Simulation of Hand Anatomy Using Medical Imaging

MIANLUN ZHENG*, University of Southern California, USA
BOHAN WANG*, University of Southern California, Massachusetts Institute of Technology, USA
JINGTAO HUANG, University of Southern California, USA
JERNEJ BARBIČ, University of Southern California, USA

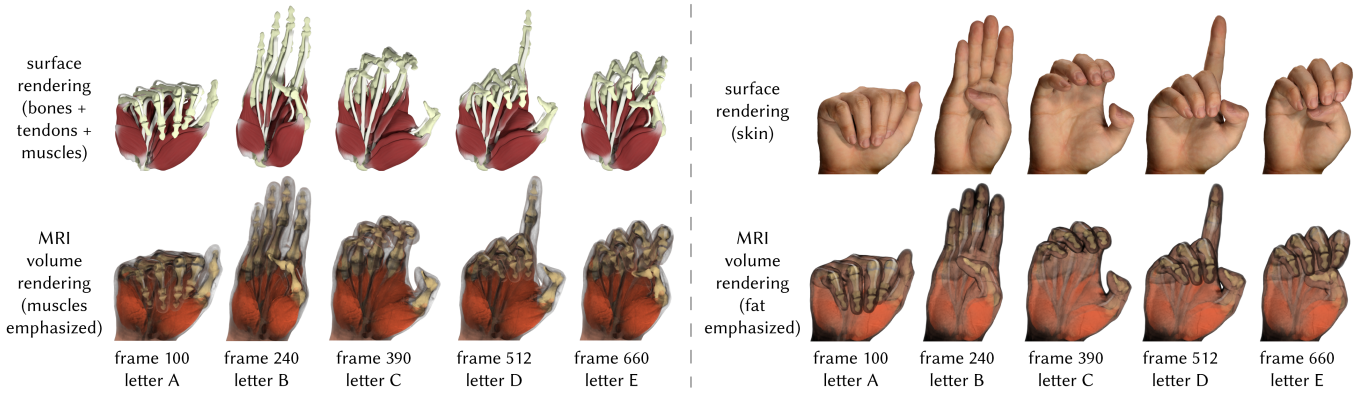


Fig. 1. **Human hand anatomy animated across the hand's range of motion:** These surface and volumetric renders display complex hand organ volumetric motion computed using our method. The hand was posed into letters A-E of the American Sign Language; these poses were not scanned by MRI, but are computer-simulated.

Precision modeling of the hand internal musculoskeletal anatomy has been largely limited to individual poses, and has not been connected into continuous volumetric motion of the hand anatomy actuating across the hand's entire range of motion. This is for a good reason, as hand anatomy and its motion are extremely complex and cannot be predicted merely from the anatomy in a single pose. We give a method to simulate the volumetric shape of hand's musculoskeletal organs to any pose in the hand's range of motion, producing external hand shapes and internal organ shapes that match ground truth optical scans and medical images (MRI) in multiple scanned poses. We achieve this by combining MRI images in multiple hand poses with FEM multibody nonlinear elastoplastic simulation. Our system models bones, muscles, tendons, joint ligaments and fat as separate volumetric organs that mechanically interact through contact and attachments, and whose shape matches medical images (MRI) in the MRI-scanned hand poses. The match to MRI is achieved by incorporating pose-space deformation and plastic strains into the simulation. We show how to do this in a non-intrusive manner that still retains all the simulation benefits, namely the ability to prescribe realistic material properties, generalize to arbitrary poses, preserve volume and obey contacts and attachments. We use our method to produce volumetric renders of the internal anatomy of the human hand in motion, and to compute and render highly realistic hand surface shapes. We evaluate our method by comparing it to optical scans, and demonstrate that we qualitatively and quantitatively substantially decrease the error compared to previous work. We test our method on five complex hand sequences, generated either using keyframe animation or performance animation using modern hand tracking techniques.

CCS Concepts: • **Computing methodologies** → **Physical simulation**.

Additional Key Words and Phrases: hand, MRI, animation, simulation, modeling, anatomy, deformable objects, FEM, optical scanning

ACM Reference Format:
Mianlun Zheng, Bohan Wang, Jingtao Huang, and Jernej Barbič. 2022. Simulation of Hand Anatomy Using Medical Imaging. *ACM Trans. Graph.* 41, 6, Article 273 (December 2022), 20 pages. https://doi.org/10.1145/3550454.3555486

1 INTRODUCTION

Modeling and animation of hands has numerous applications in film, computer games, virtual reality, CAD/CAM, medicine and related fields. Precise modeling of the motion of the internal anatomy of the human hand can help with designing hand prosthetics, better surgical tools, or improve ergonomics of tools and apparel. Hand anatomical models can also be used to generate high-quality datasets of hand shapes across the range of motion, which deep learning can use to track hand poses and recognize hand activity. The importance of hand modeling and animation has long been recognized, and there is a large body of work on the topic. Most existing methods, however, focus on the external hand shape and subsume internal anatomy with simplified non-anatomical representations. We give a method to accurately model the motion of the internal musculoskeletal anatomy of the human hand across the entire range of motion of the

*Equal contribution

Authors' addresses: Mianlun Zheng, University of Southern California, Los Angeles, USA, mianlunz@usc.edu; Bohan Wang, University of Southern California, Massachusetts Institute of Technology, Los Angeles, USA, bohanwan@usc.edu; Jingtao Huang, University of Southern California, Los Angeles, USA, jingtaoh@usc.edu; Jernej Barbič, University of Southern California, Los Angeles, USA, jnb@usc.edu.

0730-0301/2022/12-ART273
https://doi.org/10.1145/3550454.3555486

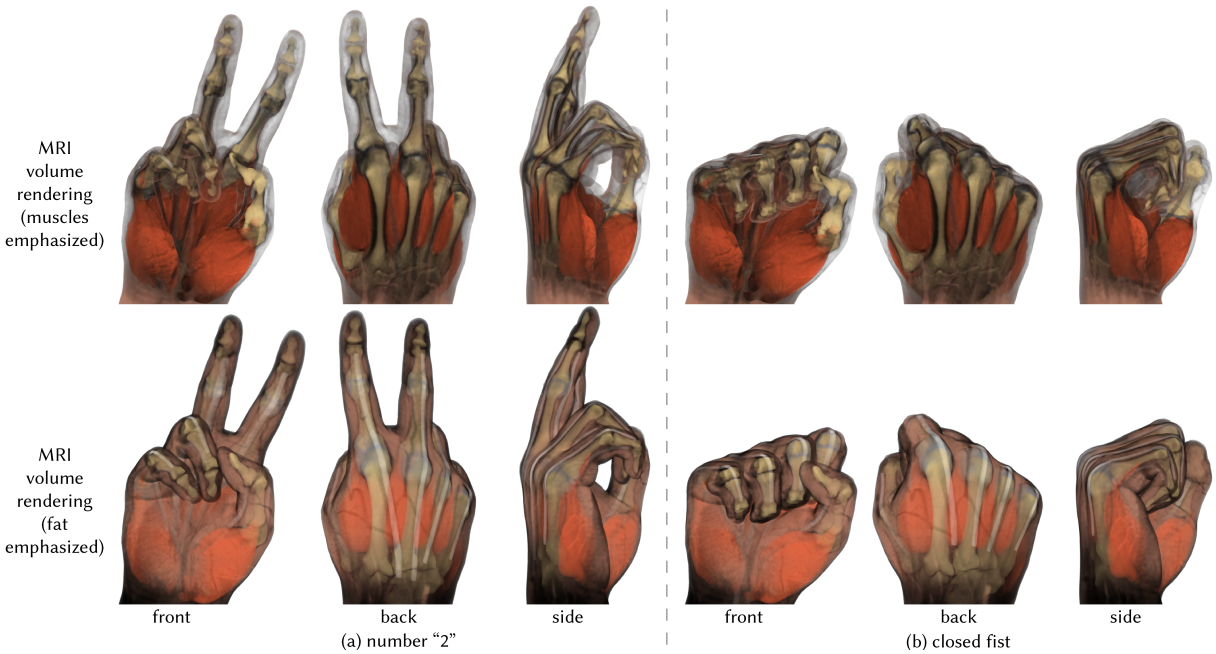


Fig. 2. **Representative frames for motion "Numbers 1-5" and "Close the fist".** We display the front, back and the side of the hand. Our method can animate the hand internal musculoskeletal organs to arbitrary poses in the hand's range of motion, in a manner that matches the medical images in the example poses (Figure 32) and even non-example poses (Figure 33). In addition to musculoskeletal organs, arteries and veins are also clearly visible in the "fat emphasized" MRI sequence.

human hand, and do so in a manner that demonstratedly matches medical imaging (MRI) across multiple hand poses. We model hand bones, tendons, ligaments, muscles, fat, all as separate volumetric three-dimensional tissues that move and deform like observed in the MRI scans of a real person's hand. Our method is a hybrid data-driven simulation method that uses data (MRI scans) to steer a Finite Element Method volumetric multibody simulation to be able to "extrapolate" from the MRI scans to the entire range of motion of the hand. The steering is achieved in different novel ways for the different tissues; for example, tendons are steered through novel sliding constraints defined using pose-space deformation [Lewis et al. 2000] against the MRI data (Section 5), and muscles and fat are steered with novel per-organ and per-pose plastic strains, optimized so that the FEM simulation matches the MRI data in the scanned MRI poses (Sections 4, 7). We limit our muscle modeling to concentric isotonic contraction, i.e., relaxed poses free of (substantial) muscle firing such as those encountered during free hand motion or gentle grasping. This is because the poses in our MRI scans are also relaxed. We are not aware of any method that has simultaneously captured hand MRI data and muscle activation patterns in multiple poses, nor do we attempt to do so in this work. This is still sufficient for free-space hand motion as commonly encountered in many applications, and analysis and simulation of its internal anatomical motion.

Previous volumetric methods have attempted to model the entire hand as a single soft tissue [Kry et al. 2002; Wang et al. 2019], but they ignored the fact that different hand organs have very different mechanical properties. For example, muscles are much stiffer than fat; and are active tissues. A hand tendon behaves like an inextensible rod. In addition, adjacent tissues usually move relative to each other, such as muscles sliding against each other. These effects cannot be captured by merely modeling the entire hand as a single soft tissue. For accurate hand modeling, it is important to mesh and simulate different organs separately. While different hand organs have been widely studied in many research fields independently, to the best of our knowledge, nobody has attempted to model the entire hand as a complete biomechanical volumetric system. Existing volumetric simulation models are not designed for matching medical images such as MRI, whereas our method matches them both qualitatively and quantitatively.

As commonly done in VFX industry, our simulation proceeds in layers [Tissue 2013]. The input to our work is a hand joint hierarchy animation generated using any suitable means, e.g., hand motion capture, or vision-based hand tracking systems. The bones (rigid objects) are then driven kinematically using the joint hierarchy, followed by bone fascia (cloth), and then soft-tissue simulation layers: tendons (rods), ligaments, muscles (elastic solids) and muscle fascia (cloth). Our last layer is the fat tissue (elastic solid), which gives us the external shape of the hand. We use the "cloth" terminology because it is common in computer animation; "cloth" here simply means an elastic thin-shell. We employ a layered simulation, as opposed to a coupled simulation of all hand organs, for two reasons: (1) even layered simulation is at the limit of what we can achieve computationally today with state of the art algorithms; coupled simulation is computationally infeasible, and (2) layered simulation is better suited for our goal of the organs matching the MRI scans

in example poses. Namely, a fully coupled hand has too many diverse tissues and too complex interdependencies, which makes it intractable to simultaneously match the MRI scans of the different hand organs in multiple poses.

We demonstrate that we can closely match the ground truth medical images in each example pose (Section 8, Figure 32, Table 4): all the organs are volumetrically matching their shapes in the MRI scans, with average errors of less than 1mm in all example poses for skin, as compared to an optical scan. Furthermore, we produce a good match to optical scans even in non-scanned poses (Section 8, Figure 33). Our volumetric simulation produces realistic organ shapes across the entire range of the hand's motion (ROM), by nonlinearly "interpolating" the shapes of organs seen in the MRI scans. We are not aware of any prior work that has demonstrated volumetric simulation of the entire hand's musculoskeletal system that matches medical image ground truth data, and that stably interpolates to the entire hand's ROM. We use our method to produce volumetric renders of the hand organs under complex nonlinear motion as the hand actuates (Figures 1, 2). Furthermore, our model qualitatively reproduces important features seen in the photographs of the subject's hand, such as similar overall organic shape and formation of bulges due to the activated muscles.

2 RELATED WORK

Human hand is one of the most flexible organs of the human body and plays a critical role in human interaction with other objects. Therefore, it has been intensively studied, not only for medical purposes but also in robotics, virtual reality, games and VFX.

The human hand is biomechanically complex. It consists of 27 bones, 34 muscles and over 100 ligaments and tendons, which are all confined to a small volumetric region. Due to this, most of the existing methods skip the complexity of the anatomy, and focus on the external hand appearance. Therefore, the widely adopted methods are data-driven methods. The human hand can simply be modeled using skinning [Jacobson et al. 2014; Kavan et al. 2008] or implicit methods [Vaillant et al. 2014]. Modeling human hand using skinning often produces artifacts, because the underlying skeleton is imprecise and the skin of the hand does not usually follow such a simple model. As shown in [Wang et al. 2019], a simple skinning method produces suboptimal results due to the incorrect positions of joint centers. To fix these issues, Pose-Space Deformation (PSD) has been proposed to incorporate artist-corrected pose shapes [Lewis et al. 2000]. Artists can incrementally add "fixes" to the existing skinning model to avoid artifacts and produce deformation closer to real human hands. Kurihara and Miyata [2004] presented a variant of PSD suitable for hand animation, and Rhee et al. [2006] demonstrated how to efficiently implement it on a GPU. PSD has been widely used in industry due to its fast performance, simplicity and the ability to incorporate real-world scans and arbitrary artist corrections. Human hands have also been successfully modeled using a SMPL model [Romero et al. 2017] or a MANO model [Li et al. 2021, 2022; Romero et al. 2017]. To fit such a model, hundreds of 3D hands were captured using a complex optical capturing system [3dMD 2022]. The database is then used for reconstructing personalized human hands from a few images [Qian et al. 2020]. Compared to these methods that focused on the external appearance of the hand, we tackle the problem of modeling and animating complex internal anatomy, and simultaneously produce high-quality output skin shapes through physically based simulation.

Recently, a data-driven method ("NIMBLE") was presented to animate internal hand anatomy [Li et al. 2022]; this method is completely data-driven (no simulation). Unlike our work, it is not focused on animation: their video only shows a few seconds of animation, and their hand motion is considerably simpler than ours. Our anatomical model is also significantly richer: NIMBLE has 7 muscles (vs ours 15), and has no ligaments, tendons, fascias, fingernails or fat. Please note that NIMBLE requires labeling segmentation masks on each MRI slice, which is a substantial manual effort. We note that physically based simulation has the benefit that, once setup, it can generate the human hand in an *arbitrary* pose; and as such the range of motion of the human hand can be "mined" fully automatically merely with simulation, which has great benefits for machine learning of hand shapes. It is our belief that future Metaverse hand applications will likely employ a combination of physically based simulation and extensive optical scanning. Finally, data-driven methods are complementary to simulation: simulation output can serve as input to SMPL, MANO, NIMBLE, etc.

We simulate hands using physically-based modeling, which requires one to first acquire accurate internal anatomy. Anatomical models can be created manually by artists based on the medical literature or medical images, but the process is very time consuming [Sifakis et al. 2005], or does not produce anatomy closing matching MRI images [Sachdeva et al. 2015]. Anatomy transfer has been successfully used for generating new human body anatomy given existing templates [Dicko et al. 2013; Kadlecek et al. 2016; Saito et al. 2015]. However, compared to a human body, a human hand is small yet complicated. Errors of a few millimeters can already cause large artifacts, because some tissues, such as distal phalanx bones or tendons, are merely 1-2mm in diameter. Moreover, existing high-quality human hand anatomy models are rare and used mainly for educational purposes. We examined well-known hand templates [C. Erolin 2019; Zygote 2016]; despite their tissues being complete and generally correctly positioned, they are primarily used for anatomy demonstrations. Their organ shapes tend to be small with empty space in between, whereas in MRI images, organs are tightly packed. Therefore, we acquire internal anatomy directly from medical imaging and optical scanning [Wang et al. 2019]. We choose MRI because it enables extraction of many different soft tissues. We then segment the tissues using existing methods [Wang et al. 2019, 2021].

Many methods have been developed for physically-based modeling of human hands. For modeling the entire hand, the existing approaches entail simulating a single soft tissue mesh constrained to the underlying skeleton [Capell et al. 2005; Garre et al. 2011; Kim and Pollard 2011; Kry et al. 2002; Liu et al. 2013; McAdams et al. 2011; Smith et al. 2018; Wang et al. 2019]. Due to modeling all soft tissues as a single volumetric layer, these methods fail to capture many key features of the human hand. Lee et al. [2009] modeled the upper human body using anatomically based simulation comprehensively. However, they only modeled human hands kinematically, without simulating the anatomical structure of the hands. In contrast, we model bones, muscles, tendons, ligaments

and fat, all resolved as separate volumetric objects. To the best of our knowledge, there are no good models for simulating hand internal anatomy that demonstratedly match medical imaging across multiple poses. We are also not aware of any methods to model the volumetric three-dimensional motion of internal hand organs in the medical literature; instead, the focus in medicine is to understand static hand shapes [Kapandji 2009], at best with minimal animation as needed to understand how to treat hand disease and injury.

We treat each tissue of a human hand separately. For bones, Kurihara and Miyata [2004] gave a bone rig extracted from multiple CT scans. Prior work also analyzed hand bone motion using MRI [Miyata et al. 2005; Rusu 2011; Stillfried 2015; van der Smagt and Stillfried 2008; Wang et al. 2019]. Keller et al. [2022] inferred the anatomic skeleton of a person from the 3D body surface. We follow the approach from [Wang et al. 2019], because it gives an artist-friendly rigging system built from MRI while simultaneously closely matching the MRI data.

Abdrashitov et al. [2021] proposed a new shape representation of musculoskeletal tissues, and an intuitive user interface to help ease the complexity of modeling volumetric anatomy. Angles et al. [2019] modeled muscles as a collection of generalized rods with volume conservation. Modi et al. [2021] proposed an efficient finite element scheme to simulate bulky muscles with heterogeneous materials. Such models have many parameters to tune and are not designed to precisely match markers delineated in MRI images. Muscle activation is often modeled by a widely used Hill model [Lee et al. 2018, 2009; Sifakis et al. 2005; Zajac 1989]. However, extracting muscle fiber directions from MRI is not easily feasible or reliable. It requires imaging techniques such as diffuse tensor imaging, which is prohibitively expensive and rarely available. If not available, it is only possible to approximate fiber fields based on the shape of the muscle [Saito et al. 2015]. When attempting to match muscle shapes to medical images in multiple poses, these facts make it difficult to apply Hill's model. Muscle activation can also be modeled using plastic deformation [Ichim et al. 2017; Saito et al. 2015], or implicit skinning [Roussellet et al. 2018]. For hand muscle activation, we adopt the 6-DOF plastic model of [Ichim et al. 2017] previously proposed for facial muscles. Compared to their method, we modeled each muscle as an individual object. This is because muscles are heavily sliding against each other inside the human hand, and therefore embedding them into a single mesh causes artifacts. To the best of our knowledge, no existing methods have previously been given to build a muscle system that matches ground truth internal anatomy seen in medical images in multiple example poses.

Existing methods for modeling tendons are primarily used for controlling hand articulation, or for medical applications [Dogadov et al. 2017; Fok and Chou 2010; Sachdeva et al. 2015; Sueda et al. 2008]. While these methods are sophisticated, the input anatomy must be created manually, and often does not precisely match any real data. Prior work has attempted to extract tendons from medical images such as MRI [Chen et al. 2011; Garland et al. 2018; Wilson et al. 1999] or ultrasound [Kuok et al. 2020], but they only focused on one or a few local regions of the hand in a single pose. We model tendons as discrete elastic rods, similar to [Bergou et al. 2008] and [Kugelstadt and Schömer 2016]. However, these previous methods were designed for forward simulation and not solving inverse problems

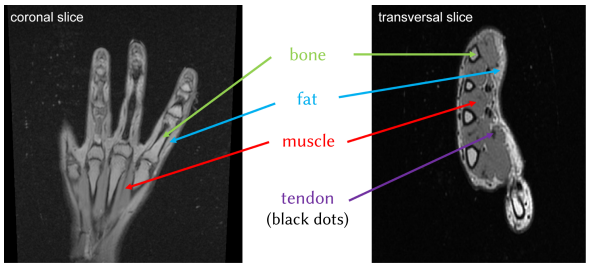


Fig. 3. **Human hand MRI slices (coronal and transversal plane).** Hand has several tissues represented by different MRI intensities.

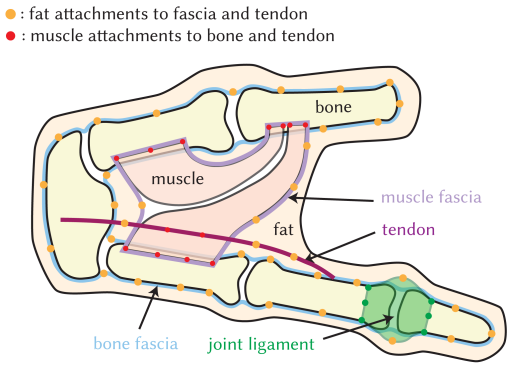


Fig. 4. **Overview of our human hand rig.** We model tendons (dark red), fat (light orange), muscles (light red), bones (yellow), and ligaments (green) around joint regions. To improve simulation quality, we also model two additional fascia layers: bone fascia (blue) and muscle fascia (purple). The fat tissue is directly attached to fascia layers and ligaments.

on rods; Bergou et al. [2008] used positions as primary variables and inferred orientation (normals) indirectly, whereas Kugelstadt and Schömer [2016] used quaternions to represent rotations. In contrast, we directly use vertex positions *and* segment normals as our primary degrees of freedom, as this better fits into the iterative closest point (ICP) algorithm. We can thus more easily incorporate real MRI tendon data to guide our tendon simulations. For example, our sliding constraints are naturally expressed as positions, skinning can directly use the position and normals to form the local coordinate system at each line segment, and non-rigid registration naturally operates with positions and normals also.

3 OVERVIEW OF OUR HUMAN HAND MODELING

A human hand contains multiple organs (Figure 3) with diverse function and mechanical properties. For example, a fat tissue is passive and softer than muscle (even when unfired); but muscles are of course active. In order to capture this variability, we model tendons, fat, muscles, bones, and ligaments as separate simulation objects (Figure 4). The input to our system is an animation of the hand's joint hierarchy, obtained using any suitable method, such as motion capture, keyframe interpolation or monocular optical tracking. Our simulation proceeds in layers (Figure 5). The output of our system is the animation of the skin (outer surface of the fat), as well as animation of the other musculoskeletal organs.

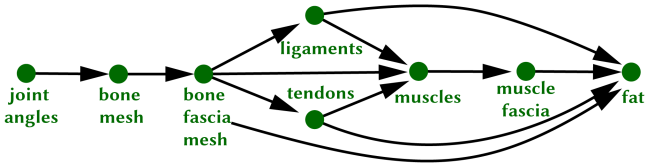


Fig. 5. **Our simulation layers.** The input to our system is a hand joint angle animation. The output is the animated external surface of the fat ("skin"), as well as the animated shapes of the internal musculoskeletal organs. Arrows denote dependencies: before a layer (green node) can be simulated, all inputs must already be simulated.

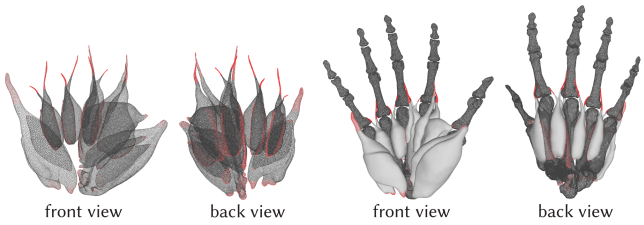


Fig. 6. **Muscle attachments.** Muscle attachment vertices are depicted as red dots. To improve the viewing of attachments, the muscles are visualized in dark wireframe in the left two figures. In the right two figures, we show bones in dark wireframe to display the relative bone locations.

In order to animate the geometry of the bones (i.e., the bone triangle meshes), we adopt the method and data made publicly available by [Wang et al. 2019]: the translation and rotation of each bone relative to the parent bone is calculated using a data-driven method, trained by observing bone motion in multiple hand poses using MRI. This gives a deterministic method that correctly positions each bone mesh for any plausible joint angles (pose), across the range of motion of the hand. Because we also use the MRI scans made publicly available by [Wang et al. 2019] to model the rest of our hand anatomy, the subject used in our work is the same as that used by [Wang et al. 2019] (male, late 20s), i.e., our non-bone simulation layers use the same MRI data that was used to create the bone rig, giving us internal consistency of our layers.

We then perform a bone fascia simulation to fill out the valleys between the bones, using a cloth solver (Section 6). Thereafter, we perform tendon simulation, using the computed bones and bone fascia mesh animations; our tendon hybrid data-driven + simulation method is novel, and is described in Section 5. Next, we perform our novel muscle simulation (Section 4), again treating the results of the previous layers as known fixed mesh animations; followed by a muscle fascia simulation. The next step is to perform joint ligament simulation. Ligaments are treated similarly to muscles, namely controlled by plastic strains. Finally, we perform our novel fat simulation, by constraining it to tendons, the muscle fascia, the bone fascia, and joint ligaments, again treating all previously simulated objects as fixed mesh animations (Section 7). Finally, we render the results using volume rendering, and standard surface-based rendering techniques (Maya Arnold and Pixar Renderman).

We extensively use the ICP algorithm in several parts of our system. Except where we explicitly state that we used Wrap3 [Wrap3 2018], we perform ICP using our implementation of [Amberg et al. 2007].

4 MUSCLES

We start the description of our layered simulation by describing our muscle preprocessing pipeline and simulation model. Although muscles are not the first simulation layer, they are the technically most complex and most important part of our paper, hence we describe them first. As always, before simulating muscles, the assumption is that the previous simulation layers have already been completed (Figure 5).

4.1 Muscle Simulation

The shapes of our muscles evolve under constraints to the bones and other muscles, as well as due to muscle firing. We model muscle firing using pose-varying muscle plasticity, controlled by joint transformations. We model our muscles as a coupled flexible multi-body dynamic system, simulated using the Finite Element Method. Each muscle is a separate deformable object. In our work, we are interested in static shapes under the given plastic strains. We found that results are more stable when performing dynamic simulations, using a simple and stable integrator, namely implicit backward Euler, as opposed to employing quasi-static solving. This is because the presence of mass and damping in the dynamic simulation acts as regularization. The dynamic simulator does produce some secondary motion, but it is extremely small and we neglect it.

Constraints: We apply four types of constraints to mimic muscle biomechanical behavior. They are (1) muscle attachments to bones and tendons; (2) muscle contacts to bones; (3) muscle inter-contacts and self-contacts; and (4) muscle inter-sliding. Namely, muscles are attached to the bones and tendons (1), they cannot penetrate the bones (2), other muscles (3), and themselves (3). Muscles are also sliding against neighboring muscles (4); this is modeled by sliding constraints. If there are no sliding constraints, large empty space appears between neighboring muscles, which is unrealistic. To specify the sliding constraints between a pair of adjacent muscles, we first find the "proximity" vertices between them, in each example pose. "Proximity" vertices are those either in contact with, or close to the neighboring muscle. The sliding vertices for this pair of adjacent muscles are the intersection of proximity vertices across all example poses. As shown in Figure 8, a muscle is attached to several rigid bones. The bones undergo rigid motions around their parent bones [Wang et al. 2019]. The attachment vertices (in red) move rigidly with the attached bones.

Pose-varying muscle activation: As stated in the introduction, we limit muscle activation to concentric isotonic contractions. Under this assumption, muscle activation becomes a unique function of the hand's pose. This means that we can model activation by calculating a pose-varying plasticity field across the hand: in each pose, the shape of the muscle is obtained by calculating the static equilibrium under the given "plastic" deformation in this pose, the nonlinear muscle elasticity, and the constraints. We note that this is related

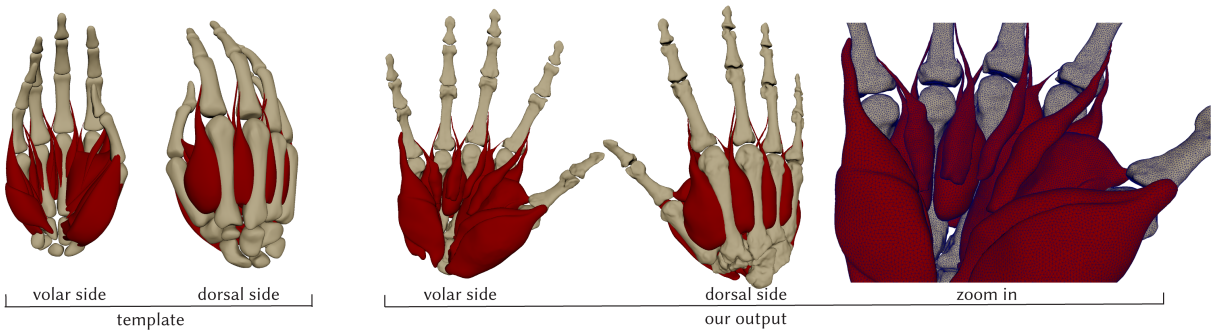


Fig. 7. **Seventeen muscles of the human hand extracted from MRI.** Observe that the template hand is larger than the scanned hand. The pose is also different. Our method addresses this using bone attachments.

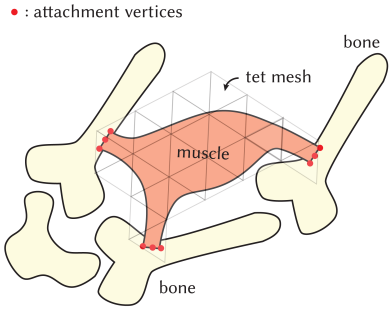


Fig. 8. **Overview of muscle simulation.** A muscle is in general attached to several bones (and/or tendons). The muscle surface mesh is embedded into a tetrahedral mesh. The pose-varying muscle contraction is modeled via the plastic strain of each tetrahedron.

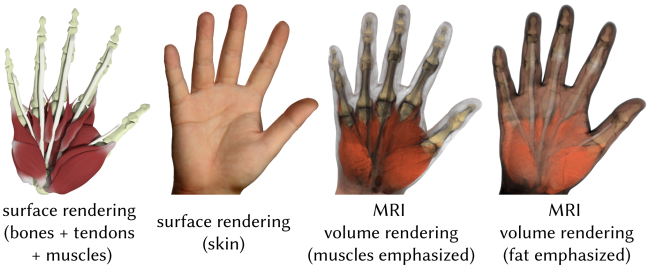


Fig. 9. **Neutral pose of the hand and its interior musculoskeletal structure as modeled in our method.** Note that muscles are not completely relaxed in this pose. However, the pose is close to the "true neutral pose" (relaxed muscles), and is easy to capture and process (no occlusions).

to the 6-DOF plastic model from [Ichim et al. 2017]; however, we do not embed all muscles into a single global tetrahedral mesh, but rather model each muscle as a separate object with its own tetrahedral mesh. The first reason for this choice is that muscles are very often undergoing sliding contact against each other, in addition to attaching to certain locations, which cannot be modeled using a single tetrahedral mesh. The second reason is that it is much more feasible and robust to define the pose-space for each muscle individually, as opposed to employ a global pose-space for all the muscles. Unlike the human face, human hand has a highly concave shape and has many flexible joints. If a single tetrahedral mesh was used, each joint would contribute several DOFs into the global pose-space, thereby causing a very large final dimension of the pose-space. This would be problematic given that our example poses are sparsely distributed in the pose-space, resulting in lower interpolation accuracy.

What remains to be discussed is how one calculates the plastic strains at each tetrahedron in the given (arbitrary) pose. We do this by extracting plastic strains in each *example* pose, from the MRI scan (Section 4.4). Next, we interpolate those strains to the given (arbitrary non-example) pose. The weights for this interpolation are different for each muscle, and are determined by defining a low-dimensional pose-space for each muscle, described in Section 4.3. During each timestep, we convert the bone rigid transformations to the pose-space vector a for each muscle (Section 4.3). The interpolation weight w_i for example pose i is then defined as

$$w_i = \frac{\phi(a, a_i)}{\sum_j^N \phi(a, a_j)}, \qquad \text{for} \qquad \phi(a, b) = \frac{1}{\|a - b\|}, \tag{1}$$

where N is the number of example poses. Plastic strain $P \in \mathbb{R}^{6m}$ of the muscle is computed as

$$P = \sum_i^N w_i P_i, \tag{2}$$

where a_i and $P_i \in \mathbb{R}^{6m}$ are the pose vector and example plastic strain at example i, respectively, and m is the number of tets of this muscle. The plastic strain vector P_i has 6 entries per tet, i.e., symmetric part of a 3×3 matrix. We use isotropic elastic materials (stable neo-Hookean material [Smith et al. 2018]) for our muscles (and also fat). Therefore, for each tet, only the the symmetric part of the 3×3 plastic strain matrix matters, as the rotation is absorbed by the elastic material; therefore, P_i has 6 entries per tet.

We do not use radial basis functions to interpolate the plastic strains, because this introduces negative weights, which in turn causes the determinant of the plastic strain to be negative or close to zero, thereby leading to simulation instabilities.

We simulate joint ligaments using the same method as muscles (Figure 10); except that we do not model contacts and sliding between ligaments, as they are not in close geometric proximity. Therefore, we can simulate each joint ligament separately (Figure 10).

4.2 Creating Neutral Muscle Shapes From MRI

During preprocessing, we extracted the shapes of all muscles in the *neutral pose* (Figure 9); we use the method described in [Wang et al. 2021]. This method starts from a generic surface and tetrahedral mesh muscle template, and reshapes both the surface and tetrahedral template to match the MRI image. Matching the medical image involves "landmarks" (locations on the template muscle and MRI image that are known to correspond to the same anatomical location), "ICP markers" (location in the MRI image that are, with a high degree of confidence, located on the boundary of the muscles; without there being a known correspondence on the template), and "attachments" (locations where a muscle attaches to a bone). We manually specified landmarks, attachments and ICP markers in the MRI images, with the aid of medical literature and a medical doctor radiologist. We first specify the attached vertices on the template mesh. Then, we extrapolate the vertex positions based on the deformation from the template bone meshes to our bone meshes, followed by any (typically minor) manual adjustments to ensure the final positions match the MRI. The resulting attachments are shown in Figure 6. The entire process to create the neutral muscle surface mesh and tet mesh took approximately 3 days total for all the 17 muscles (Figure 7) of the human hand, including manual work and computer time.

4.3 Muscle Pose Space

Without loss of generality, consider a single hand muscle. A single muscle usually attaches (originates/inserts) to several bones and tendons. The tendons, as described subsequently (Section 5), are also controlled by the motion of the bones. If a muscle is attached to tendon(s), then we treat the bones that control the tendon as the bones that also control the muscle, in addition to the directly attached bones. Thus, we can consider that the muscle is solely controlled by bones. We define the muscle pose space using the skinning rotation of the controlling bones. Skinning rotation of a bone is the rotation of the bone relative to its neutral pose. This rotation is commonly used, for example, in linear blend skinning. Because applying the same global rotation to all controlling bones should not affect the pose space, we eliminate one controlling bone from the pose-space. We express the skinning rotations of all other controlling bones relative to the skinning rotation of that bone, and then remove it. The selected/removed bone is the controlling bone that has the largest number of descendant controlling bones.

To represent a rotation, there are several choices: (1) a quaternion; (2) a 3D rotation matrix; (3) a 3D axis-angle vector ("exponential map" representation). Using a 3D rotation matrix leads to a high dimension of the pose space. In addition, when using a 3D rotation matrix or a quaternion, it is more difficult to measure the distance to the example pose; this is needed to compute interpolation weights. In contrast, a 3D axis-angle vector only has 3 DOFs, resulting in a compact pose space. As it is defined in the Euclidean space, one can easily compute the distance to each example pose. As is well-known, adding any integer multiple of 2π to the angle produces the same rotation. We first convert the rotation matrix to a unit quaternion $q = cos(\phi/2) + sin(\phi/2)a$, and finally to the 3-vector ϕa, where we enforce $\phi \in [-\pi, \pi)$. Note that using q vs $-q$ increases ϕ by 2π, but this disappears when standardizing to $[-\pi, \pi)$, i.e., identical ϕa is produced. Limiting the angle to $[-\pi, \pi)$ does not produce discontinuities in our method because no pair of controlling bones for the same muscle undergo a rotation greater than 180 degrees across the range of motion of a hand. There are pairs of joints in the human hand that can undergo a relative rotation greater than 180 degrees (but always less than 360 degrees), e.g., fingertip joint relative to palmar bones; but those do not appear as controlling bones of the same muscle in our hand model. Because the rotation is always less than 360 degrees, if such bone pairs needed to be accommodated, we could shift the $[-\pi, \pi)$ interval to a more suitable per-muscle interval, e.g., $[-\pi/4, 2\pi - \pi/4]$ or similar.

We thus convert all bone transformations into 3D vectors, and combine them into a per-muscle pose-space vector. If a muscle is controlled by N_e bones, the dimension of the pose-vector for this muscle is $3(N_e - 1)$. When applied to example pose i, this gives us the pose vector a_i for this muscle. Note that different muscles have different pose-spaces and different controlling bones and a different N_e. The maximum value of N_e for a muscle in our hand model is 3. Note that we only model muscles within the hand; we do not model lower and upper arm muscles.

4.4 Muscle Plastic Strain Extraction in Example Poses

We now describe how we find the plastic strains P_i in all example poses i, so that the plastic strains change smoothly with pose, and so that, in each example pose, the muscle in static equilibrium under the plastic strain P_i matches the MRI landmarks and ICP markers of that example pose, and the muscles are not colliding with each other. Namely, we need to achieve the property that, for each muscle, for similar poses, the corresponding plastic strains are similar. We observed that without such pose-space smoothness, the muscle simulation outputs are visually temporally non-smooth. We now describe the steps of this process. We define plastic strains on a single per-muscle tetrahedral mesh for all example poses, i.e., per-muscle, there is the same surface and tet mesh topology for all poses, but different vertex positions. Before settling on our solution method below, we tried other methods, which did not work well and we abandoned them; we describe them in our Supplementary Material.

Our method employs three stages. Stage 1 entails executing [Wang et al. 2021] in each pose, as described for the neutral shape above in Section 4.2, with some additional "helper" algorithms to provide a better initial guess than starting from the neutral tet mesh. Our initial guess is obtained simply by simulating the hand to each example pose, using the bones, bone fascia, tendons, and muscle tet meshes generated in the neutral pose. When this proved too challenging due to example poses being too different from the neutral pose, we relied on a "pre-initial" guess obtained using optimization in the "computational bodybuilding" [Saito et al. 2015] space (Supplementary Material). The output of the method of Section 4.2 applied to this non-neutral pose is a surface mesh (and a tet mesh) matching the MRI scan in this pose. Two problems now become readily apparent: as we extracted muscles separately, the different muscle meshes collide in this pose, and; for each muscle, the plastic strains induced by the optimized tet mesh are (in general) *not* a smooth function of

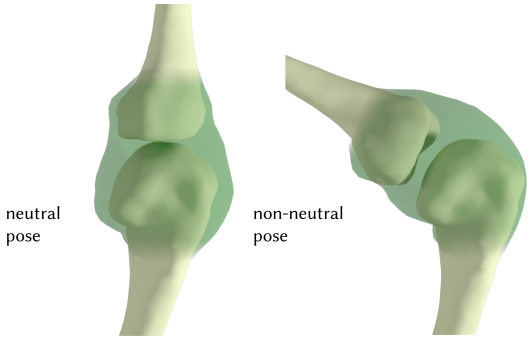


Fig. 10. **Our simulated ligament around a hand joint, in two poses.**

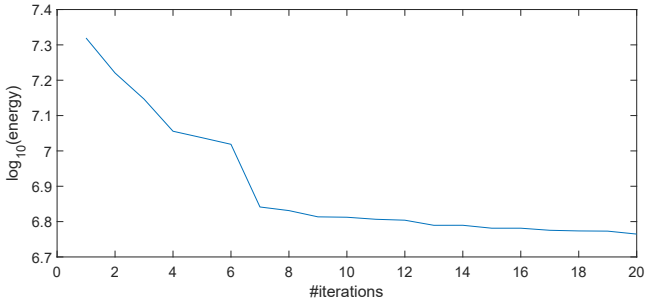


Fig. 11. **The energy of our objective function (Equation 3) in each iteration**. The curve shows the energy in each iteration when we optimize the plastic strain for the thumb muscle group.

the pose. For this reason, we discard the tet meshes, and proceed to Stage 2 whereby we correct the Stage 1 muscle surface meshes to not be colliding.

A volumetric method to remove muscle interpenetrations was reported in [Wang et al. 2019]. However, we found that it takes hours to resolve the contact in a single example pose, which is too slow to process all muscles in all example poses. This is because the muscle volumetric meshes have many DOFs, and the effect of contact propagates volumetrically, thereby imposing a large computational burden. In contrast, the surface of the muscle is easier to deform under contact if a surface-based method is used. Therefore, we prefer a surface-based deformation method. Our Stage 2 entails using the ShapeOp bending energy [Bouaziz et al. 2014] combined with a contact penalty energy term that pushes each volumetrically colliding vertex to the closest point on the surface; this dramatically improved performance. In our experiments, it takes less than 10 minutes to resolve the inter-contacts between muscles for each example pose.

Finally, in Stage 3, we ensure pose-space smoothness of the plastic strains. The output of Stage 2 are non-colliding muscle surface meshes (in each pose) that are close to MRI landmarks and ICP markers. We define the following energy which (separately for each muscle), jointly optimizes the plastic strains at all example poses, in a manner that causes the embedded muscle meshes to closely match the outputs of Stage 2, while ensuring pose-space smoothness. We note that, in order to avoid "spikes" at the sparse MRI landmarks and ICP markers, we intentionally do *not* use these landmarks and markers in this process here, but instead use a "dense correspondence" against the surface mesh output of Stage 2. This also safeguards against re-introducing muscle contacts. We optimize

$$\underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\arg\min} \quad \sum_{i=1}^{N} \left(||\mathbf{L}\,\mathbf{S}(\mathbf{x_i})||^2 + \alpha \mathcal{E}_{\text{Dense}}(\mathbf{x_i}) \right) + \\ +\gamma \sum_{j=1}^{m} ||\mathbf{L}_{\text{pose}} \left[\mathbf{S^j}(\mathbf{x_1}),\, \mathbf{S^j}(\mathbf{x_2}),\, \ldots,\, \mathbf{S^j}(\mathbf{x_N}) \right]^{\mathbf{T}} ||^2, \tag{3}$$

where $x_i \in \mathbb{R}^{3n}$ are tet mesh vertex positions in example pose i (number of tet mesh vertices is n), N is the number of example poses and $\mathbf{L} \in \mathbb{R}^{6m \times 6m}$ is the volumetric mesh Laplacian (extended trivially to operate on 6-vectors; m is the number of tets). The function $\mathbf{S}(\mathbf{x}) \in \mathbb{R}^{6m}$ first calculates the deformation gradient at each tet, relative to the neutral shape, and then performs polar decomposition to extract the symmetric part and stores it (6 entries per tet, denoted by $\mathbf{S^j}(\mathbf{x})$); the gradient and Hessian of $\mathbf{S}(\mathbf{x})$ (needed for optimization) can be computed as in [Wang et al. 2021]. Under this definition, the objective function does not penalize the growth of the object and is unaffected by rotation. The quadratic term $\mathcal{E}_{\text{Dense}}(\mathbf{x_k})$ sums the squared surface mesh vertex Euclidean distances to the muscle surface output of Stage 2. The matrix $\mathbf{L}_{\text{pose}} \in \mathbb{R}^{N \times N}$ measures pose-space smoothness of plastic strains. It is the graph Laplacian matrix of the *pose graph* defined as follows: nodes are example poses, and edges connect each pose to all other poses, with weights inversely proportional to pose-space distance. Details are in Appendix A.

To optimize Equation 3 efficiently, we first solve N separate optimization problems without the pose-space smoothness term, starting from the neutral tet mesh. The solution for each example pose is used as the initial guess $\mathbf{x}$ to our optimization problem (Equation 3), which we solve using block-coordinate descent: in each iteration, we optimize for one $\mathbf{x_i}$, while freezing all $\mathbf{x_j}$ for $j \neq i$. We repeat the entire block-coordinate iteration (all N poses) 20 times, for all our muscles. The output vertex positions $\mathbf{x_i}$ are then converted to plastic strain by calculating the deformation gradients relative to the neutral shape. The weight γ is tuned based on the maximum distance between the output surface and the input surface. We use 0.5mm as the maximum allowed distance. By using the strategy of bisection, we can automatically determine the value of γ for each muscle. Note that we did not add an explicit constraint that guarantees that the deformation gradients corresponding to $\mathbf{x}$ are positive-definite. This is because even without the constraint the eigenvalues of our resulting plastic strains for all muscles in all example poses were between 0.08 and 2.3 in our experiments. Moreover, it makes the optimization much easier. Our optimization converges for all muscles. The energy of the objective function in each iteration for one of the muscles are shown in Figure 11. The entire optimization typically takes 2.5 hours for all poses and all muscles.

5 TENDONS

Tendons are important not only because they control the motion of the hand, but also because they affect the appearance of the hand. Tendons are often beam-shaped and are always black in the MRI image, thereby giving them good contrast to the neighboring tissues.

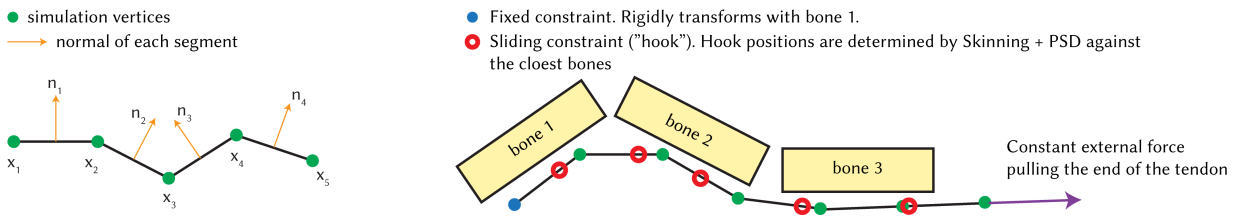


Fig. 12. **Tendon simulation model.** Left: our tendons are simulated as rods consisting of multiple segments. Each segment is represented by the positions of two end vertices and a normal defining the orientation. The normal (orange) is perpendicular to the segment. The rod is constrained to slide through a few locations ("hooks"; red circles) that are skinned (with PSD correction) to the closest bone(s). One end of the tendon is attached to a bone with a fixed constraint. We apply a constant force (purple) at the other end of the tendon to stretch it without invalidating the constraints.

However, tendon extraction is difficult in practice because of the limited resolution of the MRI; thin tendons are often impossible to extract. We model tendons extruded from the flexor digitorum superficialis/profundus muscles, flexor pollicis longus muscle, extensor pollicis longus muscle, extensor digitorum muscle, extensor indicis muscle and extensor digiti minimi muscle. All tendons are very dark (black) in the MRI. Thus, if tendons travel through the same pulley, it is difficult to distinguish them in the MRI. To address this issue, we group tendons that go through the same pulley into a tendon group. There are 10 tendon groups in our system. As a side effect, however, the tendons from the same group cannot slide relative to each other, which is a limitation of our work.

5.1 Tendon Simulation

Tendons in the real world and in our work serve as attachments for muscles, muscle fascia and skin, and therefore they substantially affect the simulations of those layers. In Section 8, we give an ablation study where we demonstrate that hand knuckles are not properly resolved without tendons (Figure 31, (a)). Our tendon simulation proceeds as follows. As illustrated in Figure 12 (left), we first discretize the tendon into Q small segments (Q is approximately 300 on average in our examples). Each segment i is controlled by the two endpoint vertex positions x_i, x_{i+1} and the normal of the segment n_i. Normal n_i is perpendicular to segment i. The simulation DOFs are x_i and n_i. As shown in Figure 12 (right), one end of the entire tendon (blue) is attached to the closest bone ("bone 1") and is rigidly transformed with it. The other end of the entire tendon is pulled by a constant external force (purple) in the longitudinal direction to mimic the fact that the muscles of the forearm are pulling the tendon. The tendon is constrained by a series of points (shown in red circles) located near the bones that we call "hooks". **The tendon centerline is constrained to slide through the hooks.**

Our tendon model is designed to keep the length of the tendon constant, and prevent tendon twisting. Note that our hooks are not biological (they do not exist in the real hand). We use them to cause the tendon simulation to conform to the MRI data; i.e., our hooks enable us to fuse tendon physically based simulation with MRI data. We note that Sachdeva et al. [2015] also modeled hand tendons biomechanically using rods. In their model, tendons slide through real anatomical pulleys, which were obtained manually by referencing the medical literature. Their goal was to create a robust control system for the hand, and not match MRI data for a specific individual. The real anatomical pulleys are unfortunately not visible in our MRI images; and hence we did not adopt their method and instead use data-driven hooks.

Given the current hook locations $\bar{x}_1, \ldots, \bar{x}_k$, constant external forces and fixed attachments, we deform our tendon by minimizing

$$\arg\min_{x,n,t} E_{\text{sliding}}(x,t) + c_1 E_{\text{pulling}}(x) + c_2 E_{\text{twist-bend}}(n) \tag{4}$$

$$\text{subject to } (x_{i+1} - x_i)^2 - \ell^2 = 0, \quad \forall i \in [1, Q], \tag{5}$$

$$n_i^T (x_{i+1} - x_i) = 0, \quad \forall i \in [1, Q], \tag{6}$$

$$n_i^T n_i - 1 = 0, \quad \forall i \in [1, Q], \tag{7}$$

$$0 \le t_j \le 1, \quad \forall j \in [1, k], \tag{8}$$

$$x_1 = \hat{x}_1. \tag{9}$$

The energies are defined as

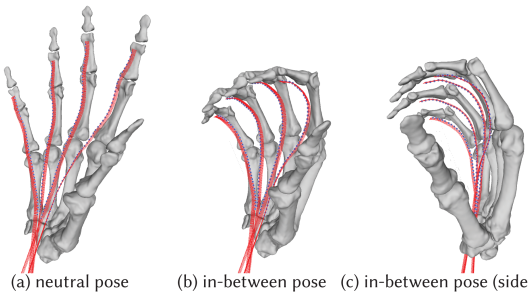
$$E_{\text{sliding}}(x,t) = \sum_{j=1}^{k} \left((1 - t_j) x_{s_j} + t_j x_{s_j+1} - \bar{x}_j \right)^2, \tag{10}$$

$$E_{\text{pulling}}(x) = -x_{Q+1}^T f, \tag{11}$$

$$E_{\text{twist-bend}}(n) = -\sum_{i=1}^{Q-1} n_i^T n_{i+1} - c_3 n_1^T \bar{n}_1. \tag{12}$$

Here, k is the number of hooks, and $t_j \in [0,1]$ (for $1 \le j \le k$) is the line segment barycentric coordinate giving the closest position to the given hook location $\bar{x}_j$; and s_j is the index of the closest segment to hook j. Within each optimization iteration, the hook is therefore constrained to stay on the same rod segment. However, we update the closest segments s_j after every optimization iteration; and this permits multiple rod segments to travel through a hook, i.e., a hook is not "stuck" on one rod segment. Parameters c_1, c_2, c_3 control the optimization tradeoffs (we use $c_1 = 0.5$, $c_2 = 1.0$, $c_3 = 1.0$), ℓ is the length of one rigid segment (computed as the total length divided by the number of segments); $\hat{x}_1$ is the fixed attachment location on bone 1; and f is the constant upper arm muscle pulling force. We define the sliding constraint as the sum of the squared distance from each hook to the closest point on the closest segment. We define the twist-bend energy as the negative dot product between two neighboring normals; the larger the dot product, the less twisting and bending. As we define our normals to point in a direction close to the bending axis, the energy $E_{\text{twist-bend}}$ is mainly penalizing the twisting (as opposed to bending). Note that it is not sufficient to determine all n_i because the tendon could be globally rotated by the same rotation. We determine the global orientation using the normal $\bar{n}_1$, calculated

by skinning the neutral pose first line segment normal with the transformation of "bone 1". Our constraints also enforce that n_i is perpendicular to the segment i, and is a unit vector. The optimization problem is solved using the interior-point method [Artelys 2019]. In practice, the optimizer was able to easily find an optimal solution. A single optimization iteration (solving Equations 4-12 under fixed ICP closest positions) takes approximately 0.1 seconds on average per tendon in our examples. Total optimization time for all tendons is approximately 6 seconds.



(a) neutral pose (b) in-between pose (c) in-between pose (side view)

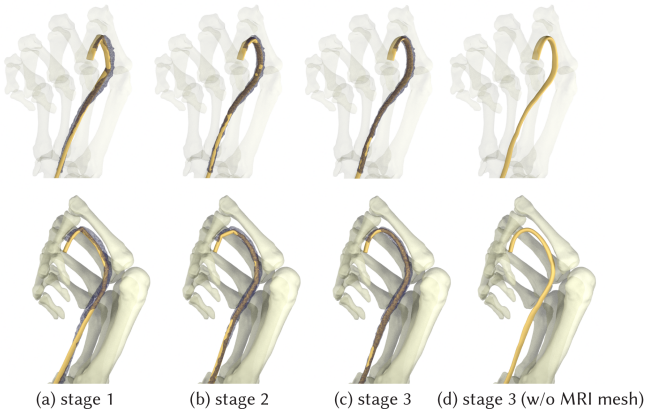
Fig. 13. **Tendon simulation.** Here, we show the rods that represent tendons. The red lines are the rigid segments and their normals. The blue dots are the hooks.

With the runtime simulation formulation now specified, the only remaining task is to determine the hook locations, the fixed attachment location, and the constant force. Fixed attachment location $\hat{x}_1$ is computed by rigidly transforming its initial position using the transformation of the closest bone ("bone 1"). The direction of the pulling force is defined as the opposite of the average direction of the last few segments in the neutral configuration; this is because these segments are usually located in the wrist region, and slide along their longitudinal (i.e., tangential) direction in the real world. At each simulation frame, we compute the hook locations using skinning, augmented via pose-space deformation ("PSD") corrections [Lewis et al. 2000], driven by bone rotations. The process to obtain skinning weights and the pose-space corrections is explained in Section 5.3. This makes the hook locations a unique function of the bone transformations. Figure 13 shows the tendon hooks (blue), and the tendon simulation results.

5.2 Tendon Extraction From MRI

To perform the simulation, we need to know the tendon hook locations. They are determined from MRI in multiple example poses. Therefore, we first extract the tendons from MRI. Although we model tendons as rods, real tendons are volumetric objects. To extract tendons, we first treat them as cylinder tubes whose centerlines are our simulation rods. We assume that the radius of the tendon does not change during the simulation and, thus, the surface shape of the cylinder is skinned by the centerline.

We create a template cylinder manually. We then extract the tendon mesh from the neutral pose MRI using classic computer vision techniques (3D Slicer [Fedorov et al. 2012]). Next, we non-rigidly deform the template cylinder tube to match the MRI mesh using Wrap3 [Wrap3 2018]. As real tendons and the target MRI-segmented mesh have a non-circular cross-section, so does the resulting ICP-ed mesh. The center rod is deformed by following the surface mesh, as we can consider it to be embedded into the tube volume. In this way, we create our neutral tendon simulation rod and its surface tube mesh. For non-neutral poses, we only extract the surface of the tendon from MRI, again using 3D Slicer. These segmented meshes will be used for tendon non-rigid registration in Section 5.3.



(a) stage 1 (b) stage 2 (c) stage 3 (d) stage 3 (w/o MRI mesh)

Fig. 14. **Stages to register tendons in non-neutral poses.** In stage 1, we simulate the tendons based on the sliding constraints that are only rigidly transforming with the closest bones. This gives a relatively correct position of the tendon. However, it does not match the MRI shape (blue wireframe). To improve the match, we first create an initial guess for our non-rigid registration (stage 2). As shown in (2), the resulting mesh more closely matches the MRI shape. In stage 3, we perform non-rigid registration, which enables us to match the MRI shape very closely. Top and bottom give different camera angles and change bone transparency for easier viewing.

5.3 Tendon Registration

Now, we have the tendon rod and the surface mesh in the neutral pose, and the extracted surface mesh in non-neutral poses. We need the tendon rod in these non-neutral poses. We want our tendon skinning surface mesh driven by the tendon rod to match the MRI mesh as closely as possible in each example pose. This is a standard non-rigid registration problem, but occurring on a skinned surface controlled by the rod. As is well-known, non-rigid registration requires a good initial guess to produce high-quality results. Therefore, the tendon registration at example poses is performed in three stages (shown in Figure 14). The first two stages create a good initial guess and the last stage performs the actual non-rigid registration.

Stage 1: Roughly deform the tendon to match the target example pose. We manually select a few points on the tendon rod, which we think are rigidly transforming with the closest bones. These points are treated as hooks (used only in this stage; discarded otherwise). We then slowly deform our bone rig from the neutral pose to the target example pose. In each iteration, we compute the positions of the selected points. Since they are rigidly transforming with the closest bones, the positions are easily obtainable. Then, we perform our tendon simulation to obtain the shape of the tendon using

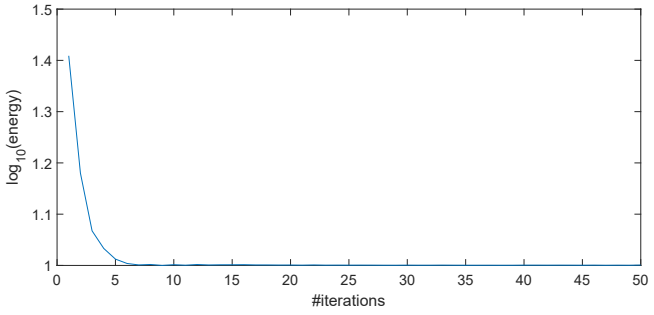


Fig. 15. **The energy of tendon non-rigid registration at each iteration.** The curve shows the energy of Equation 18 at each Newton iteration when performing the non-rigid ICP on the extensor tendon of the middle figure. The optimized energy plateau is not zero because the forearm pulling force introduces an (unimportant) constant energy offset; we added a properly chosen such an offset to improve figure readability.

Equations 4-12. The result is depicted in Figure 14(a). Stage 1 takes approximately 1 minute per tendon. We note that stage 1 is needed because otherwise the initial guess for later stages can be quite bad, due to poses being very different from the neutral pose.

Stage 2: Improve the tendon by matching the centerline of the extracted MRI mesh. After the first stage is completed, the position of the tendon in this non-neutral pose is close to the tube mesh extracted from MRI. Because the tendon is thin, we need to further match the MRI mesh to guarantee the quality of the final non-rigid registration. To do so, we first select a few points (~ 10) on the approximate tendon centerline manually, based on the MRI mesh in this pose. We use these points as hooks (again only in this stage; discarded otherwise) and run our simulation, starting from the output of stage 1. The result of stage 2 is depicted in Figure 14(b). Stage 2 takes approximately 1 minute per tendon. Stage 2 is needed because after running stage 1, sometimes the tendon surface is completely outside of the MRI surface. This causes difficulties in ICP (incorrect determination of closest point) if one goes to stage 3 directly after stage 1.

Stage 3: Deform tendon to match the extracted MRI mesh. In the last stage, we perform actual non-rigid registration by solving the following optimization problem:

$$\arg\min_{x,n} \quad c_4 E_{\text{ICP}}(x,n) + c_1 E_{\text{pulling}}(x) + c_2 E_{\text{twist-bend}}(n) \tag{13}$$

$$\text{subject to} \quad (x_{i+1} - x_i)^2 - \ell^2 = 0, \quad \forall i \in [1, Q], \tag{14}$$

$$n_i^T (x_{i+1} - x_i) = 0, \quad \forall i \in [1, Q], \tag{15}$$

$$n_i^T n_i - 1 = 0, \quad \forall i \in [1, Q], \tag{16}$$

$$x_1 = \hat{x}_1, \qquad \text{where} \tag{17}$$

$$E_{\text{ICP}}(x,n) = \sum_i \left(\hat{n}_i^T \left(\left(\sum_{j \in M_i} w_{ij} T_j(x_j, x_{j+1}, n_j) \bar{p}_{ij} \right) - \hat{p}_i \right) \right)^2, \tag{18}$$

and $E_{\text{pulling}}(x)$ and $E_{\text{twist-bend}}(n)$ are as defined earlier. Here, $\bar{p}_{ij}$ is the unskinned rest position of tendon surface mesh vertex i in the frame of reference of tendon segment j in the neutral pose; the frame of reference of a segment is defined by the two endpoints and the normal. The mapping $T_j(x_j, x_{j+1}, n_j)$ performs the skinning transformation for segment j obtained using x_j, x_{j+1}, n_j; here, w_{ij} is the skinning weight of surface vertex i against the tendon segment j. The skinning weights were determined using inverse closest distances; we use 3 segments per vertex, and those segments are stored into the set M_i. The position $\hat{p}_i$ is the ICP target location on the MRI mesh. It is updated after each iteration in the usual ICP fashion by finding the closest position on the MRI mesh to the current vertex position). The ICP target normal is $\hat{n}_i$, determined as the pseudonormal at the closest location on the MRI mesh. Again, we solve this problem using the interior-point method [Artelys 2019]. The ICP registration converged well for all our tendon groups. Figure 15 shows the energy of tendon non-rigid registration at each iteration, for one of our tendon groups. The output of stage 3 is our final registered tendon centerline and surface mesh, and is shown in Figure 14(c). As can be seen, it is smooth and closely matches the MRI mesh compared to the previous stages. We also observe that our optimization quickly converges in a few iterations, as illustrated in Figure 15. Stage 3 takes approximately 10 minutes per tendon.

5.4 Tendon Hooks

Using the method described above, we obtain all example tendon rods. Then, we sparsely sample the tendon rod in the neutral pose; these sample points are our neutral hooks. For each hook, we assume that it is driven by N_s closest bones (we use $N_s = 2$). We then find the skinning weights w by minimizing the distance of the skinned hook position to the example tendon rod mesh across all example poses,

$$\arg\min_{w} \quad \sum_{j=1}^{N} \left(\sum_{k=1}^{N_s} w_k T_{jk} \bar{p}_{ik} - \hat{p}_j \right)^2, \tag{19}$$

$$\text{s.t.} \quad \sum_{k=1}^{N_s} w_k = 1, \tag{20}$$

$$0 \le w_k \le 1, \forall k \in [1, N_s], \tag{21}$$

where T_{jk} is the example transformation for bone k in pose j, $\bar{p}_{ik}$ is the unskinned rest position of hook i to bone k, and $\hat{p}_j$ is the closest position on the rod to the hook location, in example pose j. The example rod meshes are obtained from our non-rigid registration. To solve this optimization problem, we first initialize the weight of the closest bone to 1, and the weights of the other bones to 0. Then, we compute the skinned positions in the example poses. Thereafter, we find the closest position $\hat{p}_j$ in each example pose. Next, we run our optimization to determine the skinning weights w. We repeat these steps until the change of w between two consecutive iterations is very small (relative change of 10^{-4}). After the optimization problem is solved, we store the vector between the skinned hook position and the closest point on the tendon rod as our pose-space correction in this example pose. Similarly to muscles, the pose-space vector consists of rotations of bones that drive the

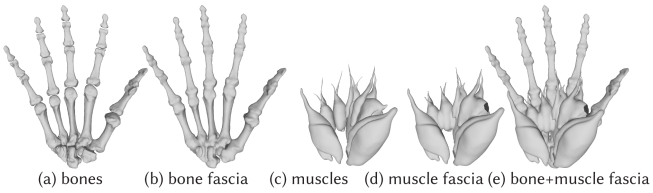


Fig. 16. **Fascia meshes.** (a) Bone meshes, (b) corresponding bone fascia mesh, (c) muscle meshes, (d) muscle fascia mesh, and (e) both fascias.

tendon. The representation of rotations as 3-vectors is the same as for muscles (Section 4.3).

6 BONE AND MUSCLE FASCIA

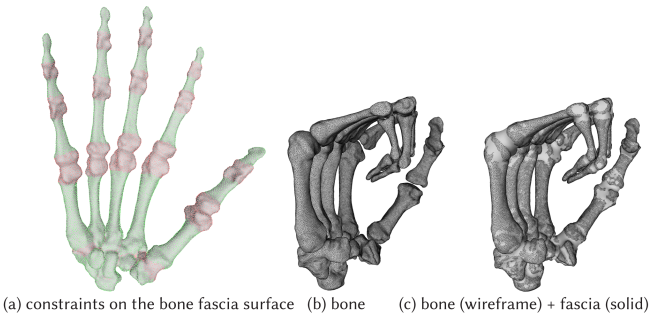


Fig. 17. **Bone fascia.** (a) The constraints applied to bone fascia simulation. Vertices in green are attached to the closest bone using fixed constraints. Vertices in red are in contact with the bone meshes. (b, c) Bones and bone fascia in the fist pose.

The gaps and valleys around bones and muscles cause problems (squeezing, mesh gets stuck, etc.) if one simulates the fat tetrahedral mesh directly on top of the bones and muscles. Therefore, we create two fascia layers, one wrapping muscles and bones, respectively (Figure 16). Each fascia is a triangle mesh. For bone fascia, we merge our bone meshes with the convex hulls of the joint space in between the bones [Wang et al. 2019]. This gives us the triangle mesh of the bone fascia. Then, we shrink the cloth material coordinates to 35% of their original size, so that the fascia tightly wraps the bone geometry. Similar to bones, we merge all muscle surface meshes together and retain only the external envelope; this produces the muscle fascia mesh. For muscles, we shrink the cloth material coordinates to 40% of their original size. Our system is not particularly sensitive to these shrinking percentages (i.e., we could easily make the two percentages equal), as long as they are small enough. The fascia meshes for bones and muscles are shown in Figure 16. In each simulation timestep, we first rigidly transform the bones and then simulate the bone fascia using a cloth solver [Volino et al. 2009]. After that, we perform muscle simulation, and then simulate the muscle fascia, also using the same cloth solver.

We now need to specify how the two fascias are constrained in the cloth simulation. There are three types of constraints in our model: fixed constraints, sliding constraints, and contact constraints,

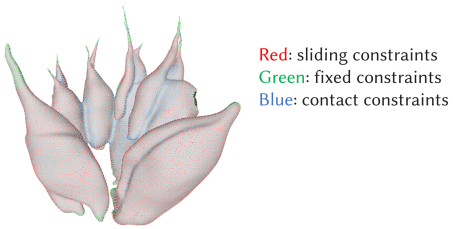


Fig. 18. **Muscle fascia constraints.** Here, we show the constraints applied during our muscle fascia simulation. Vertices in green are attached to the muscles using fixed constraints. Vertices in red are sliding against the muscle surface meshes.

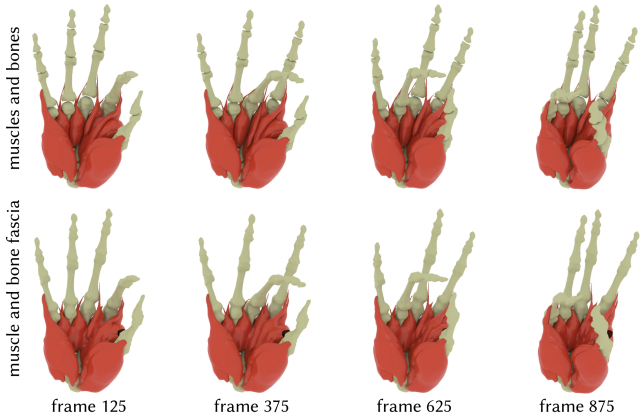


Fig. 19. **Fascia animation.** Top: a few frames of an animation sequence of bones and muscles. Bottom: muscle and bone fascia simulation results.

as shown in Figure 23. A fixed constraint anchors a point to a target location. A sliding constraint limits a point to travel in a (time-varying) target plane. A contact constraint penalizes penetration of a point beneath a (time-varying) contact plane. For bone fascia, the fascia vertices located away from bone heads are attached to the bone mesh with a fixed constraint, and move with the bone rigidly. The other bone fascia vertices (i.e., those at bone heads and at joints) are not attached, but are instead simulated to undergo contact with the bones' surface. The two types of vertices are selected manually, by painting on the bone fascia mesh in the neutral pose. Due to the attachments and contact and shrinking of the rest material coordinates, the bone fascia mesh remains tense and does not fold as the bones move (Figure 17).

For muscle fascia, we use all three constraint types: fixed, sliding and unilateral contact. We use fixed constraints near muscle insertions, sliding in regions where the muscle fascia is on top of a single muscle, and contact where two muscles meet (Figure 18). These regions were painted manually in Maya in the neutral mesh, with smooth transitions between them. Even though such a process is manual, it is a simple matter of painting regions; we use Maya as a painting tool and export the maps to our simulator. The painting usually takes less than 10 minutes; it only needs to be done once. Representative bone and fascia animations are shown in Figure 19.

7 FAT

Our fat tissue is a volume bounded by two surfaces: the exterior surface (the "skin"; Figure 21(a)), and the interior surface (Figure 21(b)) that is in contact with the bones, muscles, and joint ligaments. Each of these two surfaces is topologically a sphere. The skin surface mesh is from the same subject as the MRI scans, and was obtained from the project [Wang et al. 2019]. We mesh the volume between the skin and the inferior surface using a tetrahedral mesh, and embed the two surfaces into it. We can significantly reduce the number of tetrahedra because the volumetric mesh does not have to conform to the surfaces. The tet mesh is created by first meshing the volume enclosed by the skin, and then removing all tets that are completely inside the volume enclosed by the interior surface. We assign a single material property to the fat, based on values in medical literature (Young's modulus=1 kPa) [Comley and Fleck 2010]. After creating the tetrahedral mesh, we nevertheless need to assign spatially varying materials, because some tetrahedra are only partially occupied by the fat tissue. Therefore, we weigh the Young's modulus and mass density by the volume occupation.

We then define constraints of the fat against previous layers (Section 7.1). For all of our six MRI example poses, and six additional non-example poses, we obtained a matching high-precision (0.1 mm) optical scan from the project [Wang et al. 2019]. The six optical scans in MRI example poses serve to optimize per-pose spatially varying tet plastic strains, to cause the FEM fat simulation to match ground truth optical scans (Section 7.3). The six non-example optical scans serve to test our method in non-example poses (Section 8, Figure 33).

7.1 Constraints for Fat Simulation

After we obtain the simulation mesh, we build the constraints between the interior surface of the fat and the muscle fascia, bone fascia, and joint ligament layers. Inspired by biomechanics, for constraining the fat to the bone fascia, we create fixed constraints at vertices near the bodies of the bones, and use contact around bone heads. For constraining fat to muscle fascia, we found that the sliding constraints + contact constraints yield the best results for all vertices, except in places where the muscle is attached to bones. In such places, we attach fat to muscle fascia using fixed constraints. We place sliding constraints near the muscle fascia surface, and use contact near the valleys between muscles and/or bones. Note that if all interior fat vertices were attached to fascias using fixed constraints, this would produce visible bumps on the skin surface. We manually determined the weights for sliding constraints and contact constraints. Much like with the fascias (Section 6), this manual process is not very difficult. We use Maya's painting ability and then exported the maps into our simulator; the entire painting process takes under 15 minutes. After a simulation run, we optionally adjust the painted maps to improve the constraints, but these adjustments are also fast and not very frequent. Moreover, the weights are the same for all example poses. The procedure for the joint ligaments is similar to the muscles. We apply contact constraints to all vertices except the attachment vertices to the bones. Finally, as mentioned in [Wang et al. 2019], we also apply fixed constraints to the nails of the fingers, to keep them rigid. In addition to the described constraints on the interior and exterior surface of the fat volume, we also apply self-collision handling on the exterior surface (skin).

7.2 Modeling and Animating Finger Nails

Although finger nails can in principle deform too, they are much stiffer than fat and muscles, and we model them as rigid objects; their animation therefore entails finding their rigid body motion. We first attempted to models nails as embedded into the volumetric fat simulation, which produced visibly incorrect non-rigid nails. We then tried to rigidly transform the nails with the closest bone; however, this method also failed, producing a poor match to the nails seen in the hand skin optical scans in the example poses. To resolve this mismatch, we add a pose-space correction [Lewis et al. 2000] to the rigid transformation of each nail. To obtain the example rigid transformation for each nail at each example pose, we first perform hand simulation from the rest pose to each example pose, whereby each nail is rigidly transformed by its closest bone. Next, we perform non-rigid ICP registration to the ground truth skin mesh obtained using optical scanning. This is done by manually placing corresponding landmarks near and on the nails on the simulation output mesh and on the optically scanned skin mesh. By doing so, we obtain the ground truth nail mesh in the same mesh topology at each example pose. Next, we use shape matching [Müller et al. 2005] between the neutral pose nail and each example pose nail, to extract the example rigid transformation of the nail. Finally, the nail pose-space correction transformation is obtained by computing the difference between the transformation of its closest bone at the example pose and the example transformation. To ex/interpolate the pose-space correction to an arbitrary pose, we use the same interpolation method that we used for muscle plastic strains, controlled by the closest joint for each nail.

7.3 Fat Plastic Strains

The above setup produces simulated skin shapes that reasonably match the optically scanned ground truth mesh at each example pose, but some mismatch remains. To further eliminate this discrepancy, we apply pose-varying plastic strains to the fat layer. We stress that fat, unlike muscles, is a passive tissue and has no activation, but we re-use the plastic strains idea nonetheless, this time as a tool to steer the simulation toward optically scanned skin shapes in the example poses. We obtained optical scans using the plastic casting method of [Wang et al. 2019]. We align the optical scans with simulation as follows. Wang et al. [2019] aligned MRIs and optical scans (their Section 5.2), and we use their method. Our simulations operate in the same frame as the neutral MRI scan, and are therefore also aligned. To obtain the plastic strains for each example pose, we formulate the following optimization problem:

$$\underset{\mathbf{s},\,\mathbf{x}}{\arg\min} \quad ||\mathbf{L}\,\mathbf{s}||^2 + \alpha \mathcal{E}_{\text{skin}}(\mathbf{x}), \tag{22}$$

$$\text{subject to:} \quad \mathbf{f}_{\text{e}}\big(\mathbf{F}_{\mathbf{p}}(\mathbf{s}), \mathbf{x}\big) + \mathbf{f}_{\text{c}}(\mathbf{x}) = 0, \tag{23}$$

where x are tet mesh vertex position, and $\mathbf{s}$ contains the plastic strains at all tets. As explained in Section 4.1, $\mathbf{s}$ has 6 entries per

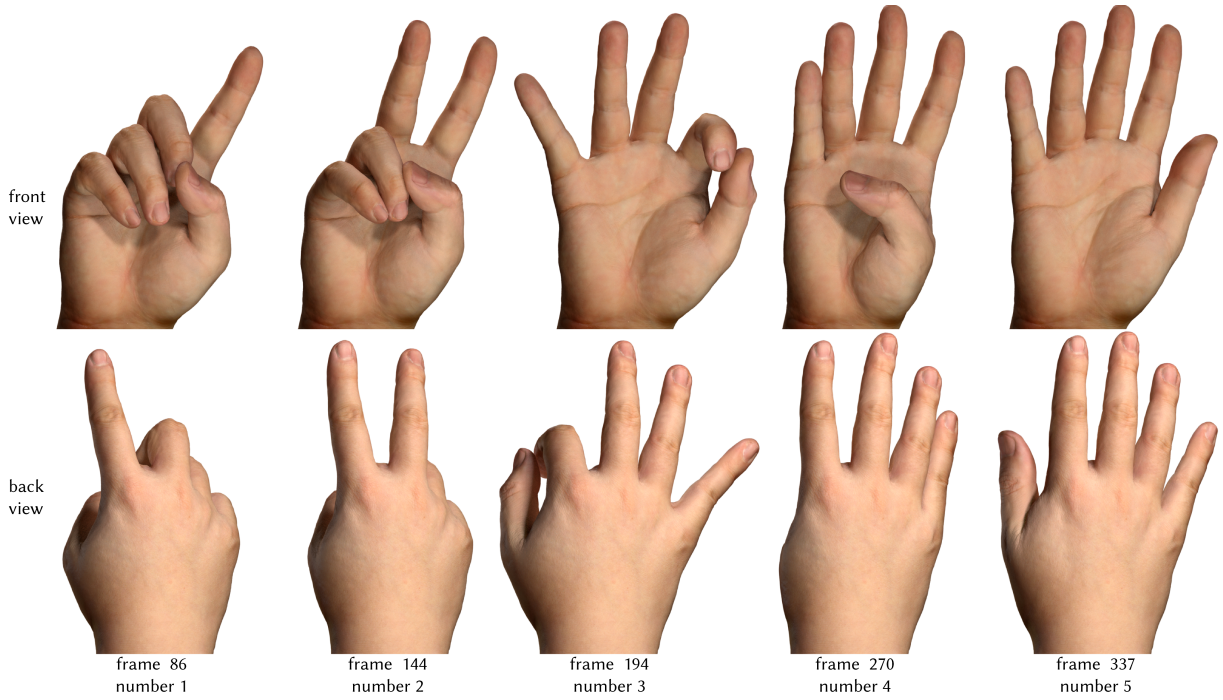


Fig. 20. **Representative frames for motion "Numbers 1-5".** Displayed are the front and back side of the skin. Our model produces realistic skin shapes.

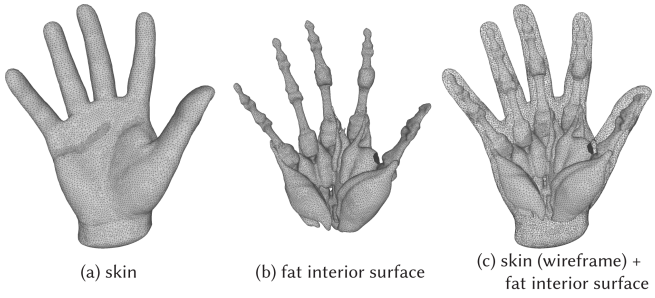


Fig. 21. **The exterior and interior surface of the fat tissue.** (a) The exterior surface of the fat tissue is the skin. (b) The interior surface of the fat tissue is composed of the bone fascia, muscle fascia, and joint ligaments.

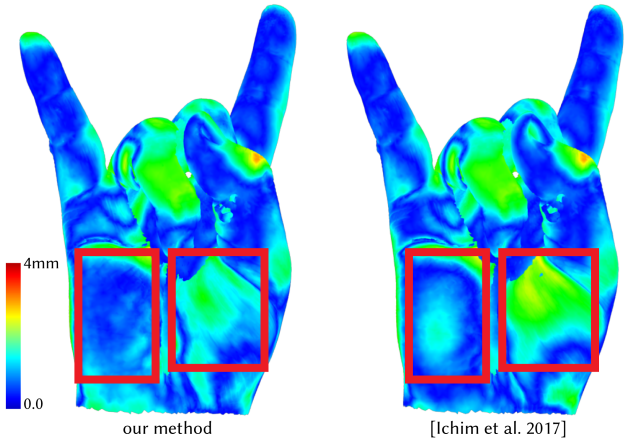


Fig. 22. **Comparison to [Ichim et al. 2017].** We compare the skin of our system and of [Ichim et al. 2017] to the ground truth optical scan at the same pose. This pose was *not* used as an example pose in either system. The color maps show the distance from the ground truth mesh to the simulated skin mesh, for each method. Due to better muscle anatomical modeling, our method has a lower error in the palm area. Namely, Ichim et al. [2017] treats the entire hand as a single soft tissue with spatially varying plastic strains, whereas we model each tissue separately, model sliding, and specialize the pose-space and spatially varying strains to each tissue.

tet. Like with muscles, we use isotropic elastic materials (stable neo-Hookean material [Smith et al. 2018]). Observe that the second equation (Eq. 23) enforces that $\mathbf{x}$ contains the static equilibrium tet mesh vertex positions under the given plastic strains $\mathbf{s}$ and fat constraints. The first term in Equation 22 represents the spatial smoothness of the plastic strains, $\mathcal{E}_{\text{skin}}(\mathbf{x})$ is the ICP energy between the simulated skin and the target ground truth skin, $\mathbf{f}_e(\mathbf{F_p}(\mathbf{s}), \mathbf{x})$ are the fat elastic forces, and $\mathbf{f}_c(\mathbf{x})$ are the forces from all types of constraints applied to the fat. This energy essentially finds per-tet plastic strains that are spatially smooth and so that in the static equilibrium under those strains and the fat constraints, the embedded skin surface mesh matches the optical scan.

We first attempted to optimize Equations 22, 23 directly, similarly to [Ichim et al. 2017], but the method failed to converge and generally did not scale when applied to our problem. To achieve a high accuracy of skin deformation (at or under a millimeter), we use a tetrahedral mesh with ~50 K vertices and ~230 K tetrahedra. On the other hand, Ichim et al. used a volumetric much with only ~8 K vertices and ~35 K tetrahedra, which produces matching errors on the order of a centimeter. We also tried using [Wang et al. 2021], but

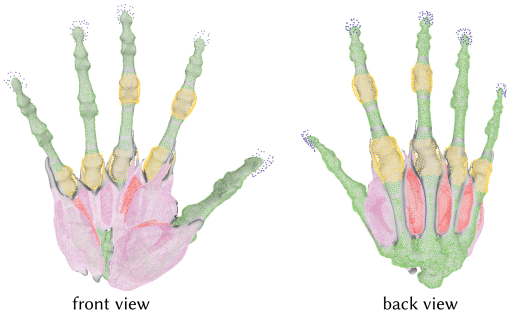


Fig. 23. **The constraints for simulating the fat.** The fat is attached to bone and muscle fascia, ligaments, and tendons. Green dots are fixed constraints to the bone fascia. Yellow dots are contact constraints to the ligaments. Pink dots are sliding constraints against the muscle fascia, and red dots are contacts against the muscle fascia. Purple dots are points for nail rigidity.

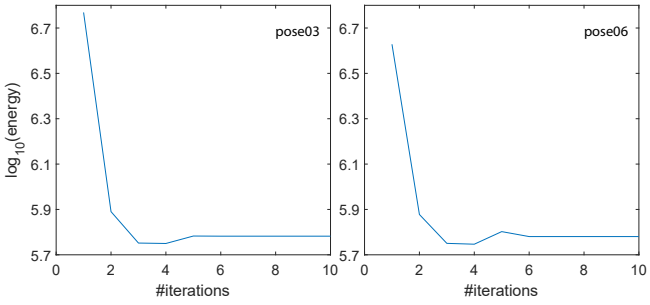


Fig. 24. **The convergence of fat plastic strain optimization.** The plot shows the energy at each iteration during optimization, for poses 03 and 06. We can see that our algorithm quickly decreases the energy.

their method failed to handle the volumetric mesh at this scale, the dense correspondences against the ground truth skin, and complex constraints such as contacts and sliding.

The main difficulty in optimizing Equations 22, 23 arises due to the highly nonlinear and highly dimensional constraints. Our approach to tackle this problem is to generate a good line search direction by approximating the complex relationship between $\mathbf{s}$ and $\mathbf{x}$ (defined using static equilibrium; Eq. 22). Namely, observe that, starting from the current shape as a "rest shape", if one slightly perturbes vertex positions by $\Delta\mathbf{x}$, this causes a small change to the deformation gradient $\mathbf{I} + G\Delta\mathbf{x}$, where G is the gradient operator, and $\mathbf{I}$ is a vector of identity matrices at all tets. Therefore, by our definition of $\mathbf{s}$, we have $\Delta\mathbf{s} = \text{Polar}(\mathbf{I}+G\Delta\mathbf{x})$, where G is the gradient operator, and $\text{Polar}(M)$ computes the symmetric matrix in the polar decomposition of M (done at all the tets separately). Furthermore, in each iteration, $\Delta\mathbf{x}$ is usually small and $\mathbf{s}^\mathbf{i}$ is a constant, i.e., we are performing polar decomposition on a nearly identity matrix, and we can approximate $\text{Polar}(\mathbf{I} + G\Delta\mathbf{x}) = \mathbf{I} + G\Delta\mathbf{x}$. This enables us to linearize the objective function of Equation 22 to (note that $\mathbf{L}\,\mathbf{I} = 0$)

$$\arg\min_{\Delta\mathbf{x}} \quad ||\mathbf{L}(\mathbf{s}^\mathbf{i} + G\Delta\mathbf{x})||^2 + \alpha\mathcal{E}_{\text{skin}}(\mathbf{x}^i + \Delta\mathbf{x}). \tag{24}$$

This is now a quadratic energy in $\Delta\mathbf{x}$, and we can solve for $\Delta\mathbf{x}$ directly by solving a linear system. Once we obtain $\Delta\mathbf{x}$, we compute the deformation gradient of each tet as $\mathbf{I} + G\Delta\mathbf{x}$, and extract the symmetric matrices using polar decomposition. These symmetric matrices form a search direction $\Delta\mathbf{s}$ for updating the plastic strain $\mathbf{s}$ at the i-th iteration. Next, we perform a line search along $\Delta\mathbf{s}$ using the original objective function from Equation 22. During each line search iteration, we first compute plastic strains as $\mathbf{s} = \mathbf{s}^i + \eta\Delta\mathbf{s}$. We then perform forward hand simulation to obtain $\mathbf{x}$, and evaluate the exact energy in each line search iteration. We find η that gives the minimal energy and update $\mathbf{s}^{i+1}, \mathbf{x}^{x+1}$ using this η. Our method converges efficiently, as shown in Figure 24. Using our method, we successfully obtained all plastic strains for all example poses.

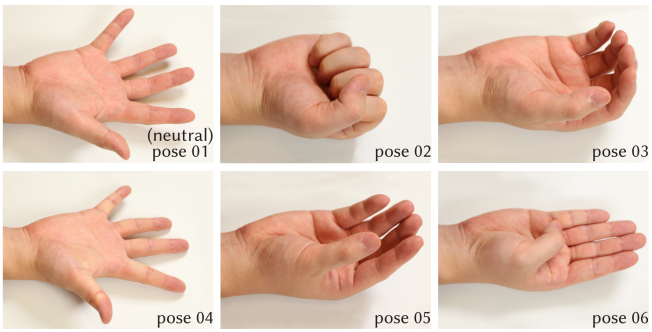


Fig. 25. **Example hand poses used in our work.**

Table 1. **Specifications of meshes of all hand tissues.** We model 23 bones, 17 muscle groups, 10 tendon groups, one bone fascia, one muscle fascia and fat. For tissue types that contain more than one tissue (e.g., 17 muscles), we also list the minimal/maximal/average/sum of the number of vertices and triangles/tetrahedra, respectively. In addition, we show the method used for animating each tissue in the "sim type" column.

tissue type	sim type	# vertices	# triangles/tetrahedra
bones	skinning	788/8,921/3,381/77,771	1.572/17,838/6,759/155,450
muscles	volumetric	457/5,728/1,580/26,857	1,408/27,194/6,566/111,614
tendons	rod	233/385/321/3,210	N/A
bone fascia	cloth	23,829	47,654
muscle fascia	cloth	26,740	53,444
fat	volumetric	48,672	227,314

8 RESULTS

We used an Intel Xeon(R) W-3275 CPU (56 cores @ 2.5 GHz) with 196 GB RAM; max RAM consumption was 30 GB during the simulation. We obtained the MRI data and matching optical scans from [Wang et al. 2019]. The MRI dataset (public at [Wang et al. 2020]) was created on a 3T GE MRI scanner using a PD CUBE (3D fast spin echo) sequence, with a slice thickness of 1 mm and slice spacing of 0.5 mm. Among 12 poses in the dataset, we selected six example poses ("training poses") for use in our work (Figure 25); the selection criterion was to pick example poses that maximize the exertion of the musculoskeletal tissues as much as possible. The remaining six dataset poses are treated as non-example poses ("testing poses"). Note that poses 01, 02, 03, 04, 05, 06 in our paper correspond to poses 01, 02, 03, 04, 05, 12 in the dataset [Wang et al. 2020], respectively.

The specification of the tissue meshes and simulation types for each layer are shown in Table 1. The parameters of our simulation and optimization are shown in Table 2. On average, it takes 47.7

Table 2. **Parameters used in our simulation and optimization.** There are some additional parameters not listed here, e.g., muscle fitting γ, and the sliding and fixed constraints stiffnesses. These parameters are either determined interactively, or painted in a spatially-varying manner.

parameter	value	parameter	value
bone/muscle fascia E	100Pa	bone/muscle fascia ν	0.48
bone fascia UV scale	35%	muscle fascia UV scale	40%
muscle E	6,000Pa	muscle ν	0.48
fat E	1,000Pa	fat ν	0.48
tendon c_1, c_2, c_3, c_4	0.5,1,1,50	muscle mass density	1,000 kg/m^3
fat mass density	500 kg/m^3	timestep	0.01s
tendon #sub-timesteps	10	muscle #sub-timesteps	1
fat #sub-timesteps	1-10	fat fitting α	1e10
muscle fitting α	1e9		

Table 3. **The time cost of each sequence.** We successfully simulated five sequences. In the "type" column, we show how the input animation was created, whereby "keyframe" refers to keyframe animation, "LeapMotion" means that the animation was created by tracking the hand of a live subject using LeapMotion [LeapMotion 2017], and "MediaPipe" means that the animation was created by tracking the hand of a live subject using Google MediaPipe [Lugaresi et al. 2019]. Columns n_f and n_s give the number of frames and the number of simulation timesteps, respectively. Column t shows the total time cost for each sequence.

sequence name	type	n_f	n_s	t[hr]
"Close the fist"	keyframe	132	1067	14.1
"Opposition of the thumb"	keyframe	996	3381	44.8
"Performance animation"	LeapMotion	653	3525	46.7
"Numbers 1-5"	MediaPipe	360	3233	42.8
"American Sign Language"	keyframe	732	4822	63.8

seconds to complete one step on the entire hand simulation (all layers), where the bone rig takes a negligible amount of time, the bone fascia takes 4.5 seconds (9.50%), tendons takes 6.1 seconds (1.27%), muscles takes 11.9 seconds (24.98%), muscle fascia takes 6.6 seconds (13.94%), and the fat takes 24.0 seconds (50.30%). We successfully executed our method on five challenging hand motion sequences; three of which were keyframe-animated, and two were acquired using hand tracking. All sequences use the same simulation settings, attachments, contact parameters, PSD corrections settings, etc. In other words, all simulation layers are completely identical for all the 5 motions; the only difference is different input joint animations. This fact demonstrates that our simulation technology is robust. Performance statistics are shown in Table 3.

Our model generates realistic skin deformation, as shown in Figure 20. To evaluate the accuracy of our model, we extensively compared our simulation results with the ground truth quantitatively and qualitatively, as shown in the remainder of this Section. The extracted meshes of the muscles (Section 4.4) in non-neutral example poses are presented in Figure 26. We compare the simulated muscles in example poses with the ground truth muscle meshes extracted from MRI in those same poses (Figure 27). We do so in three example poses that are the most extreme among all six example poses. It is evident that the simulation results closely overlap with the ground truth meshes. Still, small errors remain. These errors are expected as our model contains inter-muscle sliding forces and contact forces, for which no precise ground truth exists. Nonetheless, our method

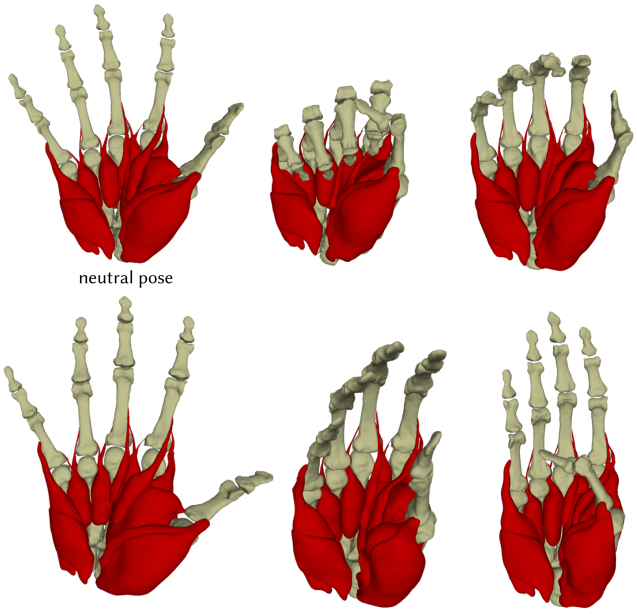


Fig. 26. **Extracted muscle meshes in example poses,** using the process described in Section 4.4.

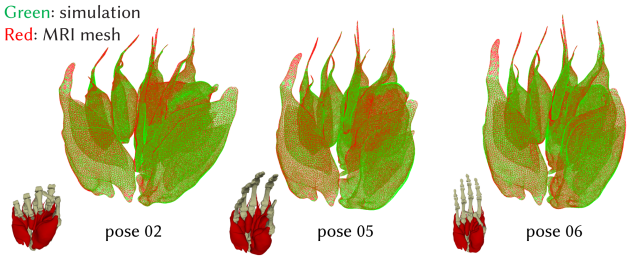


Fig. 27. **Comparisons between muscle simulation results and the ground truth meshes in example poses.** We simulated hand muscles using our model to reach a few example poses that we believe are the most extreme poses among all six example poses. Simulation results (green wireframe) closely overlap with the ground truth meshes (red wireframe).

ensures that these forces are minimized at example poses, because contact resolving cleans up the example surface meshes.

We also compare our simulated tendons to the ground truth. As shown in Figure 28, the simulation results closely match the ground truth tendon meshes. It is expected that there will be some small error, as we only control the hooks. We did not attempt to match the orientations of the tendon rod vertices (i.e., normals) in the example poses. We also evaluated the benefits of including tendons into the overall hand simulation. Tendons are not serving only as tissues to which to attach the muscles and fat, but they also produce a more correct skin output shape, e.g., in the knuckles area (Figure 31, (a)).

Our hand model is superior to previous work and better matches the ground truth. We measured the accuracy improvement between a method that simulates all soft tissues using a single tet mesh [Wang

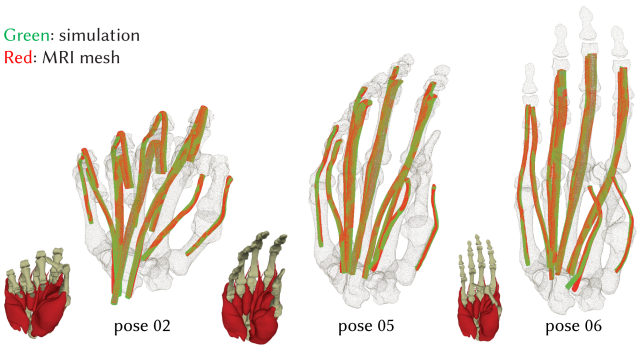


Fig. 28. **Comparisons between tendon simulation results and the ground truth meshes in example poses.** We simulated hand tendons using our model to reach a few example poses (same as in the muscle experiment). Simulation results (green wireframe) closely overlap with the ground truth meshes (red wireframe). Note that the minor mismatch at the bottom of some tendons is because we do not have MRI data in that region.

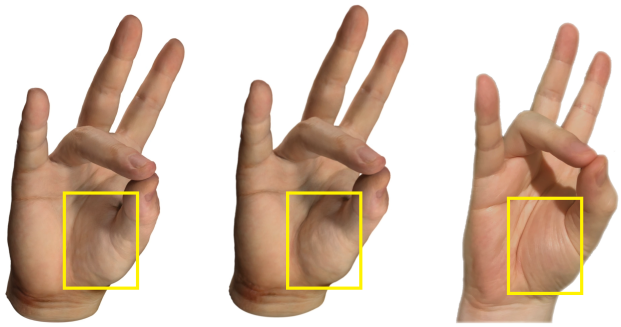


Fig. 29. **Visual comparisons to [Wang et al. 2019].** Left: [Wang et al. 2019]. Middle: our method. Right: photograph of the same subject. The muscle at the base of the thumb is too flat in [Wang et al. 2019]. Due to modeling of pose-varying muscle activations via plastic strains, the muscle bulges much more in our method, which is closer to real-world behavior. Note that this is a **non-example (unseen)** pose for our method.

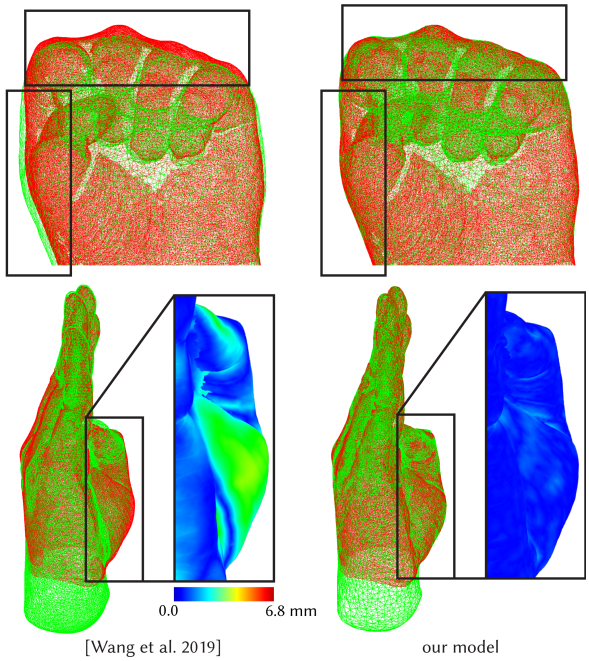


Fig. 30. **Visual comparisons to Wang et al. [Wang et al. 2019] in example poses.** The ground truth mesh captured using a optical scanner is depicted in red wireframe, whereas the simulation results are shown in green wireframe. The palm and knuckle joint areas (highlighted by rectangles) bulge more correctly in our simulation model compared to the single-layer soft tissue model. The color-mapped inset shows the error against the optical scan quantitatively: our model produces significantly lower error in areas of significant muscle activity (e.g., muscle below the thumb).

et al. 2019] vs our method. For each vertex on the ground truth optically scanned surface mesh in each example pose, we computed the closest distance to the simulation mesh, for either method. Table 4 presents average distance, median distance, and max distance for each example pose. We can see that our separate-mesh method reduces the error to 40.2% for average distance, 48.9% for median distance, and 63.4% for maximal distance. For all 12 poses, our results are better in average, median and max distance. In Table 4, we also demonstrate that applying the fat plastic strains improves the accuracy of our model, i.e., our method without fat plastic strains is clearly worse than with plastic strains (ablation study).

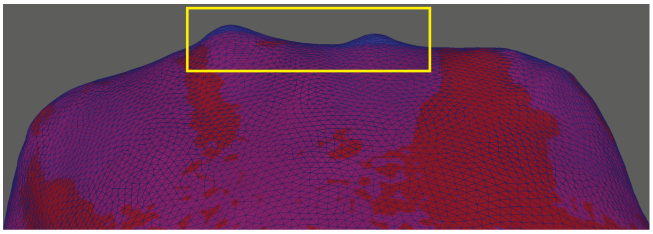
We also compared our simulation with [Wang et al. 2019] visually (qualitatively), on the motion sequence used in their work. We observed a clear improvement in the palm area (Figures 29, 30). Compared to [Wang et al. 2019], our muscles are modeled using plastic strains, controlled by bone transformations, which means that we "activate" the muscles based on the hand pose; doing so improves the simulation results. Because we model fat plastic strains, tendons and ligaments, whereas previously these tissues were not modeled, we are able to match the ground truth well (Figure 30), including better muscle bulges and more correct hand silhouettes.

We also compared our skin deformations with the method of [Ichim et al. 2017]. Note that Ichim et al. [2017] was designed for facial simulation; they did not attempt hands; and their method was not designed for matching internal anatomy to medical images. They modeled the entire face using a single tetrahedral mesh. When applying their method to a hand, we model the entire hand as a single soft tissue with spatially varying material properties and plastic strains in each example pose. One advantage of our method over [Ichim et al. 2017] is that we produce an animation of the internal anatomy. In addition, our method produces less skin position error in the palm region of an unseen pose, compared to the ground truth (Figure 22). Moreover, our model generates a sharper (more correct) silhouette around the knuckles (Figure 31(b)).

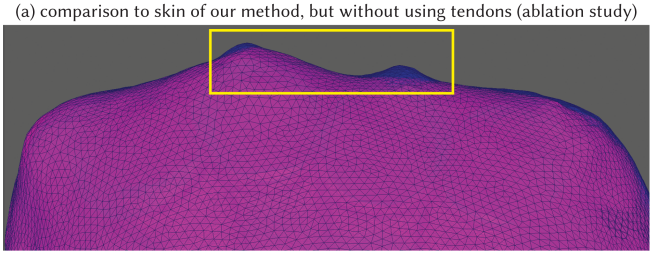
We also evaluated how well the muscle and skin shapes are reproduced in both example poses and non-example poses, against the MRI scan. Figure 32 demonstrates that our simulation result closely matches the MRI scan in example poses. Our model also produces reasonable results in non-example (unseen) poses (Figure 33).

Table 4. **Comparisons between simulated and ground truth skins.** Model m_0 refers to our proposed complete model, model m_1 denotes our model without using plastic strains in the fat layer (ablation study), and model m_2 simulates all soft tissues using a single mesh (proposed in [Wang et al. 2019]). Column "%" denotes the ratio between the value in m_0 and the value in m_2. The first five rows (plus the rest shape) are example poses used for our model, whereas the last six rows are non-example (unseen) poses. All distances are reported in **millimeters**.

pose	average				median				max			
	m_0	m_1	m_2	%	m_0	m_1	m_2	%	m_0	m_1	m_2	%
2	0.23	0.94	1.35	17.0%	0.13	0.76	1.22	10.7%	2.42	4.81	4.70	51.4%
3	0.14	0.72	1.19	11.8%	0.08	0.58	0.88	9.1%	2.72	3.89	6.34	42.9%
4	0.11	0.93	1.31	8.4 %	0.07	0.77	0.86	8.1%	2.59	3.34	7.09	36.5%
5	0.22	1.02	1.21	18.2%	0.13	0.83	1.00	13.0%	3.66	4.60	5.61	65.2%
6	0.15	0.72	1.15	13.0%	0.09	0.59	0.90	10.0%	2.60	4.89	7.86	33.1%
7	0.54	0.63	0.84	64.3%	0.43	0.50	0.66	65.2%	2.70	2.85	2.70	100.0%
8	0.59	0.89	1.02	57.8%	0.44	0.69	0.80	55.0%	4.10	5.20	4.67	87.8%
9	0.79	0.97	1.08	73.2%	0.66	0.85	0.92	71.7%	3.79	4.15	4.80	79.0%
10	0.81	1.09	1.43	56.6%	0.65	0.80	1.08	60.2%	3.42	4.89	6.30	54.3%
11	0.70	0.88	1.24	56.5%	0.46	0.62	0.87	52.9%	4.54	5.63	7.69	59.0%
12	0.89	1.01	1.05	84.8%	0.72	0.86	0.78	92.3%	4.98	4.51	5.46	91.2%
All	0.47	0.89	1.17	40.2%	0.43	0.76	0.88	48.9%	4.98	5.63	7.86	63.4%



(a) comparison to skin of our method, but without using tendons (ablation study)



(b) comparison to [Ichim et al. 2017]

Fig. 31. **Tendon modeling is necessary to produce good knuckles.** In (a), we evaluate how the presence of tendons affects the skin output shape. In the real hand, knuckles are pronounced visual features, due to the underlying tendon which lifts the skin in the knuckle area. As highlighted in the yellow rectangle in (a), when we run our method with tendons (blue wireframe), knuckles correctly appear. When we omit the tendons from our method, knuckles are not properly resolved and the skin silhouette is flat (purple solid). (b) We also compare our method with [Ichim et al. 2017]. Our model (blue wireframe) successfully captures the sharp silhouette of the knuckles, whereas [Ichim et al. 2017] (purple solid) produces a visibly flatter surface, due to the missing tendons.

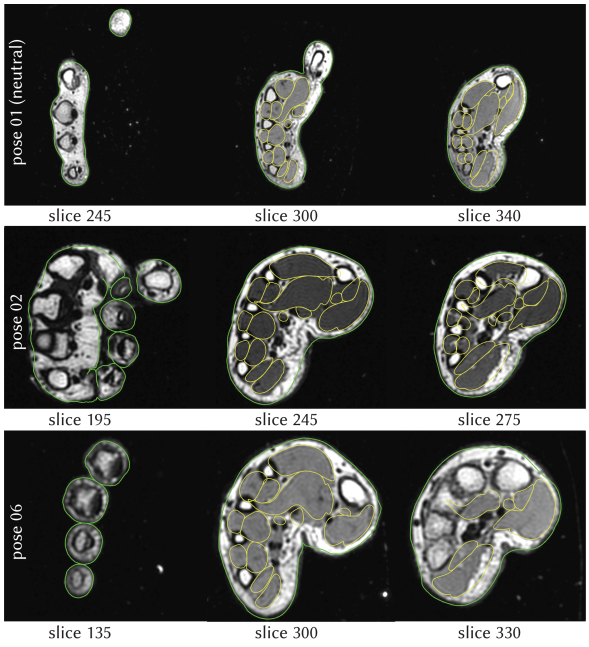


Fig. 32. **Visual comparisons to MRI slices in example poses**. We compare our simulated skin and muscle surfaces with the MRI images in example poses 1, 2 and 6. Pose 1 is the neutral pose. Pose 2 (fist) and 6 (thumb moving to the extreme opposite palm location) are extreme example poses. We intersect our surface meshes with the MRI slices. The intersections between skin and MRI slices are shown in green, and between muscles and MRI slices in yellow. Observe that the contour lines very closely match the anatomical structures seen in MRI, both for the skin and muscles.

Finally, we visualized the MRI data using volume rendering [Hadwiger et al. 2006]. As shown in Figure 2, two types of transfer functions were used to emphasize the muscles and the fat tissue, respectively. In the "muscle-emphasized" transfer function, the fat tissue was assigned low opacity and is therefore partially hidden. This visualization mode shows the internal organs very well, such as muscles and bones. In the "fat-emphasized" transfer function, every tissue except the fat was given a constant color, whereas the fat

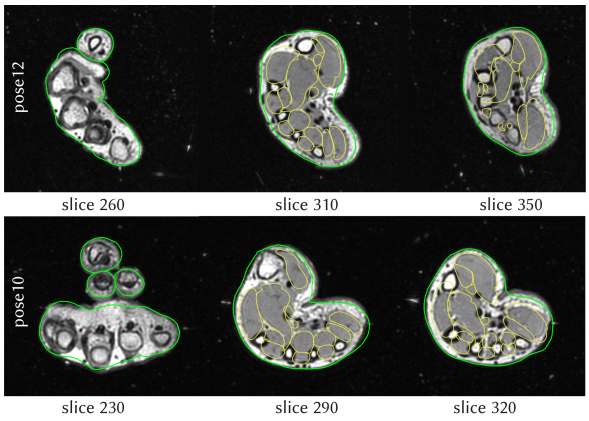


Fig. 33. **Visual comparisons to MRI slices in non-example poses**. We compare our simulated skin and muscles with the MRI images in two non-example poses. The intersections between skin and the slices are shown in green, and between muscles and the slices in yellow. We can observe a reasonably close match to the MRI data for both the skin and the muscles.

color was computed using the MRI value. As a result, we can see the nerves and veins clearly (see, e.g., the back side of the hand), and the nerves and veins nicely animate in our animation sequences.

9 CONCLUSION

We gave a method to simulate complete musculoskeletal hand anatomy in a manner that matches medical images (MRI) in scanned example poses, and that generalizes smoothly to the entire range of motion of the hand, as demonstrated on five complex hand motion sequences. We are not aware of any work that has demonstrated such capabilities before. Starting from MRI scans of the same subject's hand in multiple poses, we build a data-driven simulation system that can simulate the hand's anatomy to any pose in the hand's range of motion. This is achieved through layered FEM simulation whereby per-organ plastic strains are added to "guide" the simulation to the acquired medical images in the example poses. We present new methods to simulate hand muscles, tendons and fat, and demonstrate that this not only enables volumetric rendering of hand anatomy across the range of motion, but also improves the hand's surface shape. Although simulation times are long, they are feasible on modern hardware, and especially so with public infrastructure such as AWS, Google Cloud, etc. In the future, our method could run on the cloud, systematically exploring the entire range of motion of the hand. In this way, one could produce high-quality training data for data-driven methods (e.g., SMPL [Loper et al. 2015], or deep neural networks), which could then compactify the data and greatly increase runtime speeds. Our work processed a single subject; for Metaverse applications, more subjects will be needed, and they will need to be properly selected to capture humankind's diversity. We only modeled concentric isotonic muscle contraction, and leave other muscle interactions, such as those involving heavy grasping or lifting (isometric), or lengthening/overpowering the muscles (eccentric isotonic) for future work. In particular, (substantial) external contact will likely invalidate muscle activation. Our simulation involves one-way coupling between the layers because it is otherwise not feasible to solve the (coupled) optimization problems to make the simulation match the medical images. Two of the hand muscles (lumbricals III and IV) were too small to reliably resolve on the MRI in multiple poses, and we simulate them directly, i.e., based on their shape in the neutral shape and pose-varying bone attachments without using MRI. Our tendons are only modeled in the region where they are visible on the MRI scan. In general, MRI resolution is an important limiting factor, as many hand structures are at the limit of what the MRI scanner can resolve. However, by inferring the structure from multiple scans and when combined with physically based simulation, we were able to reasonably overcome these challenges and produce, to the best of our knowledge, the world's first simulation-ready volumetric hand musculoskeletal system that both matches and generalizes real medical images.

ACKNOWLEDGMENTS

This research was sponsored in part by NSF (IIS-1911224), USC Annenberg Fellowship to Mianlun Zheng and Bohan Wang, Bosch Research and Adobe Research. The authors would like to thank Shreya Bhaumik for her help with MRI scan segmentation.

REFERENCES

3dMD. 2022. 3dMDHands. https://3dmd.com/.
R. Abdrashitov, S. Bang, D. Levin, K. Singh, and A. Jacobson. 2021. Interactive Modelling of Volumetric Musculoskeletal Anatomy. *ACM Transactions on Graphics* 40, 4 (2021).
B. Amberg, S. Romdhani, and T. Vetter. 2007. Optimal Step Nonrigid ICP Algorithms for Surface Registration. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*.
B. Angles, D. Rebain, M. Macklin, B. Wyvill, L. Barthe, JP Lewis, J. Von Der Pahlen, S. Izadi, J. Valentin, S. Bouaziz, and Tagliasacchi A. 2019. Viper: Volume invariant position-based elastic rods. *Proc. of ACM on Computer Graphics and Interactive Techniques* 2, 2 (2019), 1–26.
Artelys. 2019. Knitro. https://www.artelys.com/solvers/knitro/.
M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun. 2008. Discrete Elastic Rods. *ACM Transactions on Graphics (SIGGRAPH)* 27, 3 (2008), 63:1–63:12.
S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Trans. Graph.* 33, 4, Article 154 (2014), 154:1–154:11 pages.
C. Erolin. 2019. Hand Anatomy. University of Dundee, Centre for Anatomy and Human Identification. https://sketchfab.com/anatomy_dundee/collections/hand-anatomy.
S. Capell, M. Burkhart, B. Curless, T. Duchamp, and Z. Popović. 2005. Physically Based Rigging for Deformable Characters. In *Symp. on Computer Animation (SCA)*. 301–310.
H. Chen, C. Chen, T. Yang, L. Kuo, IM Jou, F. Su, and Y. Sun. 2011. Model-based segmentation of flexor tendons from magnetic resonance images of finger joints. In *Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society*. 8009–8012.
K. Comley and N. A. Fleck. 2010. A micromechanical model for the Young's modulus of adipose tissue. *Int. J. of Solids and Structures* 47, 21 (2010), 2982–2990.
A. H. Dicko, T. Liu, B. Gilles, L. Kavan, F. Faure, O. Palombi, and M.P. Cani. 2013. Anatomy Transfer. *ACM Trans. on Graphics (SIGGRAPH)* 32, 6 (2013), 188:1–188:8.
A. Dogadov, M. Alamir, C. Serviere, and F. Quaine. 2017. The biomechanical model of the long finger extensor mechanism and its parametric identification. *J. of Biomechanics* 58 (2017), 232–236.
A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, J. Buatti, S. Aylward, J. V. Miller, S. Pieper, and R. Kikinis. 2012. 3D Slicer as an image computing platform for the Quantitative Imaging Network. *Magnetic Resonance Imaging* 30, 9 (2012), 1323–1341.
K. S. Fok and S. M. Chou. 2010. Development of a finger biomechanical model and its considerations. *J. of Biomechanics* 43, 4 (2010), 701–713.
A. K. Garland, D. S. Shah, and A. E. Kedgley. 2018. Wrist tendon moment arms: Quantification by imaging and experimental techniques. *J. of Biomechanics* 68 (2018), 136–140.
C. Garre, F. Hernández, A. Gracia, and M. A. Otaduy. 2011. Interactive simulation of a deformable hand for haptic rendering. In *IEEE World Haptics Conf.* 239–244.
M. Hadwiger, J. M. Kniss, C. Rezk-salama, D. Weiskopf, and K. Engel. 2006. *Real-Time Volume Graphics*. A. K. Peters, Ltd.
A.E. Ichim, P. Kadlecek, L. Kavan, and M. Pauly. 2017. Phace: Physics-based Face Modeling and Animation. *ACM Trans. on Graphics (SIGGRAPH 2017)* 36, 4 (2017).

A. Jacobson, Z. Deng, L. Kavan, and J. P. Lewis. 2014. Skinning: Real-time Shape Deformation. In *ACM SIGGRAPH 2014 Courses*.
P. Kadlecek, A.-E. Ichim, T. Liu, J. Krivanek, and L. Kavan. 2016. Reconstructing Personalized Anatomical Models for Physics-based Body Animation. *ACM Trans. Graph.* 35, 6 (2016).
A.I. Kapandji. 2009. *The physiology of the joints, 6th Edition, Vol. 1: The Upper Limb.* Elsevier Exclusive.
L. Kavan, S. Collins, J. Zara, and C. O'Sullivan. 2008. Geometric Skinning with Approximate Dual Quaternion Blending. *ACM Trans. on Graphics* 27, 4 (2008).
M. Keller, S. Zuffi, M. J. Black, and S. Pujades. 2022. OSSO: Obtaining Skeletal Shape from Outside. In *Conf. on Computer Vision and Pattern Recognition (CVPR).* 20492–20501.
J. Kim and N. S. Pollard. 2011. Fast simulation of skeleton-driven deformable body characters. *ACM Trans. on Graphics (TOG)* 30, 5 (2011), 121.
P. G. Kry, D. L. James, and D. K. Pai. 2002. EigenSkin: Real Time Large Deformation Character Skinning in Hardware. In *In Symp. on Computer Animation (SCA).*
T. Kugelstadt and E. Schömer. 2016. Position and Orientation Based Cosserat Rods. In *Symp. on Computer Animation (SCA).*
C. Kuok, T. Yang, B. Tsai, I. Jou, M. Horng, F. Su, Y. Sun, et al. 2020. Segmentation of finger tendon and synovial sheath in ultrasound image using deep convolutional neural network. *Biomedical engineering online* 19, 1 (2020), 1–25.
T. Kurihara and N. Miyata. 2004. Modeling deformable human hands from medical images. In *Symp. on Computer Animation (SCA).* 355–363.
Leal, Allan. 2018. autodiff. https://github.com/autodiff/autodiff/.
LeapMotion. 2017. https://www.leapmotion.com.
S. Lee, R. Yu, J. Park, M. Aanjaneya, E. Sifakis, and J. Lee. 2018. Dexterous manipulation and control with volumetric muscles. *ACM Transactions on Graphics (SIGGRAPH 2018)* 37, 4 (2018), 57:1–57:13.
S. H. Lee, E. Sifakis, and D. Terzopoulos. 2009. Comprehensive Biomechanical Modeling and Simulation of the Upper Body. *ACM Trans. on Graphics* 28, 4 (2009), 99:1–99:17.
J. P. Lewis, Matt Cordner, and Nickson Fong. 2000. Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. In *Proc. of ACM SIGGRAPH 2000.* 165–172.
Y. Li, M. Wu, Y. Zhang, L. Xu, and J. Yu. 2021. PIANO: A Parametric Hand Bone Model from Magnetic Resonance Imaging. *arXiv preprint arXiv:2106.10893* (2021).
Y. Li, L. Zhang, Z. Qiu, Y. Jiang, N. Li, Y. Ma, Y. Zhang, L. Xu, and J. Yu. 2022. NIMBLE: a non-rigid hand model with bones and muscles. *ACM Transactions on Graphics (SIGGRAPH 2022)* 41, 4 (2022), 1–16.
L. Liu, K. Yin, B. Wang, and B. Guo. 2013. Simulation and control of skeleton-driven soft body characters. *ACM Trans. on Graphics (SIGGRAPH Asia)* 32, 6 (2013), 215.
M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. on Graphics (SIGGRAPH Asia 2015)* 34, 6 (2015), 248:1–248:16.
C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C. Chang, M. G. Yong, J. Lee, W. Chang, W. Hua, M. Georg, and M Grundmann. 2019. MediaPipe: A Framework for Building Perception Pipelines.
A. McAdams, Y. Zhu, A. Selle, M. Empey, R. Tamstorf, J. Teran, and E. Sifakis. 2011. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. on Graphics (SIGGRAPH 2011)* 30, 4 (2011).
N. Miyata, M. Kouch, M. Mochimaru, and T. Kurihara. 2005. Finger joint kinematics from MR images. In *IEEE Int. Conf. on Intelligent Robots and Systems.* 2750–2755.
V. Modi, L. Fulton, A. Jacobson, S. Sueda, and D. Levin. 2021. Emu: Efficient muscle simulation in deformation space. In *Computer Graphics Forum*, Vol. 40. 234–248.
M. Müller, B. Heidelberger, M. Teschner, and M. Gross. 2005. Meshless Deformations Based on Shape Matching. In *Proc. of ACM SIGGRAPH 2005.* 471–478.
N. Qian, J. Wang, F. Mueller, F. Bernard, V. Golyanik, and C. Theobalt. 2020. HTML: A Parametric Hand Texture Model for 3D Hand Reconstruction and Personalization. In *Proc. of the European Conf. on Computer Vision (ECCV).* Springer.
T. Rhee, J.P. Lewis, and U. Neumann. 2006. Real-Time Weighted Pose-Space Deformation on the GPU. In *Proc. of Eurographics*, Vol. 25.
J. Romero, D. Tzionas, and M. J. Black. 2017. Embodied Hands: Modeling and Capturing Hands and Bodies Together. *ACM Trans. on Graphics (SIGGRAPH Asia 2017)* 36, 6 (2017), 245:1–245:17.
V. Roussellet, N. A. Rumman, F. Canezin, N. Mellado, K. Kavan, and L. Barthe. 2018. Dynamic implicit muscles for character skinning. *Computers & Graphics* 77 (2018), 227–239.
A. Rusu. 2011. *Segmentation of bone structures in Magnetic Resonance Images for human hand skeletal kinematics modelling.* Master's thesis. German Aerospace Center.
P. Sachdeva, S. Sueda, S. Bradley, M. Fain, and D. K. Pai. 2015. Biomechanical Simulation and Control of Hands and Tendinous Systems. *ACM Trans. Graph.* 34, 4 (2015), 42:1–42:10.
S. Saito, Z. Zhou, and L. Kavan. 2015. Computational Bodybuilding: Anatomically-based Modeling of Human Bodies. *ACM Trans. on Graphics (SIGGRAPH 2015)* 34, 4 (2015).
E. Sifakis, I. Neverov, and R. Fedkiw. 2005. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Trans. on Graphics (SIGGRAPH 2005)* 24, 3 (2005), 417–425.
B. Smith, F. De Goes, and T. Kim. 2018. Stable Neo-Hookean Flesh Simulation. *ACM Trans. Graph.* 37, 2 (2018), 12:1–12:15.
G. Stillfried. 2015. *Kinematic modelling of the human hand for robotics.* Ph.D. Dissertation. Technische Universität München.
S. Sueda, A. Kaufman, and D. K. Pai. 2008. Musculotendon Simulation for Hand Animation. *ACM Trans. Graph. (Proc. SIGGRAPH)* 27, 3 (2008).
Tissue. 2013. Weta Digital: Tissue Muscle and Fat Simulation System.
R. Vaillant, G. Guennebaud, L. Barthe, B. Wyvill, and M.P. Cani. 2014. Robust Iso-surface Tracking for Interactive Character Skinning. *ACM Trans. on Graphics (SIGGRAPH Asia 2014)* 33, 6 (2014), 189:1–137:11.
P. van der Smagt and G. Stillfried. 2008. Using MRI data to compute a hand kinematic model. In *Conf. on Motion and Vibration Control (MOVIC).*
P. Volino, N. Magnenat-Thalmann, and F. Faure. 2009. A Simple Approach to Nonlinear Tensile Stiffness for Accurate Cloth Simulation. *ACM Trans. Graph.* 28, 4, Article 105 (2009), 16 pages.
B. Wang, G. Matcuk, and J. Barbič. 2019. Hand Modeling and Simulation Using Stabilized Magnetic Resonance Imaging. *ACM Trans. on Graphics (SIGGRAPH 2019)* 38, 4 (2019).
B. Wang, G. Matcuk, and J. Barbič. 2020. Hand MRI dataset. http://www.jernejbarbic.com/hand-mri-dataset.
B. Wang, G. Matcuk, and J. Barbič. 2021. Modeling of Personalized Anatomy using Plastic Strains. *ACM Trans. on Graphics (TOG)* 40, 2 (2021).
D. L. Wilson, Q. Zhu, J. L. Duerk, J. M. Mansour, K. Kilgore, and P. E. Crago. 1999. Estimation of tendon moment arms from three-dimensional magnetic resonance images. *Annals of Biomedical Engineering* 27, 2 (1999), 247–256.
Wrap3. 2018. Nonlinear Iterative Closest Point mesh registration software. https://www.russian3dscanner.com.
FE Zajac. 1989. Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. *Critical reviews in biomedical engineering* 17, 4 (1989), 359–411.
Zygote. 2016. Zygote body. http://www.zygotebody.com.

A GRADIENTS AND HESSIANS

For muscle optimization in Equation 3, the quantity $\mathcal{E}_{\text{Dense}}(\mathbf{x_i})$ is simply the squared distance energy. We use block-coordinate descent. The gradient and Hessian of the pose-space smoothness energy $\mathcal{E}_{ps}(\mathbf{x_i}) = \frac{1}{2}||\mathbf{L}\,\mathbf{S}(\mathbf{x_i})||^2$ are

$$\nabla_{\mathbf{x_i}}\mathcal{E}_{ps}(\mathbf{x_i}) = (\mathbf{L}^T\mathbf{L}\mathbf{S})^T\frac{\partial \mathbf{S}}{\partial \mathbf{F}}\mathbf{G}, \tag{25}$$

$$\nabla^2_{\mathbf{x_i}}E_{ps}(\mathbf{x_i}) = \mathbf{G}^T(\frac{\partial \mathbf{S}}{\partial \mathbf{F}}^T\mathbf{L}^T\mathbf{L}\frac{\partial \mathbf{S}}{\partial \mathbf{F}} + \mathbf{S}^T\mathbf{L}^T\mathbf{L}\frac{\partial^2 \mathbf{S}}{\partial \mathbf{F}^2})\mathbf{G}, \tag{26}$$

where $\mathbf{F} = \mathbf{G}(\mathbf{x}_i - \mathbf{X}_i)$ is a vector concatenating deformation gradients of all elements, and $\mathbf{G}$ is the linear gradient operator matrix. The gradient and Hessian of $\mathbf{S}$ with respect to $\mathbf{F}$ can be computed as specified in [Wang et al. 2021]. We evaluate all terms using the sparse matrix form, except the term $\mathbf{S}^T\mathbf{L}^T\mathbf{L}\,\partial^2\mathbf{S}/\partial\mathbf{F}^2$. This term is efficiently evaluated by first calculating $\mathbf{S}^T\mathbf{L}^T\mathbf{L}$, which results in a vector. Then, the product of this vector and the 3D tensor $\partial^2\mathbf{S}/\partial\mathbf{F}^2$ is evaluated at each element in parallel (due to independence).

For tendon optimization, we use "autodiff" [Leal, Allan 2018] to calculate the gradient and Hessian of $E_{\text{sliding}}(x,t)$ (Equation 10) and $E_{\text{ICP}}(x,n)$ (Equation 18). Autodiff is a C++17 library that enables automatic efficient computation of derivatives. The gradients and Hessians of $E_{\text{pulling}}(x)$ and $E_{\text{twist-bend}}(n)$ are

$$\nabla_x E_{\text{pulling}}(x) = -f, \qquad \nabla^2_x E_{\text{pulling}}(x) = 0, \tag{27}$$

$$\nabla_{n_i,n_{i+1}} E_{\text{twist-bend}}(n) = (-n^T_{i+1}, -n^T_i)^T, \tag{28}$$

$$\nabla^2_{n_i,n_{i+1}} E_{\text{twist-bend}}(n) = \begin{bmatrix} & -\mathbb{I}_3 \\ -\mathbb{I}_3 & \end{bmatrix}. \tag{29}$$

For the fat optimization (Equation 24, after linearization of Equation 22), quantity $\mathcal{E}_{\text{skin}}(\mathbf{x}^i + \Delta\mathbf{x})$ is a squared distance energy and $||\mathbf{L}(\mathbf{s^i} + G\Delta\mathbf{x})||^2$ is a quadratic energy.