5 \cdot 10^6$) generated robust convergence in our examples.

Serial Element. The Serial Element models the tendons on either side of the contractile segment of the muscle, and transmits the force to the bone insertion. Physiologically, the tendon is very stiff compared to the muscle and sustains very minimal elongation even under full muscle activation. Therefore, it was our design decision to model the tendon as a taut, inextensible wire that provides the boundary condition for the volumetric muscle simulation, by following the path of the conventional piecewise line-segment muscle primitive, from the insertion and through any way-points, until its reference length has been traversed. The endpoint of the tendon thus routed provides the Dirichlet boundary condition for our volumetric muscle simulation.

We also use this idealization of the tendon as an inextensible wire, which is routed through the muscle way-points to transmit the force generated by the volumetric muscle to the attached bones. We do so by rigidly transforming (i.e. rotation) the force along the way-points. Focusing on the way-points of origin of ith muscle, the axis of wire between $(j-1)$th and jth way-point can be defined as $\mathbf{u}_j = \frac{\mathbf{p}_{j-1} - \mathbf{p}_j}{\|\mathbf{p}_{j-1} - \mathbf{p}_j\|}$ (See Figure 3). Starting from the Dirichlet boundary, $j = 0, \cdots, s-1$ is an index of the way-points, s is the number of the way-points on the origin side and $\mathbf{p}_j$ is position of jth way-point. The force at the Dirichlet boundary $\tilde{\mathbf{f}}$ is transmitted to the end of wire, applying tension forces to the way-points:

$$\mathbf{f}_j^- = \|\tilde{\mathbf{f}}\|\mathbf{u}_j, \qquad \mathbf{f}_j^+ = -\mathbf{f}_{(j+1)}^-$$

where $\mathbf{f}_j^-$ is the tension force of jth way-point, and $\mathbf{f}_j^+$ is a reaction to the force $\mathbf{f}_{(j+1)}^-$, resulting in conservation of total momentum.

$$\mathbf{f}^{\text{origin}} = (\mathbf{f}_0^T, \mathbf{f}_1^T, \cdots, \mathbf{f}_{(s-1)}^T)^T$$

where $\mathbf{f}_j = \mathbf{f}_j^- + \mathbf{f}_j^+$. We apply the same procedure to the way-points on the insertion side and lump the forces into single vector $\mathbf{f}$.

A single muscle volume may contain multiple individually activated motor units. Our model captures this ability by attaching multiple serial elements to individual Dirichlet nodes of a muscle mesh, and modeling several distinct contractile elements within a single simulation volume. For example, the Biceps consist of two motor units (Figure 3), one originating from the long head, the other from the short head. Our upper body model incorporates a total of 130 motor units.

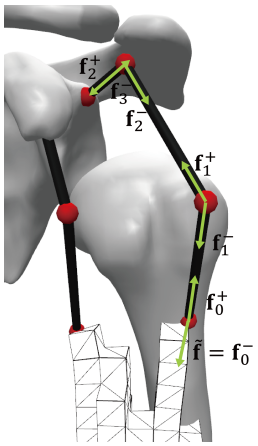


Fig. 3. Transmission of muscle force to bone. A rigid transformation is applied such that a vector in the direction of the central (contractile) line segment will be rotated parallel to the line segment adjacent to the insertion.

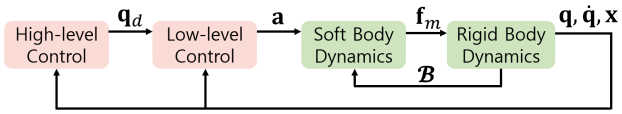


Fig. 4. The overview of our simulation and control framework: Rigid body states set Dirichlet boundaries for muscle force computation. Our hierarchical controller takes both rigid and soft body states as input and optimizes joint trajectories and muscle activation levels.

3.3 Skeleton model

The Euler-Lagrange equations for the dynamics using generalized coordinates can be represented as follows:

$$\mathbf{M}(\mathbf{q})\mathbf{q} + \mathbf{c}(\mathbf{q}, \mathbf{q}) = \mathbf{J}_m^T \mathbf{f}_m(\mathbf{a}) + \mathbf{J}_{\text{ext}}^T \mathbf{f}_{\text{ext}} \tag{10}$$

where $\mathbf{q}$ is the vector of joint angles, $\mathbf{M}(\mathbf{q})$ is the generalized mass matrix, $\mathbf{c}(\mathbf{q}, \mathbf{q})$ represents the Coriolis and gravitational forces, $\mathbf{f}_m = (\mathbf{f}_{(0)}^T, \mathbf{f}_{(1)}^T, \cdots, \mathbf{f}_{(r-1)}^T)^T$ are the muscle forces with the number of muscles r, $\mathbf{f}_{(i)}$ is the force acting on ith muscle and its way-points, $\mathbf{f}_{\text{ext}}$ is the external force, $\mathbf{J}_m$ and $\mathbf{J}_{\text{ext}}$ are Jacobians which map generalized coordinates to Cartesian coordinates, and $\mathbf{a} = (a_0, a_1, ..., a_{r-1})$ are the muscle activation levels, where a_i corresponds to the ith motor unit. Internally, $\mathbf{J}_m$ includes all the information of the muscle attachment points.

3.4 Coupling

Forward simulation for the musculoskeletal model proceeds by synchronous evolution of the FEM simulator and the rigid body simulator. At time t_n, for a given skeletal pose the boundary conditions for the volumetric muscle simulation are computed via the Serial Elements, as mentioned earlier in this section. The FEM simulator solves for the equilibrium shape of the muscle volume, in

accordance with equation (5). Muscle forces are transmitted from the Dirichlet nodes of simulated meshes to the bones via the Serial Element, and finally the dynamic state of the skeleton is advanced using the equations of motion, as summarized in Algorithm 1.

Algorithm 1: Forward Simulation

Data: $\mathbf{a}(t)$: muscle activation levels.
begin
  $\mathbf{x}_0$: initial positions of the soft body.
  $\mathbf{q}_0, \dot{\mathbf{q}}_0$: initial positions and velocities of rigid body.
  for $t = t_0, t_1, \cdots,$ **do**
    $\mathcal{B} \longleftarrow$ SetBoundaryConditions($\mathbf{q}_i$)
    $\tilde{\mathbf{f}} \longleftarrow$ SolveQuasiStatics($\mathcal{B}, \mathbf{x}_i, \mathbf{a}(t_i)$)
    $\mathbf{f}_m \longleftarrow$ TransferForces($\tilde{\mathbf{f}}$)
    $\mathbf{q}_{i+1}, \dot{\mathbf{q}}_{i+1} \longleftarrow$ ForwardDynamics($\mathbf{q}_i, \dot{\mathbf{q}}_i, \mathbf{f}_m, \mathbf{f}_{\text{ext}}$)
  end
end

4 CONTROL

We animate the musculoskeletal model by proposing a two-level hierarchical controller. The low-level controller tracks the desired motion on a per-frame basis and the high-level controller optimizes the motion given a specific task, such as juggling. Combining the robustness of the low-level controller and the generality of the high-level controller, our model can generate a diverse range of desired motion patterns under dynamic situations.

The main objective of the low-level controller is to find optimal activation levels for all muscles, tracking a given reference motion at each frame. To achieve this, we adapt the QP-based control method [Lee et al. 2009, 2014] to our model with volumetric muscles. We explain how the muscle Jacobians are computed by leveraging the quasistatic simulation assumption for muscle volumes, and how precomputed factors in the Projective Dynamics formulation can accelerate their computation. Using the low-level control, our high-level controller constructs the reference motion by solving an optimization problem at each frame. Our controller can flexibly generate desired motion patterns using parameterized curves without the need for any motion capture or key framing data.

4.1 Low-level controller

The goal of the low-level controller is to compute muscle activations for tracking the desired motion on a per-frame basis. A slight complication is that our musculoskeletal model is under-determined because the number of degrees of freedom for the skeleton is smaller than that for the muscle activations. Thus, there are many solutions for the same pose, and among them our system must choose one. As detailed below, we compute the solution minimizing three objectives (tracking, effort, and smoothness) through the optimization.

Tracking. To track the given motion $\mathbf{q}_d$, we compute the desired acceleration $\ddot{\mathbf{q}}_d$ based on PD (Proportional Derivative) control and penalize the difference between the actual joint acceleration $\ddot{\mathbf{q}}$ produced by muscle actuation.

$$\begin{aligned} E_{\text{tracking}} &= w_{\text{tracking}} \|\ddot{\mathbf{q}}_d - \ddot{\mathbf{q}}\|^2, \\ \ddot{\mathbf{q}}_d &= k_p(\mathbf{q}_d - \mathbf{q}) + k_v(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) \end{aligned} \tag{11}$$

where k_p, k_v are gains for PD control.

Effort. Humans move so as to minimize the required effort [Ralston 1976]. Thus, we introduce an objective function that penalizes effort by minimizing the required muscle activations.

$$E_{\text{effort}} = w_{\text{effort}} \|\mathbf{a}\|^2 \tag{12}$$

Smoothness. Physiologically, in activation dynamics, there are processes that convert neural signals to muscle activations [Zajac 1989]. This prevents sudden changes in activation levels. Thus, we penalize the variation of current muscle activations.

$$E_{\text{smooth}} = w_{\text{smooth}} \|\dot{\mathbf{a}}\|^2 \tag{13}$$

Using the three objectives described above, our low-level controller can be formulated as the following quadratic program:

$$\begin{aligned} \min_{\ddot{\mathbf{q}}, \mathbf{a}} \quad & E_{\text{tracking}} + E_{\text{effort}} + E_{\text{smooth}} \\ \text{subject to} \quad & \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{J}_m^T \mathbf{f}_m(\mathbf{a}) + \mathbf{J}_{\text{ext}}^T \mathbf{f}_{\text{ext}} \\ & 0 \le a_i \le 1 \quad \text{for} \quad i = 0, 1, \cdots r-1 \end{aligned} \tag{14}$$

where the equality constraints keep the equations of motion of the skeleton, and the inequality constraints enforce all muscle activations in the range of $[0, 1]$.

It is important to appreciate that equation (14) is *nonlinear* on the activations $\{a_i\}$, due to the nonlinearity of the muscle force $\mathbf{f}_m(\mathbf{a})$. The reason for this nonlinearity is subtle, and can be best elucidated by considering the line-segment idealization of the Hill-type primitive (e.g. [Lee et al. 2014]). In accordance with this model, *for a given muscle length*, the muscle force is an affine function of the activation. However, when the aggregate *length of the musculotendon* is held constant, e.g. for a given skeletal pose, activation of the contractile element will alter the lengths of the muscle and tendon individually, even if their sum is held constant. This variation of the muscle length infuses an additional nonlinearity in the muscle force as a consequence of activation, rendering the muscle force no longer an affine function of activation.

This nonlinearity is often consciously overlooked [Lee et al. 2014], by making an assumption that the muscle length variation is minimal; there would in fact be no error in this approximation if the tendon was infinitely stiff. With volumetric muscles, however, this approximation would be inexact even if the tendons were fully inextensible, as the volumetric contractile muscle has the potential to alter its geometry via non-uniform deformation that can certainly alter the muscle force produced, even if the longitudinal length of the muscle remained constant. An accurate linearization of the constraint equation (14) needs to employ a proper first-order Taylor exansion $\mathbf{f}_m(\mathbf{a}) \simeq \mathbf{f}_m(\mathbf{a}^*) + \frac{\partial \mathbf{f}_m}{\partial \mathbf{a}}(\mathbf{a} - \mathbf{a}^*)$, for which the Jacobian $\mathbf{J} = \partial \mathbf{f}_m / \partial \mathbf{a}$ needs to be evaluated.

4.2 Jacobian Computation

Notably, no closed-form expression of the force $\mathbf{f}_m(\mathbf{a})$ exists, since it incorporates the solution of nonlinear quasistatic equilibrium problem. It is, however, possible to compute its Jacobian *analytically* via careful differentiation of the quasistatic equilibrium condition. For any given value of the Dirichlet conditions applied at the endpoints of the muscle volume, denote by x the *free* nodes of the simulation mesh (excluding Dirichlet boundaries). Let us denote by $\mathbf{x}^*(\mathbf{a})$ the equilibrium positions of these nodes, under applied activations $\mathbf{a}$. Inserting these values in the expression of the total (passive and active) force $\mathbf{f}(\mathbf{x}, \mathbf{a})$ satisfies, by definition, the quasistatic equilibrium condition $\mathbf{f}(\mathbf{x}^*(\mathbf{a}), \mathbf{a}) = 0$. Differentiating with respect to $\mathbf{a}$ yields:

$$\frac{\partial}{\partial \mathbf{a}}\mathbf{f}(\mathbf{x}^*(\mathbf{a}), \mathbf{a}) = \frac{\partial \mathbf{f}}{\partial \mathbf{a}} + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}^*}{\partial \mathbf{a}} = \mathbf{0} \tag{15}$$

where all partial derivatives of forces are evaluated at $\mathbf{x} = \mathbf{x}^*(\mathbf{a})$. This allows us to compute the Jacobian of the muscle force under constant quasistatic equilibrium conditions, as follows:

$$\mathbf{J} = \frac{\partial \mathbf{f}_m(\mathbf{x}^*(\mathbf{a}), \mathbf{a})}{\partial \mathbf{a}} = \frac{\partial \mathbf{f}_m}{\partial \mathbf{a}} + \frac{\partial \mathbf{f}_m}{\partial \mathbf{x}}\frac{\partial \mathbf{x}^*}{\partial \mathbf{a}} \tag{16}$$

Closed form expressions of $\partial \mathbf{f}_m / \partial \mathbf{x}$ and $\partial \mathbf{f}_m / \partial \mathbf{a}$ are readily available, while the derivative of the quasistatic solution $\partial \mathbf{x}^* / \partial \mathbf{a}$ is computed by solving the linear system (15). Incidentally, the coefficient matrix of this system (i.e. the stiffness matrix $\partial \mathbf{f} / \partial \mathbf{x}$) has already been factorized for the needs of the Projective Dynamics simulation.

4.3 High-level Controller

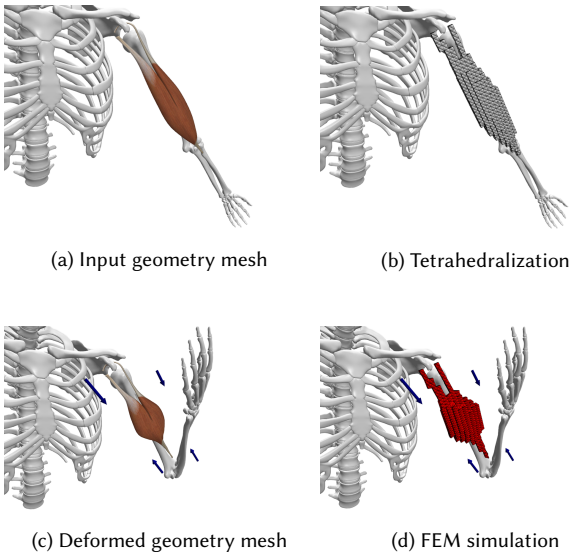
The high-level controller adjusts the reference motion of the upper body model in order to control it in a delicate fashion. We adopted the optimal control technique to solve this finite horizon problem

$$\begin{aligned} \min_{\mathbf{q}_d(t)} \quad & J(\mathbf{s}(t), \mathbf{q}_d(t)) \\ \text{subject to} \quad & \mathbf{q}_{\text{lower}} \le \mathbf{q}_d(t) \le \mathbf{q}_{\text{upper}} \\ & \dot{\mathbf{s}}(t) = \mathbf{g}(\mathbf{s}(t), \mathbf{q}_d(t), t) \quad \text{for} \quad 0 \le t \le t_f \\ & \text{given} \quad \mathbf{s}(0) = \mathbf{s}_0 \end{aligned} \tag{17}$$

where $\mathbf{q}_d(t)$ is the trajectory to be optimized, $\mathbf{s}(t)$ are the states of the system which contain positions and velocities of the rigid bodies, and the positions of all soft bodies, $\mathbf{q}_{\text{lower}}$ and $\mathbf{q}_{\text{upper}}$ are joint limits, $\mathbf{g}(\mathbf{s}(t), \mathbf{q}_d(t), t)$ governs the dynamics of the system, and $J(\mathbf{s}(t), \mathbf{q}_d(t))$ is an objective function which describes the high-level tasks. To convert this infinite-dimensional problem into a finite-dimensional optimization, we parametrize the joint trajectory $\mathbf{q}_d(t)$ with a Cubic Bézier spline as follows:

$$\mathbf{q}_d(t) = \sum_{i=0}^{3} \mathbf{B}_i(t)\mathbf{c}_i \tag{18}$$

where $\mathbf{B}_i(t)$ is the basis of Bézier spline, and $\mathbf{c}_i$ are the control points. By parameterizing $\mathbf{q}_d(t)$, the original problem is changed to the finite-dimensional problem of optimizing the control points $\mathbf{c}_i$. Once the optimal $\mathbf{c}_i$ values have been found, the reference trajectory $\mathbf{q}_d, \dot{\mathbf{q}}_d$ is readily computed through the basis functions of the spline. Specifically, in the juggling problem, our optimization problem is



(a) Input geometry mesh (b) Tetrahedralization

(c) Deformed geometry mesh (d) FEM simulation

Fig. 5. Muscle modeling and rendering

described as follows:

$$\begin{aligned} \min_{\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3} \quad & w_p \|\mathbf{p}_{\text{desired}} - \mathbf{p}_{\text{ball}}\|^2 + w_v \|\mathbf{v}_{\text{desired}} - \mathbf{v}_{\text{ball}}\|^2 \\ \text{subject to} \quad & \mathbf{q}_{\text{lower}} \le \mathbf{c}_i \le \mathbf{q}_{\text{upper}} \\ & \dot{\mathbf{s}}(t) = \mathbf{g}(\mathbf{s}(t), \mathbf{q}_d(t), t) \quad \text{for} \quad t_0 \le t \le t_f \\ & \text{given} \quad \mathbf{s}(t_0) = \mathbf{s}_0 \end{aligned} \tag{19}$$

Since the Bézier spline satisfies the convex hull property, just by applying bounds to the control points $\mathbf{c}_i$, we can produce a trajectory $\mathbf{q}_d(t)$ that is bounded above and below. The first term of the objective function penalizes the difference between the ball position and the desired position at the end of the swing phase. The second term penalizes the difference in velocity. A single evaluation of the energy requires simulations through the entire time interval $t_0 \le t \le t_f$; thus this is a performance-sensitive optimization operation.

5 EXPERIMENTS AND RESULTS

We implemented our muscle-driven control system in C++. The open source library DART was used for articulated body simulation [Lee et al. 2018]. Our upperbody musculoskeletal model includes 8 joints: two wrists, two elbows, two shoulders, two collar bones, and one torso. The wrist, the shoulder, and the torso are 3-DOF ball-and-socket joints and all the others are 1-DOF revolute. The model includes 72 muscles that affect the actuation of arm joints.

We used IPOPT [Wachter and Biegler 2006] to solve our per-frame optimization in equation (14). The control optimization and FEM simulation are updated at the rate of 200 Hz, while the articulated body dynamics is integrated at the rate of 1000 Hz. We used the gradient descent method for trajectory optimization with numerical differentiation of the objective function. Trajectory optimization requires 10 to 20 gradient descent iterations to converge and usually takes 10 to 15 minutes per iteration on an Intel i7-6700K 4.0GHz CPU.

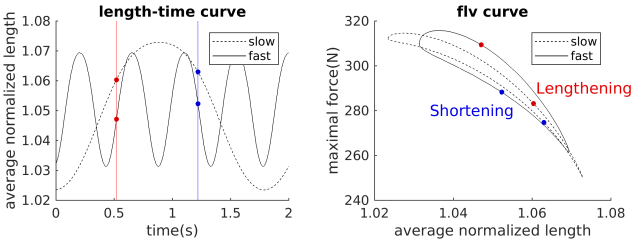


Fig. 6. Force-length-velocity curves. (Left) Maximal contractile force at the origin of the Biceps short head when it lengthens and shortens periodically at two different speeds. (Right) The Biceps generates larger force when it lengthens quickly.

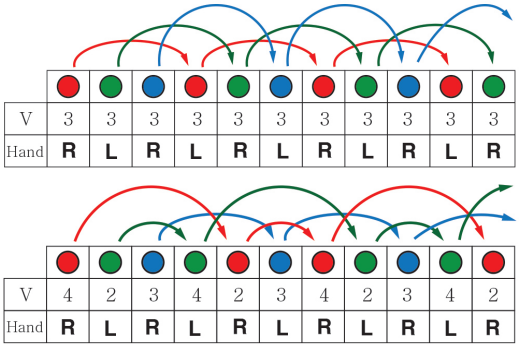


Fig. 7. Siteswap patterns of (up) 333 and (down) 423 juggling.

5.1 Muscle Geometry and Tetrahedralization

Starting with high-resolution geometric meshes of bones and muscles (Figure 5(a)), we annotated the origin and insertion of upper-body muscles in 3D geometry and tetrahedralized each individual muscle. The resolution of tetrahedralization is determined so that geometric features, such as bifurcating heads of the Biceps, are expressed clearly (Figure 5(b)). In our model, Biceps, Triceps, and Deltoids have about one thousand tetrahedra for each and all the others are simpler. The tetrahedral mesh undergoes deformation via FEM simulation (Figure 5(d)). We deform the original geometry mesh accordingly for the visualization of muscle contraction (Figure 5(c)). To transfer the deformation of the tetrahedral mesh to the geometric mesh, The 3D vertex location in the geometric mesh is expressed by the barycentric coordinates in its enclosing tetrahedra. Most of the vertices are enclosed by the tetrahedra and only small outliers fall outside the tetrahedral mesh. Each outlier vertex is mapped to the closest tetrahedron and its barycentric coordinates would have negative values.

Although high-resolution tetrahedral meshes are ideal for the accuracy of FEM simulation, low-resolution, hand-crafted meshes are also useful for the efficiency of simulating dexterous tasks. We used high-resolution meshes to examine the functionality and strength of each individual muscle under various conditions, and low-resolution meshes to simulate and control two-hand manipulation tasks.

The volumetric muscles driven by FEM simulation inherit the contraction dynamics of the Hill-type model. Each volumetric muscle generates its maximal contractile force when it is at its rest length and becomes weaker when it lengthens or shortens. The elasticity of the deformable material prevents it from lengthening excessively. The force-length-velocity curve in Figure 6 shows velocity-dependent contraction dynamics. The muscle generates its maximal force when it lengthens, which is called *eccentric contraction*.

5.2 Juggling

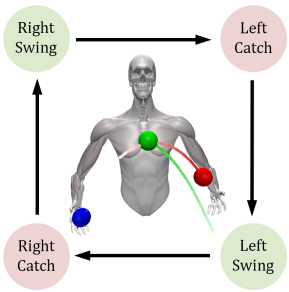
Juggling is a sophisticated performance with two hands for entertainment, art, or sports. A juggler usually manipulates more than three balls at the same time, while all of them are dynamic. The balls are repeatedly thrown by one hand, floating in the air, and finally captured by the other hand.

Mathematical expressions for juggling can be explained by the Siteswap value. Let T be the time per beat, D be the ratio of holding time that the ball spends in the hand per beat, and V be the Siteswap value which defines the pattern of juggling [Polster 2003]. For each beat, the ball should be thrown such that it lands on the other hand after V beats. For example, if the sequence of V is $3, 3, 3, \cdots$, the first ball thrown by the left hand will land on the right hand after 3 beats. While the first ball is still in the air, the right hand pitches the second ball after one beat. Figure 7 illustrates Siteswap patterns.

The maximum height of the ball is proportional to its Siteswap value V. Provided that T, D, and the sequence of V are given, we can determine the time of flight of individual balls, $t_{\text{flight}} = T(V - 2D)$. Note that we multiply D by two because one round trip of the ball takes two swing phases. If the motion of the ball is parabolic in the Y-axis, we can determine the initial velocity $\mathbf{v}_y = \frac{1}{2} g t_{\text{flight}}$, where $g = 9.8 m/s^2$ is the gravitational acceleration.

Our controller generates hand trajectories based on a finite state machine, where each state specifies either swing or catch tasks (Figure 8). For each beat, catch action moves the hand toward the landing position by solving inverse kinematics of the hand. Once the ball lands into the hand, a zero-DOF, welded joint is used to attach the ball to the hand. Swing (pitch) action requires trajectory optimization to match the desired position and velocity at the end of the swing phase.

Juggling Patterns. A *Cascade* is the simplest juggling pattern that pitches the balls to the same height. There are many juggling patterns other than Cascade patterns, varying the height of the balls, the symmetry/asymmetry of patterns, the shape of the projectiles, and the number of jugglers. We first demonstrate cascade patterns starting from 3 balls and adding balls one-by-one to end up with 5 balls (Figure 9(a)). With more balls, the juggler has to pitch them higher to maintain the cascade pattern. Our control system can seamlessly adapt to the addition/removal of balls and the switching between juggling patterns. Our muscle-driven control system can also simulate non-cascade juggling patterns. 423 juggling is a non-cascade pattern using three balls. The juggler pitches two balls higher than the third (Figure 9(b)). 64 juggling exhibits an asymmetric pattern with five balls. The right hand juggles with three balls,



k_p	500	w_{tracking}	1.0
k_v	44.7	w_{effort}	10.0
		w_{smooth}	10.0
w_p	0.01	T	0.3s
w_v	0.01	D	0.7
$t_f = T \cdot D$	0.21s	V	2to6

Fig. 8. A finite state machine for juggling and simulation parameters.

while the left hand independently juggles with the other two (Figure 9(c)). Our controller adaptively optimizes the swing trajectory to pitch the balls toward the desired direction at the desired speed, starting from the same initial configuration and parameter settings.

External Perturbation. We tested our controller under external pushes. Random forces of magnitude 300N and duration 0.5s are exerted on the torso, shoulder and elbow joints. Our solver re-optimizes the disturbed swing trajectory to adapt to the pushes. Using the original trajectory as initial guess, re-optimizing the trajectory takes only a few iterations to converge. The controller can endure large pushes on the torso and the shoulder, while even small perturbations at the extremities could be critical for dexterous manipulation such as juggling.

Mass Variations. This example shows the Cascade pattern with three balls of different mass (0.1kg, 0.5kg and 2kg). All three are thrown to the same height regardless of their mass difference and consequently their flight duration is the same. Having the flight time fixed, the response to different mass necessarily leads to the modulation of the swing duration. We regulate the release timing t_f of the ball in response to the loading mass.

$$t_f = c_1 m_{\text{ball}} + c_2 \tag{20}$$

where c_1 and c_2 are scalar coefficients and m_{ball} is the mass of the ball. The hand pulls back further with the heavy ball to absorb impact and travels longer to compensate for the mass. Our controller also accounts for the physiological property of the muscles, which can generate larger force when they are in eccentric contraction. Eccentric contraction of major agonistic muscles occurs when the swing arm pulls backward.

5.3 Muscle Disorder

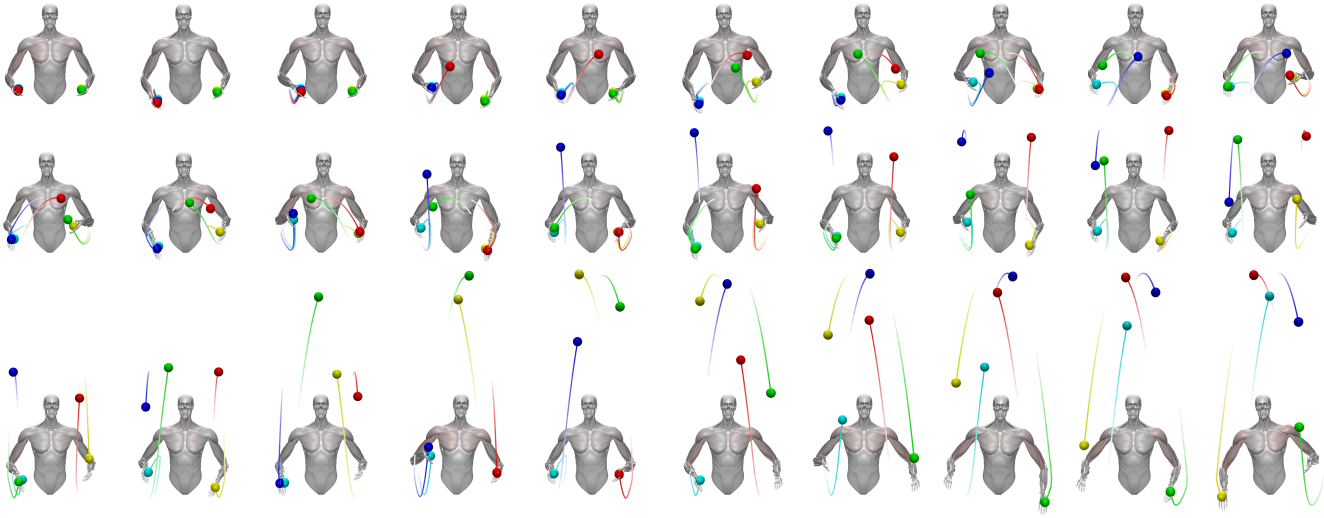
There are many types of muscle diseases with different causes and outcome. With muscle models simplified to line segments, we do not have many options to formulate the symptoms of muscular disorders into computational models. The most popular approach is to manipulate the force-length and force-velocity curves of the Hill-type model, which governs the muscle contraction dynamics. The Hill-type model is an analytic function based on in vitro measurement of muscle deformation and material properties. Therefore, manipulating the Hill-type model is an indirect approach based on approximations. The use of volumetric muscles opens up new possibilities in this regard, since the geometry and material properties of volumetric cells can be specified and manipulated directly.

Atrophy and Hypertrophy. Atrophy indicates the loss of mass and strength of muscles, which can cause disability or difficulty of actions. Conversely, *hypertrophy* is the increase of muscle volume and enhanced muscle strength. The symptoms of atrophy and hypertrophy can be simulated by changing the geometry of the tetrahedral mesh. As suggested by Kadlecek *et al.* [2016], we scaled the cross-section of the tetrahedral mesh by a factor of 0.5 (atrophy) and 1.5 (hypertrophy) to observe the weakening and strengthening of muscle capacity (Figure 10).

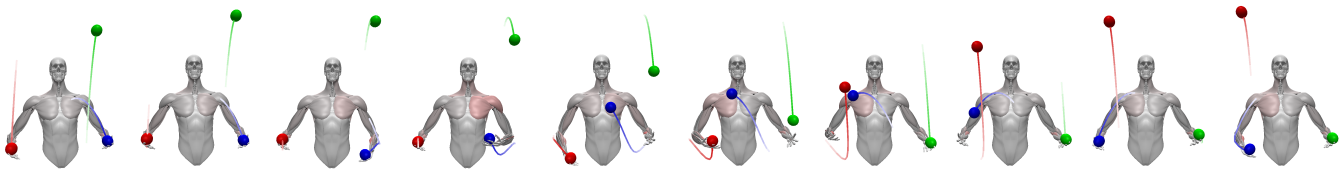
Deficiency and Paralysis. Since the material property of our volumetric muscles is derived from the Hill-type model, we can use the curve trick to simulate muscle deficiencies (Figure 11). The force-length curve indicates the strength of the muscle. In our simulation, scaling the magnitude of the force-length graph by a factor of 0.5, 0.2, and 0.05 resulted in the progressive weakness of the muscle. We can observe the increased activation of the nearby muscles which compensate for the weakness. The effect of such graph scaling always affects the entire muscle if the model is simplified to a line segment. Our volumetric model offers the flexibility to edit the material property of individual cells. Figure 12 demonstrates the effect of paralysis spreading progressively over the Biceps and Brachialis. The deactivated cells shown in dark brown do not generate any contractile force since the muscle-length curve is set to zero at all lengths.

Contracture. Contracture is the shortening or stiffening of muscles, that results in decreased movements and range of motion (Figure 13(middle)). The symptoms of contracture can be simulated by manipulating either the force-length curve or the volumetric mesh. With the Hill-type model, muscle shortening is described by its passive element that engages earlier than the normal force-length curve. Alternatively, our volumetric muscles offer a simpler, more intuitive approach. Shrinking the rest shape of the tetrahedral mesh results in increased tension between the origin and insertion of the contracted muscle.

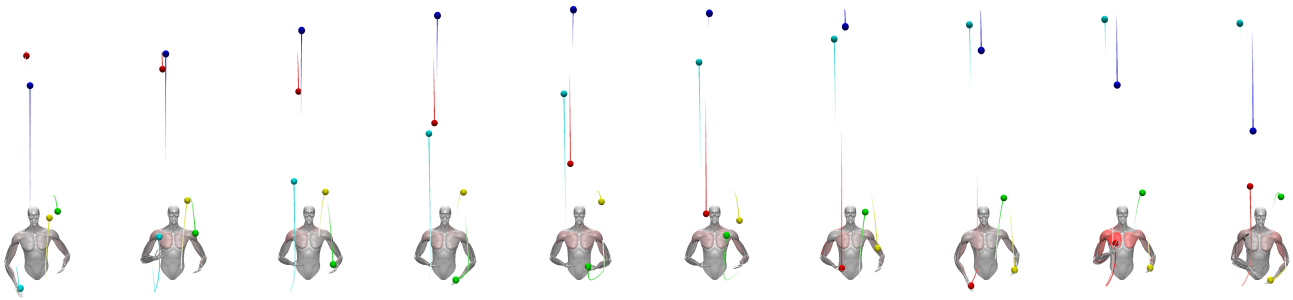
Orthopedic Surgery Simulation. Contracture occurs frequently in patients with cerebral palsy. The treatments include orthopedic surgery, which either lengthens musculotendons or reduces the tension by displacing the insertion of the muscle. A muscle that is a flexor at one joint may also be a flexor or extensor at another. These are called *two-joint muscles*, which span across two joints. Displacing the insertion across a joint by surgery can turn a two-joint muscle into one-joint. This type of surgery not only reduces muscle tension, but has the side-effect of changing the functionality of the muscle. Therefore, being able to evaluate the effects and side-effects of the surgery is of practical importance. We implemented a simple user interface system to simulate orthopedic surgery (Figure 13). The user interface shows an atlas of bone texture maps and allows the user to specify a new insertion point on the atlas. Our simulation system updates the muscle geometry instantly and reflects the update to the musculoskeletal simulation, visualizing the effects of the surgery.



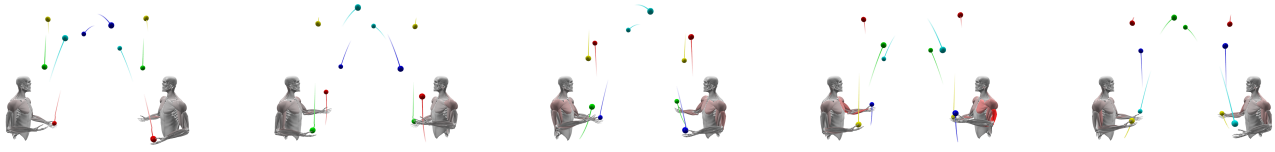
(a) 345 juggling



(b) 423 juggling



(c) 64 juggling



(d) Two person juggling

Fig. 9. Juggling patterns.

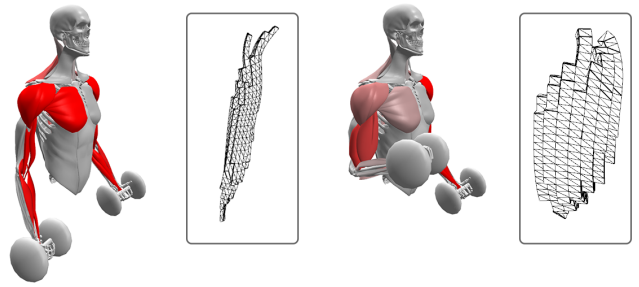


Fig. 10. Atrophy and Hypertropy of Biceps and Brachialis in the right arm. (Left) The crossectional area is scaled down by a factor of 0.5. (Right) The crossectional area is scaled up by a factor of 1.5. Muscle hypertrophy allows the weight (10 kg) to be lifted easily at low muscle activation levels.

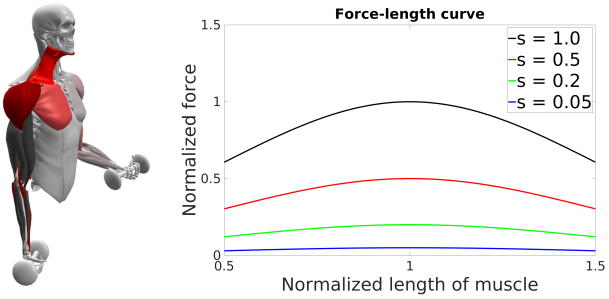


Fig. 11. Muscle weakness simulation. The weight of the dumbbell is 5kg. (Left) The muscles in the right arm shown in dark brown are weaker than normal ability. (Right) The scaling of the force-length curve determines the level of weakness.

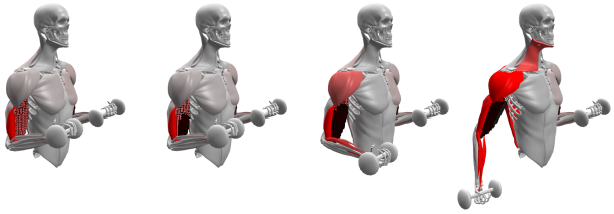


Fig. 12. Progressive paralysis of Biceps and Brachialis in the right arm. (From left to right) As dark cells spread, Biceps and Brachialis become weaker and therefore the nearby muscles in the forearm and the shoulder activate more to compensate for the weakness.

5.4 Limitations and Failure Cases

Even though successful applications of volumetric muscles have been demonstrated so far, we have also faced failure cases. First, the shoulder range of motion of our model is narrower than normal, making it difficult to raise the arm over the head. The range of joint motion is influenced by many factors including the geometric configuration of muscle origins, insertions, and their way-points, muscle strength, the discretization of simulation meshes, and the choice of a contraction dynamics model. Furthermore, parameters

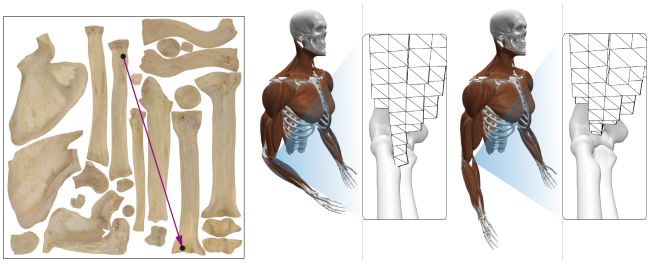


Fig. 13. Muscle contracture and orthopedic surgery simulation. (Left) A 2D point on the texture atlas maps to a 3D point on the bones. The user can easily specify the new insertion of the muscle on the texture atlas. (Middle) The increased tension of the contracted Biceps results in the flexed elbow at the relaxed arm. (Right) The surgery simulation displaces the insertion of the Biceps from the top of the *Radius* to the bottom of the *Humerus*. As a result, the Biceps becomes a one-joint muscle. Since the surgery increases the range of motion, the elbow becomes fully extended at the relaxed arm. The simulation also confirms the side-effect that the maximum torque at the elbow becomes weaker with the displaced muscle insertion.

of Hill-type curves affect the stability of dynamics simulation, as discussed by Sachdeva et al. [2015]. The negative slope of the force-length curve could incur numerical instability and consequently restrict the range of motion. We found it non-trivial to hand-tune parame.ters to achieve a desired range of motion. Optimization-based automatic parameter tuning is highly desirable in future study.

Secondly, the generated motion looks stiffer than we expected. The main cause of stiffness is large PD gains, which are necessary for accurate control. Juggling in particular requires precise aiming and timing of the balls. There is a trade-off between control accuracy and motion stiffness. Even though the total sum of muscle activation is minimized during trajectory optimization, the influence of large PD gains still remains to a certain extent. A potential remedy is the use of variable PD gains over the trajectory. Large gains are necessary only when it throws a ball with precision aiming at the end of arm swing. Small gains are preferred in the middle of arm swing for motion compliance. Variable PD gains optimized together with arm trajectory would alleviate stiffness without sacrificing accuracy.

6 DISCUSSION

Our attempt to transition the composition of an active simulated musculoskeletal system from the established practice of line-segment approximations to volumetric primitives represents a significant step towards the ultimate goal of a true biomemetic digital replica of the human anatomy and its complexity. Nevertheless, our current framework still consciously submits to a number of limitations, both in its scope of applicability, as well as biomechanical accuracy and computational capability. A number of these limitations stem from the computational cost of incorporating a system with a large number of dynamic degrees of freedom inside a full control loop; we expect that the continually increasing computational capabilities of current platforms provide an encouraging roadmap for addressing such performance-oriented challenges. Secondly, the degree of biomechanical accuracy that our system can afford is restricted by

the limited availability of highly detailed digital models of anatomy, not only in terms of geometrical shapes, but also in terms of the governing laws (material constitutive models, fiber fields of anisotropic contractile muscles, mechanical response models of fascia and viscoelastic tissues). We aspire that the foundation we have laid will allow us to leverage better models and governing laws, as those crystallize and become validated in relevant literature.

A number of technical considerations that complicate our pursuit of biomechanical accuracy result from the fact that the very nature of the line-segment simplification conceals certain challenges that are inherently present in real volumetric musculature. A line segment muscle primitive is constrained, by definition, to retain the shape of a straight line between any two successive via-points (or in its entirety, if the muscle primitive is not segmented). As a consequence, when the skeleton is articulated in a way that would cause the aggregate length of the musculotendon to shrink from its rest length, there is no ambiguity as to what the resulting shape of the primitive would be: it remains a piecewise linear curve. If a true volumetric muscle was modeled in complete isolation from its surrounding passive/connecting tissue and adjoining muscles (in direct analogy to how line-segment primitives are), there would be ambiguity in its resulting equilibrium shape, as there is a multitude of directions in which the geometry of the muscle could be laterally deflected, buckled or bent (in the case of tendons). This incurs a degree of ambiguity in the muscle forces that a compressed volumetric muscle produces as a result of skeleton-induced reduction of the length of the medial axis of the musculotendon (this is alleviated, in part, in scenarios where muscle activation incurs tension in the muscle). This behavior is stabilized, in reality, by the contact and collision between the muscle and its surrounding tissue, which helps resolve the resulting equilibrium shape. However, in our initial exploration presented in this work, we have not incorporated explicit contact and collision handling between muscle volumes and their surrounding tissues. As a consequence, our simulated muscles may experience bending or buckling modes under compression that are not fully representative of the biophysical behavior. From the standpoint of numerics, this also induces a practical limitation in the minimum thickness we can allow the tendon regions to assume, as excessively thin tendons would both increase this modeling error, and aggrevate the presence of inaccurate buckling modes. We expect this deficiency to be cured in future work by the incorporation of careful contact and collision processing between a fully-coupled set of muscles and passive/connective tissue.

In the real human body, individual muscle volumes are mechanically correlated via their contact coupling. For example, it is possible for the insertions of a given muscle to exert tension to the skeleton even if the muscle is fully inactive, at a kinematic state where the line-segment approximation would yield no tension at all; this would be possible if the contraction of a neighboring muscle, coupled via connective tissue and contact handling, causes a volumetric deformation in the inactive muscle to be deformed as a side-effect, producing tension (and forces at the insertions) that would not be possible with fully-independent line-segment muscle primitives. We must highlight that our current formulation only partially captures this real-world behavior, on volumetric muscle primitives with multiple independently-activated contractile regions modeled in the same volumetric simulation mesh; on the other hand, we do not currently capture this coupling by means of contact processing between distinct muscle volumes, which are simulated independently. Our mathematical formulation for computing force Jacobians is able to capture this dependency on muscles with several contractile regions. Extending this capability (and the Jacobian computation) to contact-coupled muscles will be more challenging, and possibly restrict our options for contact handling (e.g. creating a preference for coupling via "penalty" contact forces, instead of impulse-based corrections mediating momentum exchange [Bridson et al. 2003]).

Finally, the simplicity of the line-segment muscle primitives reduces the complexity of modeling individual muscles to a specification of the muscle path, the muscle/tendon ratio (without any need for localization), and parameters governing its maximum force-generating potential. Volumetric muscle primitives have much broader modeling flexibility, and as a consequence many more opportunities for less-than-accurate geometry or material parameter specifications to create deviations from the ground truth. In a sense, the increased flexibility of volumetric models to approach reality comes at the cost of increased opportunities for modeling errors to create deviations from it. As an example, the cross-sectional geometry of tendons can have a significant effect on the resulting geometry of the muscle volume (if the Young's modulus is set to a near-constant value, as biophysically expected). Line-segment models can mask such parameter tuning choices behind an individualized setting of the tendon stiffness on a muscle-by-muscle basis, making the parameter space easier to tune (even if such parameters have a very loose connection to the underlying first principles). Our existing model consciously focuses on upper body motion; future work should explore extensions to a full-body human model, capable of resolving more diverse motion.

REFERENCES

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective dynamics: fusing constraint projections for fast simulation. *ACM Transactions on Graphics* 33, 4, Article 154 (2014).

Robert Bridson, Sebastian. Marino, and Ron Fedkiw. 2003. Simulation of Clothing with Folds and Wrinkles. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 28–36.

Steve Capell, Seth Green, Brian Curless, Tom Duchamp, and Zoran Popović. 2002. Interactive Skeleton-driven Dynamic Deformations. *ACM Transactions on Graphics* 21, 3 (2002), 586–593.

Stelian Coros, Sebastian Martin, Bernhard Thomaszewski, Christian Schumacher, Robert Sumner, and Markus Gross. 2012. Deformable objects alive! *ACM Transactions on Graphics* 31, 4, Article 69 (2012).

Ana Lucia Cruz Ruiz, Charles Pontonnier, Nicolas Pronost, and Georges Dumont. 2017. Muscle-Based Control for Character Animation. *Computer Graphics Forum* 36, 6 (2017), 122–147.

Michael Damsgaard, John Rasmussen, SÃÿren T. Christensen, Egidijus Surma, and Mark D. Zee. 2006. Analysis of musculoskeletal systems in the AnyBody Modeling System. *Simulation Modelling Practice and Theory* 14, 8 (2006), 1100 – 1111.

Scott L. Delp, Frank C. Anderson, Anderson S. Arnold, Peter Loan, Ayman Habib, T. John, Eran Guendelman, and Darryl G. Thelen. 2007. OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement. *IEEE Transactions on Biomedical Engineering* 54, 11 (2007), 1940–1950.

Ye Fan, Joshua Litven, and Dinesh K Pai. 2014. Active volumetric musculoskeletal systems. *ACM Transactions on Graphics* 33, 4, Article 152 (2014).

François Faure, Christian Duriez, Hervé Delingette, Jérémie Allard, Benjamin Gilles, Stéphanie Marchesseau, Hugo Talbot, Hadrien Courtecuisse, Guillaume Bousquet, Igor Peterlik, and Stéphane Cotin. 2012. SOFA: A Multi-Model Framework for Interactive Physical Simulation. In *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*. Studies in Mechanobiology, Tissue Engineering and Biomaterials, Vol. 11. 283–321.

Thomas Geijtenbeek, Michiel van de Panne, and A. Frank van der Stappen. 2013. Flexible Muscle-based Locomotion for Bipedal Creatures. *ACM Transactions on Graphics* 32, 6, Article 206 (2013).
Alexandru-Eugen Ichim, Petr Kadleček, Ladislav Kavan, and Mark Pauly. 2017. Phace: physics-based face modeling and animation. *ACM Transactions on Graphics* 36, 4, Article 153 (2017).
Petr Kadleček, Alexandru-Eugen Ichim, Tiantian Liu, Jaroslav Křivánek, and Ladislav Kavan. 2016. Reconstructing personalized anatomical models for physics-based body animation. *ACM Transactions on Graphics* 35, 6, Article 213 (2016).
Meekyoung Kim, Gerard Pons-Moll, Sergi Pujades, Seungbae Bang, Jinwook Kim, Michael J. Black, and Sung-Hee Lee. 2017. Data-driven Physics for Human Soft Tissue Animation. *ACM Transactions on Graphics* 36, 4, Article 54 (2017), 54:1–54:12 pages.
Yeara Kozlov, Derek Bradley, Moritz Bächer, Bernhard Thomaszewski, Thabo Beeler, and Markus Gross. 2017. Enriching Facial Blendshape Rigs with Physical Simulation. *Computer Graphics Forum* 36, 2 (2017), 75–84.
Jeongseok Lee, Michael X. Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S. Srinivasa, Mike Stilman, and C. Karen Liu. 2018. DART: Dynamic Animation and Robotics Toolkit. *The Journal of Open Source Software* 3, 22 (2018), 500.
Sunghee Lee, Eftychios Sifakis, and Demetri Terzopoulos. 2009. Comprehensive biomechanical modeling and simulation of the upper body. *ACM Transactions on Graphics* 28, 4, Article 99 (2009).
Sung Hee Lee and Demetri Terzopoulos. 2006. Heads Up!: Biomechanical Modeling and Neuromuscular Control of the Neck. *ACM Transactions on Graphics* 25, 3 (2006), 1188–1198.
Yoonsang Lee, Kyungho Lee, Soon-Sun Kwon, Jiwon Jeong, Carol O'Sullivan, Moon Seok Park, and Jehee Lee. 2015. Push-Recovery Stability of Biped Locomotion. *ACM Transactions on Graphics* 34, 6 (2015).
Yoonsang Lee, Moon Seok Park, Taesoo Kwon, and Jehee Lee. 2014. Locomotion control for many-muscle humanoids. *ACM Transactions on Graphics* 33, 6, Article 218 (2014).
C. Karen Liu. 2008. Synthesis of Interactive Hand Manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 163–171.
Libin Liu, KangKang Yin, Bin Wang, and Baining Guo. 2013. Simulation and Control of Skeleton-driven Soft Body Characters. *ACM Transactions on Graphics* 32, 6, Article 215 (2013), 8 pages.
Tiantian Liu, Sofien Bouaziz, and Ladislav Kavan. 2017. Quasi-Newton Methods for Real-Time Simulation of Hyperelastic Materials. *ACM Transactions on Graphics* 36, 3, Article 23 (2017).
John E. Lloyd, Ian Stavness, and Sidney Fels. 2012. ARTISYNTH: a fast interactive biomechanical modeling toolkit combining multibody and finite element simulation. (2012).
Richard Malgat, Benjamin Gilles, David IW Levin, Matthieu Nesme, and François Faure. 2015. Multifarious hierarchies of mechanical models for artist assigned levels-of-detail. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 27–36.
Nathan Mitchell, Mridul Aanjaneya, Rajsekhar Setaluri, and Eftychios Sifakis. 2015a. Non-manifold level sets: A multivalued implicit surface representation with applications to self-collision processing. *ACM Transactions on Graphics* 34, 6, Article 247 (2015).
Nathan Mitchell, Eftychios Sifakis, et al. 2015b. GRIDiron: An interactive authoring and cognitive training foundation for reconstructive plastic surgery procedures. *ACM Transactions on Graphics* 34, 4, Article 43 (2015).
Akihiko Murai, Q Youn Hong, Katsu Yamane, and Jessica K Hodgins. 2017. Dynamic skin deformation simulation using musculoskeletal model and soft tissue dynamics. *Computational Visual Media* 3, 1 (2017), 49–60.
Taylor Patterson, Nathan Mitchell, and Eftychios Sifakis. 2012. Simulation of complex nonlinear elastic bodies using lattice deformers. *ACM Transactions on Graphics* 31, 6, Article 197 (2012).
Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. 2017. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics* 36, 4, Article 41 (2017).
Burkard Polster. 2003. *The Mathematics of Juggling*.
Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J Black. 2015. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics* 34, 4, Article 120 (2015).
HJ Ralston. 1976. Energetics of human walking. In *Neural control of locomotion*. 77–98.
Prashant Sachdeva, Shinjiro Sueda, Susanne Bradley, Mikhail Fain, and Dinesh K Pai. 2015. Biomechanical simulation and control of hands and tendinous systems. *ACM Transactions on Graphics* 34, 4, Article 42 (2015).
Shunsuke Saito, Zi-Ye Zhou, and Ladislav Kavan. 2015. Computational bodybuilding: Anatomically-based modeling of human bodies. *ACM Transactions on Graphics* 34, 4, Article 41 (2015).
Weiguang Si, Sung-Hee Lee, Eftychios Sifakis, and Demetri Terzopoulos. 2014. Realistic biomechanical simulation and control of human swimming. *ACM Transactions on Graphics* 34, 1, Article 10 (2014).
Eftychios Sifakis and Jernej Barbic. 2012. FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses*. 20.
Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. 2005. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Transactions on Graphics* 24, 3 (2005).
Ian Stavness, Alan G Hannam, John E Lloyd, and Sidney Fels. 2010. Predicting muscle patterns for hemimandibulectomy models. (2010), 483–91.
Julien Stelletta, Raphaël Dumas, and Yoann Lafon. 2017. Modeling of the thigh: a 3D deformable approach considering muscle interactions. In *Biomechanics of Living Organs*. 497–521.
Shinjiro Sueda, Andrew Kaufman, and Dinesh K. Pai. 2008. Musculotendon Simulation for Hand Animation. *ACM Transactions on Graphics* 27, 3, Article 83 (2008), 8 pages.
Jie Tan, Greg Turk, and C Karen Liu. 2012. Soft body locomotion. *ACM Transactions on Graphics* 31, 4, Article 26 (2012).
Joseph Teran, Sylvia S Blemker, Victor Ng-Thow-Hing, and Ronald Fedkiw. 2003. Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*. 68–74.
Joseph Teran, Eftychios Sifakis, Silvia S Blemker, Victor Ng-Thow-Hing, Cynthia Lau, and Ronald Fedkiw. 2005a. Creating and simulating skeletal muscle from the visible human data set. *IEEE Transactions on Visualization and Computer Graphics* 11, 3 (2005), 317–328.
Joseph Teran, Eftychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. 2005b. Robust quasistatic finite elements and flesh simulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*. 181–190.
Darryl G Thelen et al. 2003. Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults. *Transactions-American Society Of Mechanical Engineers Journal Of Biomechanical Engineering* 125, 1 (2003), 70–77.
Andreas Wachter and Lorenz T. Biegler. 2006. On the Implementation of an Interior-point Filter Line-search Algorithm for Large-scale Nonlinear Programming. *Math. Program.* 106, 1 (2006), 25–57.
Jack M. Wang, Samuel R. Hamner, Scott L. Delp, and Vladlen Koltun. 2012. Optimizing Locomotion Controllers Using Biologically-based Actuators and Objectives. *ACM Transactions on Graphics* 31, 4, Article 25 (2012).
Jungdam Won, Jongho Park, Kwanyu Kim, and Jehee Lee. 2017. How to train your dragon: example-guided control of flapping flight. *ACM Transactions on Graphics* 36, 6, Article 198 (2017).
Hongyi Xu and Jernej Barbič. 2016. Pose-space Subspace Dynamics. *ACM Transactions on Graphics* 35, 4, Article 35 (2016), 14 pages.
Felix E Zajac. 1989. Muscle and tendon Properties models scaling and application to biomechanics and motor. *Critical reviews in biomedical engineering* 17, 4 (1989), 359–411.
Lifeng Zhu, Xiaoyan Hu, and Ladislav Kavan. 2015. Adaptable anatomical models for realistic bone motion reconstruction. *Computer Graphics Forum* 34, 2 (2015), 459–471.

ACKNOWLEDGMENTS

This work was supported by *Samsung Research Funding Center* under Project Number SRFC-IT1801-01, and *Eftychios Sifakis* was supported in part by National Science Foundation grants IIS-1253598, CCF-1533885.

APPENDIX: CONTRACTILE ELEMENT TREATMENT

We seek an expression for $\mathbf{p}(l, \dot{l}, a)$ in equation (8) which allows the Piola stress $\mathbf{P}_{CE} = \partial\Psi_{CE}/\partial\mathbf{F}$ to match the expression of $\mathbf{P}_m$ in equation (7) associated with the conventional active stress induced by Hill-type muscles. The differentiation is written as follows:

$$\begin{aligned}\frac{\partial\Psi_{CE}}{\partial\mathbf{F}} &= \frac{\partial}{\partial\mathbf{F}}\left\{\frac{k}{2}(\mathbf{Fd}-\mathbf{p})^T(\mathbf{Fd}-\mathbf{p})\right\} \\ &= k(\mathbf{Fdd}^T - \mathbf{pd}^T) \\ &= k(1-\gamma)\mathbf{Fdd}^\mathbf{T} \qquad (21)\end{aligned}$$

where equation (21) incorporates the assumption that $\mathbf{p} = \gamma\mathbf{Fd}$ is taken among the multiples of the deformed fiber direction $\mathbf{Fd}$. The expression for $\mathbf{p}$ in equation (9) results from equating the right hand sides of equations (21) and (7).