Anatomically Based Human Hand Modeling and Simulation

by

Bohan Wang

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Computer Science)

December 2021

I dedicate this thesis to my family for love and encouragement, my PhD advisor Prof. Jernej Barbic for grateful support, inspiration, and guidance, my lab mates for help and discussions, and committee members for reading and improving the thesis.

Table of Contents

List of Tables

List of Figures

Abstract

Human hands and their modeling and animation are of paramount importance in many applications, such as computer games, films, ergonomic design, tracking, and medical treatment. Unfortunately, complex hand biomechanics today is not modeled, measured, or resolved in any qualitative manner. While there are several computational models of hand kinematics, few have attempted to model internal structures, let alone produce anatomically precise models that match real-world data.

This thesis demonstrates how to acquire complete human hand anatomy in multiple poses using magnetic resonance imaging (MRI), how to extract each different organ—such as bones and muscles from MRI, and how to build an accurate animation-ready "rig" for the entire hand. Acquiring human hand anatomy in multiple poses was previously difficult because MRI scans must take a long duration for high-precision results (over 10 minutes) and because humans cannot hold their hands perfectly still in non-trivial and badly supported poses. Thus, we invent a manufacturing process whereby lifecasting materials commonly employed in the film special effects industry are used to generate hand molds that are personalized to the subject and to each pose. These molds are both ergonomic and encasing, and they also stabilize the hand during scanning. We also demonstrate how to efficiently segment the MRI scans into individual bone meshes in all poses, and how to correspond each bone's mesh to the same mesh connectivity across all poses. Next, we interpolate and extrapolate the MRI-acquired bone meshes to the entire range of motion of the hand, thereby producing an accurate data-driven animation-ready rig for bone meshes. We also demonstrate how to acquire not just bone geometry (using MRI) in each pose, but also a matching highly accurate surface geometry (using optical scanners) in each pose, modeling skin pores and wrinkles. We also give a soft tissue Finite Element Method simulation "rig", consisting of novel

tet meshing for stability at the joints, spatially varying geometric and material detail, and quality constraints to the acquired skeleton kinematic rig.

To further improve the hand model in both qualitative and quantitative aspects, we also separately extract and model other organs such as muscles and tendons. Organs like muscles are often difficult to extract from MRI, because the boundaries between muscles and other organs are often blurry. To extract them, we propose a method for modeling solid objects undergoing large spatially varying and/or anisotropic strains given sparse observations (blurry boundary). Our novel shape deformation method uses plastic strains and the Finite Element Method to successfully model shapes undergoing large and/or anisotropic strains, specified by sparse point constraints on the boundary of the object. We extensively compare our method to standard second-order shape deformation methods, variational methods, and surface-based methods and demonstrate that our method avoids the spikiness, wiggliness, and other artefacts of previous methods. Further, we demonstrate how to perform such shape deformation both for attached and unattached ("free flying") objects, using a novel method to solve linear systems with singular matrices with a known nullspace. While our method is applicable to general large-strain shape deformation modeling, we use it to create personalized 3D triangle and volumetric meshes of human organs, based on MRI or CT scans. Given a medically accurate anatomy template of a generic individual, we optimize the geometry of the organ to match the MRI or CT scan of a specific individual. Using our method, we successfully extract all hand muscles in six example poses.

Finally, we developed a comprehensive system for hand animation by modeling bones, muscles, tendons, joint ligaments, and fat separately. We gave a pipeline that pre-processes MRI and optical scans into a simulation-ready hand rig. We demonstrated that our model is accurate. Our model does not only match the ground truth skin, with average errors less than 1mm in all example poses, but it also produces matched interior structures in all example poses. Furthermore, our model produces realistic hand motions. It qualitatively reproduces important features seen in the photographs of the subject's hand, such as similar overall organic shape and fold formation.

Chapter 1

Introduction

1.1 Hand modeling and simulation using stabilized magnetic resonance imaging

Hands and their modeling and animation are of paramount importance in numerous applications. In computer games and film, clothing may occlude the body of the characters, but the hands are often exposed and important for the story. In engineering, hand anatomical models can be used to design tools and equipment. In computer vision, anatomically based modeling can improve the tracking of hands, because it provides better statistical priors on hand shapes. In healthcare, accurate hand shapes and motions can be used to design better finger and partial hand prosthetics, as well as better tools for surgeons. As is well-known and argued by medical authorities in the field [62], the dexterity of a hand stems from a thumb opposing four fingers (Figure 1.1), thereby making the hand perfectly suited for precise grasping, lifting, climbing and daily manipulations of objects. To improve realism, virtual hands should be modeled similarly to biological hands, and this requires building precise anatomical and kinematic models of real human hands.

Unfortunately, complex hand biomechanics is currently not modeled, measured, or resolved in any qualitative manner. While there are several computational models of hand kinematics, few have attempted to model internal structures, let alone produce anatomically precise models that match real-world data. Existing detailed human hand anatomy books like [62] focus on applications in

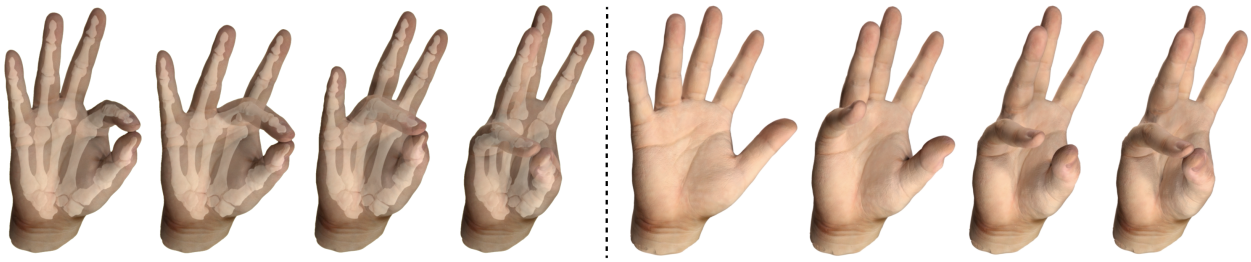


Figure 1.1: **Opposition of the thumb:** Our method produces an accurate data-driven skeleton mesh kinematic "rig." Here, we used our rig to reliably reproduce the well-known opposition of the thumb to all other four fingers. The solid soft tissue was computed using an FEM simulation attached to the articulated bone meshes. The fingers do not merely touch but also orient to be co-planar at the contact location, subject to biomechanical limits. The right-most four images show representative FEM frames of the opposition between the thumb and the pinky finger.

medicine, where the goal is to treat and repair hand injuries, not analyze the functioning of healthy hands.

It is challenging to acquire the motion of the internal human hand anatomy. It must be performed *in vivo* (on a live person), as the hand motion is greatly affected by the tendons, fat tissue, and active muscles. It cannot be reliably performed using exterior scanning techniques (motion capture) due to tissue deformations and sliding. We give a method to acquire high-resolution geometry of the bones of the human hand (collectively referred to as the "skeleton") in multiple hand poses, using MRI scanning. We scan two subjects (1 male and 1 female) in 12 poses. Our poses were selected to reasonably sample the range of motion of each important hand joint. Note that in medicine, it is common to use cadavers for human anatomy research. We exclude such studies because our aim is to study the kinematics of healthy hands, as activated by a live person.

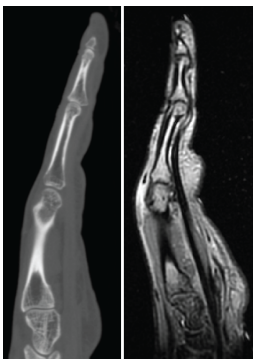


Figure 1.2: Left: Kurihara's CT scan [71]. Reprinted with permission from Eurographics. Right: Our MRI scan.

In modern medicine, there are two imaging techniques that are potentially suitable to acquire 3D hand anatomy. In Computed Tomography (CT), a 3D image is generated from a large number of two-dimensional X-ray images taken around a fixed axis of rotation. In computer animation,

hand bone skeletons have been acquired using CT scans in the pioneering work of Kurihara et. al. [71]. Similarly, Marai et al. have also used CT scans to acquire the carpal bone motion of the human hand [81]. Unfortunately, a CT machine emits ionizing radiation, which can cause cancer in high radiation doses. Building a quality skeleton animation model requires acquiring many poses for each individual, leading to unacceptably high radiation doses. Therefore, we decided to exclude CT from our work. Magnetic Resonance Imaging (MRI) does not use any ionizing radiation nor pose other healthy risks for the vast majority of people. Compared to Kurihara's CT work, which produced CT scans for one individual in five poses, we produced MRI scans for two individuals in twelve poses without any health risks. We performed our MRI scans in a hospital, using clinical 3T scanners manufactured by General Electric. We obtained human subjects research approval for these experiments from our institution's review board (IRB). An additional advantage of MRI over CT is that CT only generates sharp images of bones, whereas MRI reproduces all major hand organs, including bones, muscles, and fat. For example, in the CT images in Kurihara's work [71], one can barely observe any significant hand components other than the bones and the surface (Figure 1.2). Although we do not utilize muscles and fat in this work, their availability bodes well for further research in this area.

Although MRI scanners can provide great anatomical detail, there is a practical challenge that prior work has not addressed: the hand must be kept perfectly still in the scanner for 10-15 minutes. Long scanning times are necessary to decrease the signal-to-noise ratio, which decreases with the square root of the scanning time [115]. If the hand is not still, the scanning image quickly loses resolution and becomes useless. If the goal is to scan just one pose, this is doable (not easily comfortable, but doable) for most people—lie down on the scanner bed, firmly press the hand against the bed, and hold it still during the scan. However, general hand poses cannot be scanned in this manner because they position the palm and fingers in a certain general configuration—for example, bending the fingers, closing the hand into a fist, partially closing the hand, etc. There is no means to physically support the hand inside the scanner. Without support, it is impossible to

sharply scan such hand poses, as a human simply cannot keep an unsupported hand pose still for more than a few seconds.

Our solution is to generate molds that hold the hand in a known and fixed pose. We generate our molds by placing a hand in a chosen pose into a liquid lifecasting solution that solidifies into an elastic rubber-like solid and then retrieve the hand. Thereafter, we fill the hole with liquid plastic, generating a plastic replica of the hand in each pose. This replica captures extremely high detail on the skin's surface. We then scan the plastic replica using an optical 3D scanner (Artec Spider [8]), and repeat the lifecasting process to generate a rubber mold (negative image of plastic hand). We then cut the mold into two parts, place the hand into the mold, cover it with the upper part, and place the hand into the MRI scanner. To the best of our knowledge (including that of our co-author Dr. Matcuk who is a medical doctor/radiologist and active researcher in radiology), we are the first to provide a solution for a stable long-time ($>$10 minutes) MRI scanning of the hand in multiple poses. Another key benefit of our approach is that we can obtain, for each pose, not just the anatomy in that pose, but also a high-resolution surface skin detail in that pose, which can be used for high-resolution hand rendering.

Next, we demonstrate how to segment the MRI scans into bone meshes for each pose. We show how to register the bone meshes to have equal number of vertices and triangle connectivity. Although there is substantial literature on *body* MRI data segmentation, the literature on *hand* MRI segmentation is very sparse. We give new methods that improve the state-of-the-art (Section 4.1). We then demonstrate how to build a data-driven skeleton kinematic model (a skeleton animation "rig"). This rig is able to articulate the skeleton mesh into an arbitrary pose, using interpolation and extrapolation of the acquired MRI meshes. Our rig captures complex real-world rotations and translations of bones relative to their parent bone.

To model the motion of the human hand, we first provide a new Finite Element Method (FEM) simulation method that treats the entire hand as a single soft tissue ("soft tissue rig"). Naive tet meshing produces unstable tet meshes that explode due to pinching of the elastic material at the joints, or over-constraining at the joints. We show how to stably constrain the single soft tissue

tetrahedral mesh to the skeleton mesh animated using our bone rig. We give a method to increase the spatial mesh resolution and adjust elastic material properties in selected regions, which can reliably and automatically simulate folds and creases, both on the palm and under fingers. Further, we show experimentally that our bone rig greatly improves the stability of our soft tissue rig, compared to naive bone rigs. We also demonstrate that our single soft tissue rig outperforms skinning with standard Maya skinning weights.

As is well known, different organs of the hand have very different mechanical properties. For example, a muscle tissue is much stiffer than a fat tissue; and is an active tissue. A hand tendon behaves like an inextensible rod. In addition, adjacent tissues usually move relatively 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. Therefore, to further improve the accuracy of our hand simulation model and capture key features of the human hand, it is important to treat different organs of the human hand 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. Moreover, few methods have attempted to match medical data (namely MRI in our case) in both qualitative and quantitative manner. First, without our proposed capturing method, it is difficult to capture the hand of a specific live subject in different poses. As a result, existing methods use anatomy either obtained from a single medical image or constructed based on medical literature. Second, due to the lack of medical data, existing volumetric simulation models are not designed for matching medical scans such as MRI.

With the MRI scans captured using our method, we are able to explore and design such a volumetric model whereby the organs are modeled as separate but coupled objects, and that matches both internal structure and external appearance of the human hand in example poses. Our work models bones, muscles, joint ligaments, tendons, and fat. For each organ group, we give a complete pipeline that takes medical and/or optical scans as inputs and produces a simulation-ready "rig". We use a comprehensive layered simulation system that begins with the kinematic bones, and is

followed by the bone fascia layer, tendon simulation, muscle simulation and muscle fascia simulation. Finally, we simulate the fat tissue, which gives us the external shape of the hand. 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 make 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: all the organs are volumetrically matching their shapes in the MRI scans, with average errors less than 1mm in all example poses for skin, as compared to an optical scan. Furthermore, our volumetric simulation model produces realistic volumetric deformations of the organs across the entire range of the hand's motion (ROM). Our method nonlinearly "interpolates" the organs in the MRI scans across the entire ROM of the hand. We are not aware of any other work that has demonstrated a volumetric simulation method for the entire hand's musculoskeletal system that matches medical image ground truth data, and that stably interpolates to the entire hand's ROM. 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 activated muscles.

1.2 Modeling of personalized anatomy using plastic strains

In this thesis, we demonstrate how to model anatomically realistic *personalized* three-dimensional shapes of human organs, based on medical images of a real person, such as MRI or CT. Such modeling is crucial for personalized medicine. For example, after scanning the patient with an MRI or CT scanner, doctors can use the resulting 3D meshes to perform pre-operative surgery planning. Such models are also a starting point for anatomically based human simulation for applications in computer graphics, animation, and virtual reality. Constructing volumetric meshes

that match an organ in a medical image can also help with building volumetric correspondences between multiple MRI or CT scans of the same person [96]—for example, for medical education purposes.

Although the types, number, and function of organs in the human body are generally the same for any human, the shape of each individual organ varies greatly from person to person due to natural genetic variation across the human population. The shape variation is substantial: any two individuals' organs $\Omega_1 \subset \mathbb{R}^3$ and $\Omega_2 \subset \mathbb{R}^3$ generally vary by a non-trivial shape deformation function $\Phi : \Omega_1 \rightarrow \Omega_2$ that often consists of large and spatially varying anisotropic strains (see Figure 5.1). A "large anisotropic strain" implies that the singular values of the 3x3 gradient matrix of Φ are both different from each other and substantially different from 1.0, that is, the material locally stretches (or compresses) by large amounts; and this amount is different in different directions and varies spatially across the model.

We tackle the problem of how to model such large shape variations by using volumetric 3D medical imaging (such as MRI or CT scan), and a new shape deformation method capable of modeling large spatially varying anisotropic strains. We note that the boundary between the different organs in medical images is often blurry. For example, in an MRI of a human hand, the muscles often "blend" into each other and into fat without clear boundaries; a CT scan has even less contrast. Therefore, we manually select as many reliable points ("markers") as possible on the boundary of the organ in the medical image; some with correspondence ("landmark constraints") to the same anatomical landmark in the template organ, and some without ("ICP constraints"). Given a template volumetric mesh of an organ of a generic individual, a medical image of the same organ of a new individual, and a set of landmark and Iterative Closest Point (ICP) constraints, we identify how to deform the template mesh to match the medical image.

Our first attempt to solve this problem was to use standard shape deformation methods commonly used in computer graphics, such as as-rigid-as-possible energy (ARAP) [113], bounded biharmonic weights (BBW) [59], biharmonic weights with linear precision (LBW) [137], and a finite element method static solver (FEM) [11]. As presented in Section 5.8, none of these methods

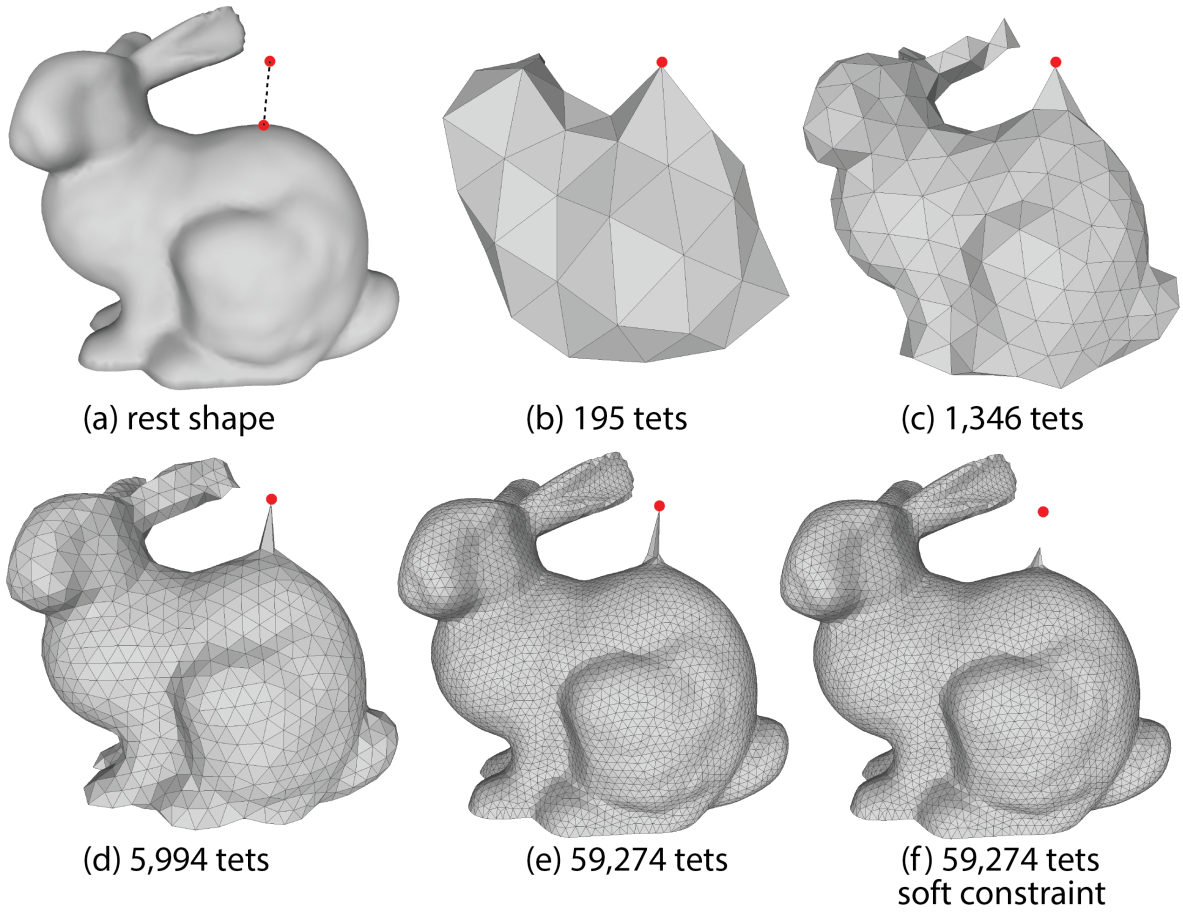


Figure 1.3: **Second-order methods produce spiky outputs as the tet mesh is refined.** Here, we illustrate the output of an FEM static solver under a single hard point constraint (seen in (a)), that is, we minimize the FEM elastic energy under the depicted hard point constraint. Bunny is fixed at the bottom. Poissons's ratio is 0.49. As we increase the tet mesh resolution in (b)–(e), the spike becomes progressively narrower, which is undesirable. Changing the elastic stiffness (Young's modulus) of the bunny does not help (see Appendix G). Converting the constraint into a soft spring constraint also does not help (f); now, the constraint is not satisfied.

was able to capture the large strains observed in medical images. Namely, these standard methods either cannot model point constraints when the shape undergoes large spatially varying strains, or introduce excessive curvature. For example, in the limit where the tetrahedral mesh is refined to finer and finer tets, the FEM static solver produces spikes (Figure 1.3, Appendix G). We explore the limitations of other methods in Section 5.8.

We provide a new shape deformation method that uses plastic strains and the Finite Element Method to successfully model shapes undergoing large and/or anisotropic strains, controlled by

the sparse landmark and ICP point constraints on the boundary of the object. In order to do so, we formulate a nonlinear optimization problem for the unknown *plastic deformation gradients* of the template shape Ω_1, such that under these gradients, the shape Ω_1 transforms into a shape that matches the medical image landmark and ICP constraints. The ICP constraints are handled by appropriately incorporating the ICP algorithm into our method. We note that in solid mechanics, plastic deformation gradients are a natural tool to model large volumetric shape variations. We are, however, unaware of any prior work that has used plastic deformation gradients and the Finite Element Method to model large-strain shape deformation.

In order to make our method work, we needed to overcome several numerical obstacles. The large-strain shape optimization problem is highly nonlinear and cannot be reliably solved with off-the-shelf optimizers such as the interior point method [9]. Furthermore, a naive solution requires solving large dense linear systems of equations. We demonstrate how to adapt the Gauss-Newton optimization method to robustly and efficiently solve our shape deformation problem, and how to numerically avoid dense linear systems. In order to optimize our shapes, it is essential to derive analytical gradients of internal elastic forces and the tangent stiffness matrix with respect to the plastic strain, which will be useful for further work on using plasticity for optimization and design of 3D objects. In addition, we address objects that are attached to other objects, such as a hand muscle attached to one or more bones as well as unattached objects. An example of an unattached object is a liver, where the attachments to the surrounding tissue certainly exist, but are not easy to define. It is practically easier to just model the liver as an unattached object. In order to address unattached objects, we present a novel numerical method to solve linear systems with singular matrices with a known nullspace. Such linear systems are commonly encountered in applications in geometric shape modeling and nonlinear elastic simulation. Our examples include human hand muscles, a liver, a hip bone, and a hip abductor muscle ("gluteus medius"), all of which undergo substantial and non-trivial shape change between the template and the medical image.

Chapter 2

Related Work

2.1 Anatomically based simulation

Anatomically based simulation of the human body has been explored in multiple publications. For example, researchers simulated human facial muscles [109], the entire upper human body [73], volumetric muscles for motion control [74] and hand bones and soft tissue [134]. Anatomically based simulation is also popular in film industry [121]. Existing papers largely simulate generic humans because it is not easy to create accurate anatomy personalized to each specific person. Our method can provide such an input anatomy, based on a medical image of any specific new individual.

2.2 Animating the human hand

Hands are biomechanically complex and it is therefore natural to model their shapes using physically based simulation; such as, for example, simulating a solid mesh constrained to the underlying skeleton [27, 68, 78]. Lee et al. comprehensively modeled the upper human body using anatomically based simulation [73]. However, they excluded hands from simulation and modeled them kinematically. The skin and tendons of the human hand have been simulated to enhance the visual appearance or control of hand articulation [76, 117, 100]. McAdams et al. [82] and Smith et al. [110] used FEM to simulate cartoon hands. For human hands, Kry et al. [70] and Garre et

al. [41] modeled hand deformations using a layer of FEM soft tissue around articulated bones. We give a hand FEM simulation model that is more elaborate than previous approaches. We model spatially varying geometric detail and material properties, automatically remove elastic material in between the heads of bones, and determine attachments to bones at the joints to avoid pinching. If such features are omitted, it is in our experience impossible to stably simulate hands that aspire to look like real hands. Our approach can, for example, model the fold formation on the entire hand.

To a first-degree approximation, hands can be animated using skinning techniques [60]. Skinning can be improved using several techniques, such as dual quaternions [63], or implicit methods [127]. Pose-Space Deformation (PSD) is a method that combines skeleton subspace deformation [79] with artist-corrected pose shapes [75]. It is widely used in industry due to its speed, simplicity and the ability to incorporate real-world scans and arbitrary artist corrections.Kurihara et. al. presented a variant of PSD suitable for hand animation [71], and Rhee et al. demonstrated how to efficiently implement it on a GPU [97].

Recently, Romero et al. [98] gave a surface scanning approach to animate human hands together with the rest of the body. Their model is purely data-driven and does not model interior anatomy. Different from our approach, their goal is to generate a statistical model to parameterize the variations of human hand shapes due to personalization and pose. In contract to data-driven methods [71, 98] and model-based methods [70, 41], we demonstrate how to *simultaneously* acquire both a kinematic model of the internal anatomy and high-quality surface scans in matching poses. We are not aware of any prior work that has achieved this for hands.

Our hands are driven by standard and familiar skeleton joint angle animations. Such animations can be created manually by artists, or acquired from real performances, for example, using gloves [136, 32, 50], or computer vision techniques [72, 135, 92]. In computer animation, there are many methods that compute kinematic hand skeleton models based on surface tracking methods, such as optical or magnetic motion capture; see [140] for a good survey. Unlike surface-based techniques, our imaging-based method can generate internal anatomy, which is required for anatomically based simulation. Because of their small motion and combined effect on the surface

of the palm, surface-based techniques have difficulties tracking the bones in the palm. Our MRI method works equally well for all bones. We were able to extract the rigid motion of palmar bones without any difficulty.

2.3 Acquiring medical images of human hand

Common medical imaging techniques that are in principle applicable to in-vivo measurements include X-rays, Computed Tomography (CT), Positron Emission Tomography (PET), Magnetic Resonance Imaging (MRI), and Ultrasound (for a good review, see [38, 40]). PET requires injecting the subjects with positron-emitting radionuclides, equivalent to 8-years of natural background radiation [95]. Although ultrasound is a safe imaging techniques and has been studied for decades [40, 111], it is not suitable for hands. This is because ultrasound signals are blocked by bones, and are thus unable to generate complex three-dimensional geometry due to sonic occlusions. MRI provides good contrast for all anatomical structures [33, 138], and can capture soft tissues such as muscles and fat. Commercial companies have used MRI to build complete human body 3D anatomy [147]. A few publications analyzed the hand bone geometry using MRI scans [88, 99, 128, 115]. However, because the MRI scanning times need to be long to decrease the signal to noise ratio (10-15 minutes in our work), prior work has not addressed an important limitation. Namely, if the hand is not held perfectly still during the scan, the MRI image is blurred (useless). Of course, humans cannot hold the hand still in arbitrary poses for 10 minutes. Researchers attempted to overcome this limitation by employing clay support, and by asking the subject to hold objects during the scan. Using such an approach, Miyata et al. [88] succeeded in MRI-scanning and modeling 2 finger joints on 4 fingers, in 4 poses. In our work, we scanned the complete hand in 12 poses. We use ergonomic molds which, due to complete hand encasement, provide superior support to using clay or relying on grasping. To combat hand shake, Stillfriend's work [115] used a balancedSSFP MRI scanning sequence to shorten the scanning time to 2-3 minutes. In medicine, this sequence is usually employed to scan moving objects (e.g., a beating heart).

While this reduces hand fatigue, it also decreases tissue detail and sacrifices the overall image quality. Furthermore, even if such procedures are used, the specific MRI-scanned poses in prior work do not correspond to any optical scan of the surface geometry. Note that the quality of any hand skin surface extracted from a MRI scan is far inferior to what can be achieved via surface optical scanning. In our work, we use the PD Cube MRI sequence, which is a common MRI sequence to image stationary objects. That said, the "fast MRI" (balancedSSFP) approaches are orthogonal to our mold stabilization approach and could be combined if so desired.

2.4 Segmentation techniques for medical images of human hands

Most medical imaging segmentation techniques focus on the general body (and many on the hip area), and not on hands (see [84] for a good review). Although segmentation algorithms with prior knowledge have succeeded in automatic musculoskeletal segmentation of the hip [47, 107, 44, 106, 16], they have not been demonstrated to work on hands. Note that hands have a much smaller and more complicated structure. Compared to the hip, a human hand (without the wrist) has 27 bones and 19 flexible joints, which dramatically increases the segmentation complexity. The most common approach (aimed at general human bodies, but also applicable to hands) in practice is to mostly perform a manual segmentation, with the help of a few basic computer vision techniques such as thresholding, filtering and water-shedding [128, 115]. Such capabilities are often integrated into commercial medical image analysis software, such as Amira [7], VTK [130] and ITK-SNAP [58]. We tried Amira, but found it too inefficient for hands and not automated enough for our application. Recently, with the rapid development of GPUs, deep neural network techniques (DNN) became a popular and powerful technique for image segmentation, including MRI segmentation [66, 34]. However, these techniques are still in their infancy, e.g., [66] does not solve the inhomogeneous bone problem. DNN approaches are currently inhibited due to a lack of hand medical imaging datasets and approaches to stably build such datasets.

2.5 Medical image registration

Deformable models are widely used in medical image analysis [83]. Extracting quality anatomy geometry from medical images is difficult. For example, Sifakis and Fedkiw [109] reported that it took them "six months" (including implementing the tools) to extract the facial muscles from the visible human dataset [126], and even with the tools implemented it would still take "two weeks." With our tools, we are able to extract all the 17 muscles of the human hand in 1 day (including computer and user-interaction time). Bones generally have good contrast against the surrounding tissue and can be segmented using active contour methods [120] or Laplacian-based segmentation [48, 134]. For bones, it is therefore generally possible to obtain a "dense" set of boundary points in the medical image. Gilles et al. [46] used this to deform template skeleton models to match a subject-specific MRI scan and posture. They used the ARAP energy and deformed surface meshes. In contrast, we give a method that is suitable for soft tissues where the image contrast is often low (our hand muscles and liver examples) and that accommodates volumetric meshes and large volumetric scaling variations between the template and the subject. If one assumes that the template mesh comes with a registered MRI scan (or if one manually creates a template mesh that matches a MRI scan), musculoskeletal reshaping becomes more defined because one can now use the pair of MRI images, namely the template and target, to aid with reshaping the template mesh [45, 105, 43]. The examples in these papers demonstrate non-trivial musculoskeletal reshaping involving translation and spatially varying large rotations with a limited amount of volumetric stretching (Figure 14 in [43]). This is consistent with their choice of the similarity metric between the template and output shapes: their reshaping energy tries to keep the distance of the output mesh to the medial axis the same as the distance in the template [43], which biases the output against volume growth. For bones, a similar idea was also presented in [108, 104], where they did not use a medial-axis term to establish similarity to a source mesh, but instead relied on a PCA prior on the shapes of bones, based on a database of 29 hip and femur bone shapes. Our work does not require any pre-existing database of shapes. Because our method uses plasticity, it can accommodate large and spatially varying volumetric stretching between the template and the subject. We do not need

a medical image for the template mesh. We only assume that the template mesh is plausible. Of course, the template mesh itself might have been derived from or inspired by some MRI or CT scan, but there is no requirement that it matches any such scan.

2.6 Non-rigid iterative closest point method

Non-rigid ICP methods are widely used for deforming the template mesh to a given target surface mesh (or a dense point cloud) [5, 6, 77]. To apply ICP to medical imaging, one can first delineate the organs in medical images using segmentation techniques, and then deform a template mesh to the segmented surface. In prior work, this has been done by segmenting dense organ boundaries [91, 42]. Organ boundaries in medical images are often not clear, and therefore segmenting dense boundaries is a laborious process, and even then, the boundaries are often unreliable. Our work only requires identifying a sparse set of boundary points in the medical image. Combining such sparse inputs with ICP methods has not been investigated in prior work. One could apply existing ICP methods to a sparsely sampled boundaries, but this produces suboptimal results under large strains and rotations (Figures 5.7, 5.23, 5.25). It is important to emphasize that the source of errors in previous methods is not the miss-correspondence of sparse markers: suboptimal results are produced even if given perfect correspondence (Figure 5.25).

2.7 Geometric shape modeling

Geometric shape modeling is an important topic in computer graphics research; e.g., see the Botsch and Sorkine [20] survey and the SIGGRAPH course notes by Alexa et al. [3]. Popular methods include variational methods [21], Laplacian surface editing [112], as-rigid-as-possible (ARAP) deformation [56, 113], coupled prisms [22] and partition-of-unity methods such as bounded biharmonic weights (BBW) [59] and biharmonic weights with linear precision [137]; we provide a comparison in Section 5.8 and in several other Figures in the thesis. Our method reconstructs the surface shape from a set of un-oriented point observations; this goal is similar to variational

implicit surface methods [123, 54]; we give a comparison in Section 5.9. Point clouds can also be used to optimize rest shapes [125] and material properties of 3D solids [133]. Such a method cannot be applied to our problem because it assumes a 4D dense point cloud input; whereas we assume 3D sparse point inputs as commonly encountered in medical imaging. Point constraint artefacts of second-order methods can be addressed using spatial averaging [17, 64]; however this requires specifying the averaging functions (often by hand) and, by the nature of averaging, causes the constraints to be met only approximately. Our method can meet the constraints very closely (under 0.5 mm error in our examples), i.e., in the precision range of the medical scanners.

2.8 Plasticity

Elastoplastic simulations are widely used in computer animation. O'Brien et al. [93] and Muller and Gross [89] used an additive plasticity formulation, whereas Irving et al. [57] presented a multiplicative formulation and argued that it is better for handling large plasticity; we adopt multiplicative formulation in our work. The multiplicative model was used in many subsequent publications to simulate plasticity, e.g., [15, 116, 29]. Because plasticity models shapes that undergo permanent and large deformation, it is in principle a natural choice also for geometric shape modeling. However, such an application is not straightforward: an incorrect choice of the optimization energy will produce degenerate outputs, elastoplastic simulations in equilibrium lead to linear systems with singular matrices, optimization requires second-order derivatives for fast convergence, and easily produces large linear systems with dense matrices. We present a solution to these obstacles. To the best of our knowledge, we are the first to present such a comprehensive approach for using plasticity for geometric shape modeling with large and anisotropic strains.

2.9 Anatomy transfer

Recently, great progress has been made on anatomically based simulations of humans. Anatomy transfer has been pioneered by Dicko and colleagues [35]. Anatomical muscle growth and shrinkage have been demonstrated in the "computational bodybuilding" work [102]. Kadlecek et al. [61]

demonstrated how to transfer simulation-ready anatomy to a novel human, and Ichim et al. [55] gave a state-of-the-art pipeline for anatomical simulation of human faces. Anatomy transfer and a modeling method such as ours are complementary because the former can interpolate known anatomies to new subjects, whereas the latter provides a means to create the anatomies in the first place. Namely, anatomy transfer requires a quality anatomy template to serve as its source, which brings up the question of how one obtains such a template. Human anatomy is both extremely complex for each specific subject, and exhibits large variability in geometry across the population. Accurate templates can therefore only be created by matching them to medical images. Even if one creates such a template, new templates will always be needed to model the anatomical variability across the entire population; and this requires an anatomy modeling method such as ours.

Chapter 3

Acquiring MRI Scans in Many Poses

We now describe our procedure to acquire bone geometry **and** detailed external hand surface geometry and color texture in **multiple** hand poses, spanning the typical range of motion of the human hand (Figure 3.1). Our skeleton geometry is acquired using MRI, and the external geometry is acquired using laser scanner metrology (for shape) and photogrammetry (for color texture). For each bone, we generate a mesh for each pose, with equal mesh connectivity. We then use the skeleton geometry to run FEM simulations to generate high-quality unscanned "in-between" poses, producing registered high-resolution surface geometry that can be articulated to any continuous pose of interest.

We stabilize the human hand inside the MRI scanners in known prescribed poses, by manufacturing rigid molds. Our technology "silver bullet" is the observation that a human hand can be physically **cloned** into a shape made of a variety of materials (plastic, silicon, etc.) using lifecasting materials (Figure 3.3). Inspired by the film special and visual effects industry, we observe that it is possible to generate extremely precise replicas of human hands, using commonly available rubber-like materials such as the AljaSafe^{TM} material from Smooth-On, Inc [4]. We note that lifecasting has been used in the visual effects industry to model human faces; we are not aware of prior usage to model hands in multiple poses. Lifecasting has been used for fabricating an accurate neutral shape of a human hand in robotics [69, 103], to perform hand control, such as for grasping.

We perform lifecasting using AljaSafe [4], an alginate skin-safe material naturally occurring in the cells of brown algae. The resulting hand replicas capture extremely detailed surface hand

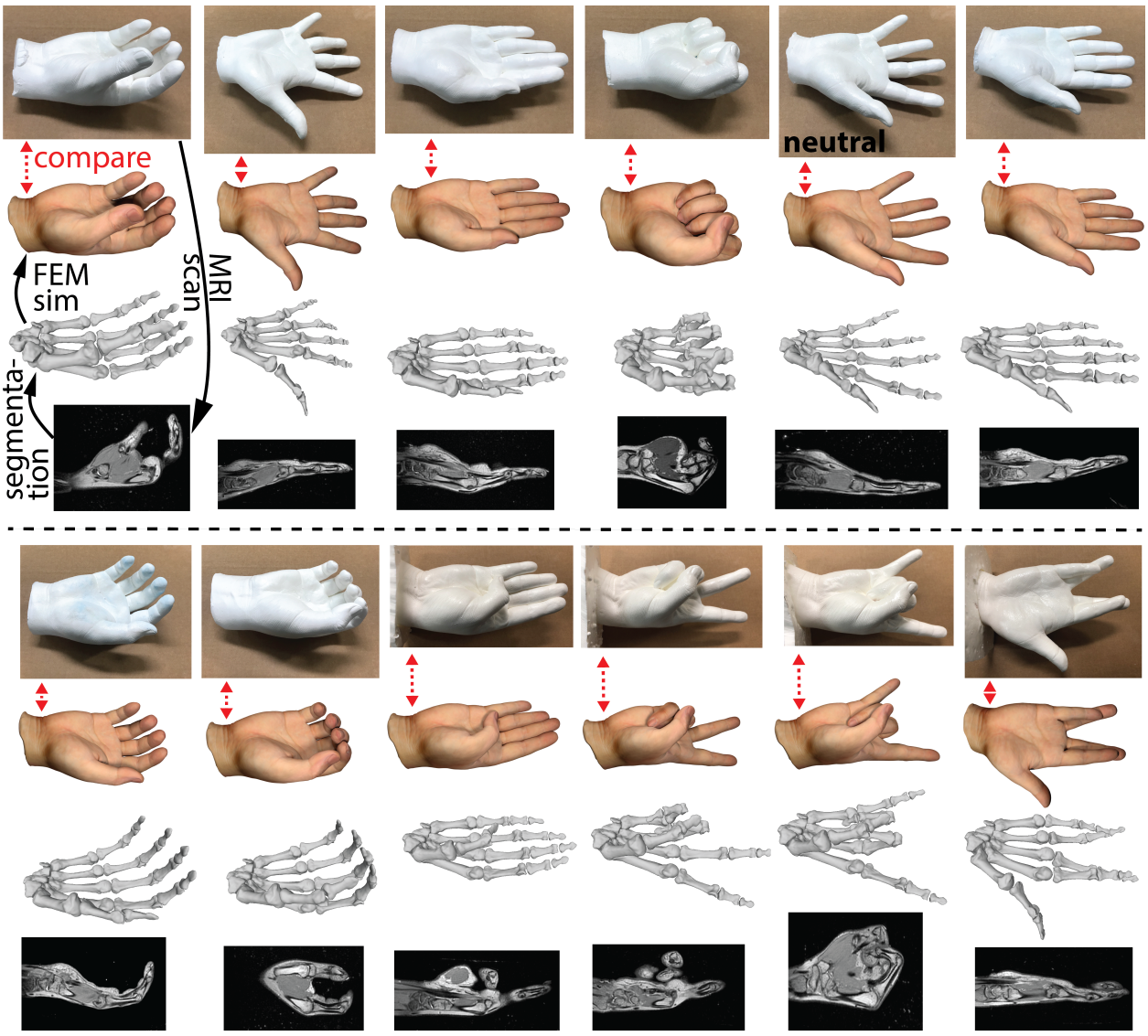


Figure 3.1: **12 MRI-scanned poses (1 subject; male; late 20s).** Four rows are the cloned plastic hand (the "ground truth" shapes), our FEM result, skeleton mesh obtained from the MRI scan, and the MRI scan. The last four plastic poses have a plastic stand (seen on the left of each image) to easily stand up on a table. FEM closely matches the plastic hand in all poses, despite not actually using plastic shapes anywhere in our system (except the neutral shape: column 5 in the top set). The solid black arrows indicate the logical sequence of steps; the FEM and cloned hands are shown adjacent to each other for easier comparison. The image can be zoomed in on.

features, down to individual pores and tiny lines on the hand surface (Figure 3.3, g). In order to make the replicas, we prepare a liquid AljaSafe solution (mixture of powder and water) in a properly hand-sized bucket, and then position the hand into it, in a chosen pose. The solution

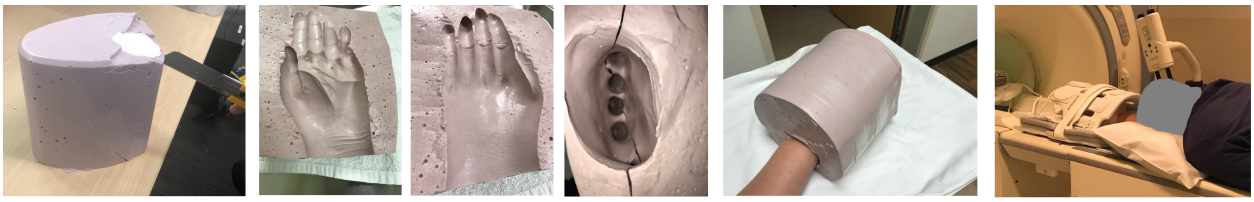


Figure 3.2: **Stabilized MRI scanning:** (a) The mold with a plastic hand before cutting. (b1,2,3) Mold has been cut into two parts. (c) Hand secured into the mold. (d) MRI scanning with the mold. Clinical MRI scanner is manufactured by General Electric. Magnetic field strength of 3T, and resolution of 0.5 x 0.5 x 0.5 mm^3.

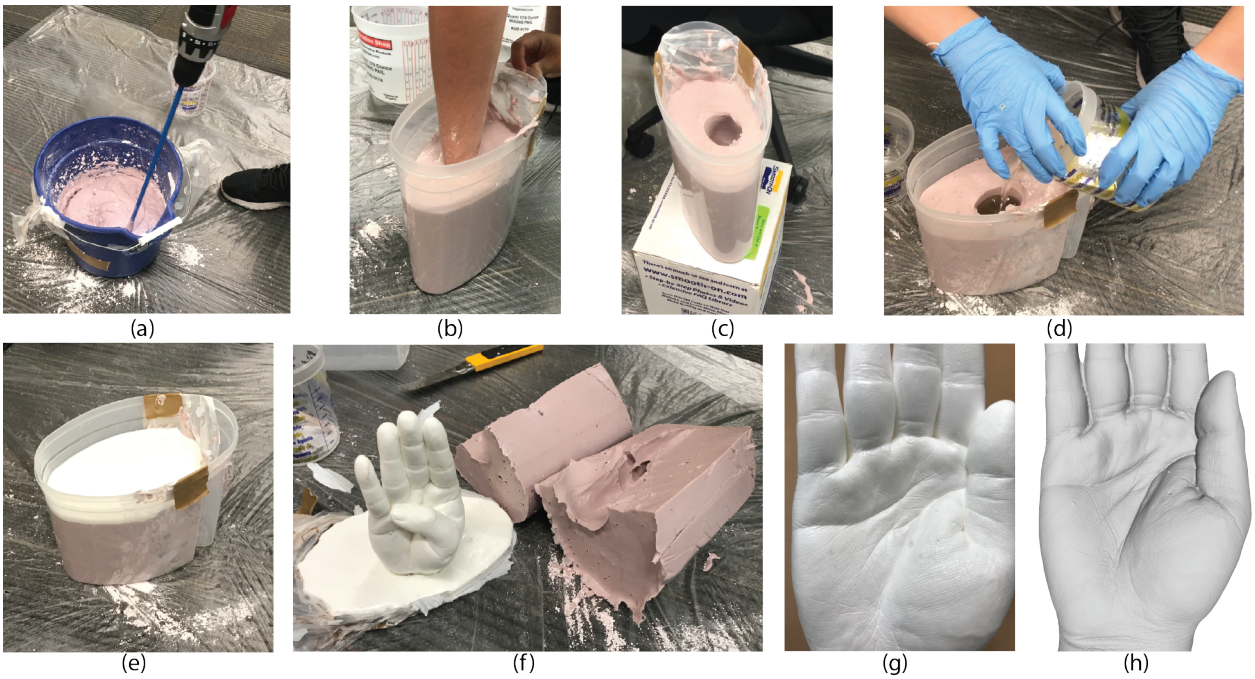


Figure 3.3: **We "cloned" this hand onto a plastic hand.** (a) Mixing the AljaSafe. (b) The hand in a bucket filled with liquid AljaSafe. (c) Hand removed, leaving a hand-shaped hole. (d, e) Casting a liquid plastic into solidified AljaSafe. (f) Removed the AljaSafe to obtain the plastic hand. (g) Photo of plastic hand. (h) Rendered 3D-scanned mesh of plastic hand. Scanner: Artec Spider. Note the high-resolution detail in g, h (This can be zoomed in on in PDF).

solidifies in approximately 8-10 minutes into a rubbery solid. The shape of the hand and the micro-detail on the surface get imprinted into the AljaSafe surface. We note that AljaSafe gently encases the hand. It does not change volume when it solidifies, so there is minimal hand compression; only small water-like hydrostatic pressure due to gravity. Because AljaSafe is sturdy, but sufficiently

flexible, it is easily possible to withdraw the hand, without damaging the imprinted surface detail. This process produces a bucket of solidified AljaSafe rubber with a hole in the shape of the hand pose. We then fill this hole with liquid plastic (Smooth-Cast^{TM} 300Q), which solidifies in a few minutes. We then remove the AljaSafe rubber. The result is a high-quality plastic replica of the hand in the specific pose (Figure 3.3, g, h). This manufacturing process is also shown in our video. We note that the subject just has to approximately eyeball each pose during lifecasting. The created mold and the plastic hand are automatically consistent.

We then scan the plastic hand using a precise laser scanner, such as Artec Spider [8]. We scan the plastic hand, as opposed to the real hand directly, because the plastic hand is perfectly still, and therefore such a scan can achieve high precision (Figure 3.3, h), on the order of 0.05 - 0.1mm with Artec Spider, producing a mesh with 11M triangles. For comparison, the number of hand triangles in the dataset on the MANO website [98] is 100K. The Artec Spider precision already is an order of magnitude better than the precision of the skin geometry acquired by our MRI scan (which is 0.5 - 1mm). We are not aware of any prior computer graphics work to use such a chemistry-based solution to "clone" hands. We observe that computer graphics and vision acquisition research typically focuses on optical surface scanning technologies, e.g., reconstructing shapes from multiple camera views. Chemistry-based approaches offer clear advantages in their ability to capture high-precision surface detail on human body parts with **minimal** costs: we only spent about $600 worth of chemical materials to make our replicas for 1 subject.

By repeating the above process for every pose, we obtain a high-quality plastic hand replica in each pose, as well as a high-precision surface geometry scan of each shape. Next, we use these plastic hands to stabilize the real hand during the MRI scan, by generating a **mold** into which the subject inserts the hand prior to MRI scanning (Figure 3.2). The mold keeps the hand still in a fixed, known, optically scanned pose. The presence of the mold does not corrupt the MRI signal in any way. We tested this in practice and discussed the issue with MRI technicians. Our mold generates almost no signal outside of the hand, and zero signal inside the hand. We generate the mold by repeating the AljaSafe casting process. We prepare the liquid AljaSafe solution, and then

place the **plastic** hand into it. After AljaSafe solidifies, we cut the resulting mold in two parts along the hand's frontal (i.e., coronal) plane, using a precision knife (Figure 3.2, a). For MRI scanning, we place the lower part of the mold onto the scanner bed. The subject inserts her/his hand into the mold, and then the hand is covered and secured by the top part of the mold, and additionally secured and fastened with tape (Figure 3.2, c). Because the top mold piece is largely supported by the bottom piece, and because our molds are precise hand negative images, there is only a minimal gravity and twisting during MRI scanning. Note that if one does not need the plastic hand, one can already use the first AljaSafe cast for the MRI mold. However, molds obtained from plastic hands can reorient the hand, for MRI ergonomics. The mold has to be sturdy to keep the hand still, but still somewhat flexible, so that the subject can "nudge" the hand position slightly before the scan, for ergonomics during the scan. This will inevitably result in a small mismatch between the plastic hand pose that was used to make the mold, and the actual scanned MRI pose. We address this in Section 4.4.

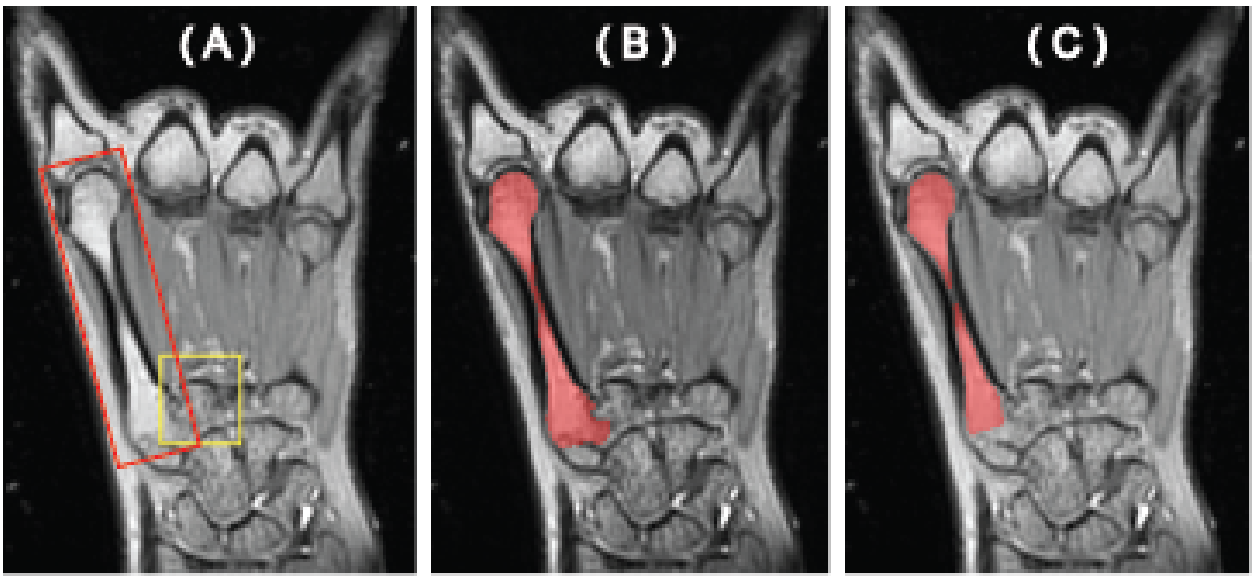


Figure 3.4: **Our method outperforms prior hand bone segmentation methods.** It is challenging to segment the "metacarpal II" bone (red rectangle) for two reasons: (1) the MRI signal (depicted in A) on the bone is not of uniform intensity, and (2) the neighboring bone "metacarpal III" (yellow rectangle; only head of the bone is visible in this 2D view) has very similar signal intensity to "metacarpal II." Our method (depicted in B) successfully segmented this challenging case. The prior method [99] uses a region-based active contour method and produces a suboptimal result: a significant part of II's bone head is missing (C).

Chapter 4

Hand Simulation using Hand Skeleton and Soft Tissue

4.1 Segmenting MRI data into bone meshes

The MRI scan data consist of a regular volumetric grid of scalar values proportional to tissue mass density, and must be segmented into individual bone meshes. Although this task is well-understood for general full-body MRI scans, the literature on segmenting **hand** MRI scans is very sparse. We found only a single research paper (in any field) that discusses (in any substantial detail) how to segment bones from **hand** MRI images [99], and even this publication only showed a single example. We implemented their method and found it challenging to use: (1) there are more than four parameters to tune, and (2) the result cannot be incrementally enhanced by users. Moreover, in many cases it did not work well (for example, see Figure 3.4), so we abandoned it. We give a new technique to perform such a segmentation. In general, common issues in MRI images include (a) inhomogeneous bone tissue intensity: cortical bone (i.e, outer bone layer) is darker than cancellous bone (inner layer), and (b) unclear and fuzzy bone boundaries in articulation areas, as described in [106].

We give a method that segments the image with as few parameters as possible and is as automatic as possible. We use a variant of Laplacian-based segmentation [47], but show how to extend it to 3D in a scalable manner. We also give an intuitive interface specifically for segmentation of hands. Our method has only one parameter, and users are able to incrementally improve the result by delineating additional bone voxels which the segmentation classified incorrectly. The algorithm

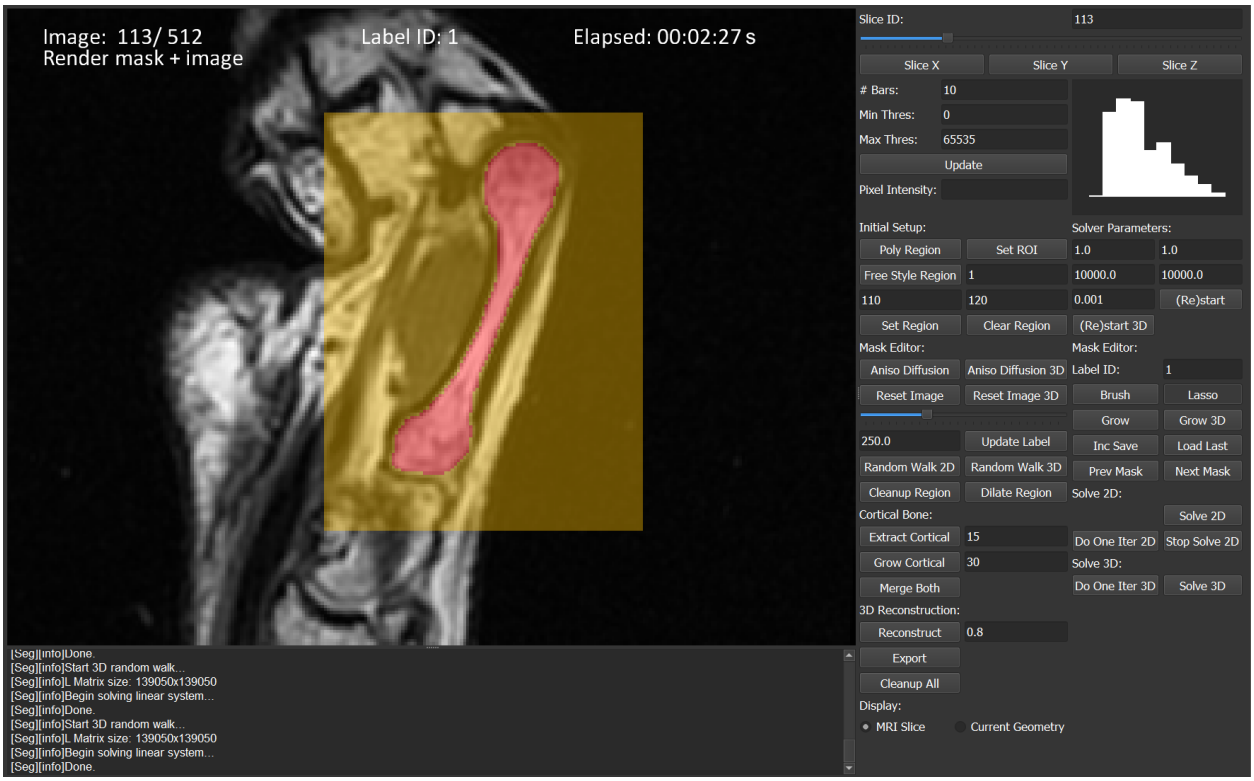


Figure 4.1: **Our segmentation interface.** The result of the first step of our 3D Laplacian-based segmentation. The segmented bone is red. Yellow is the local segmentation region selected by the user for this bone.

in [47] is unacceptably slow when solving the Laplace equation in 3D on the entire volume. Our solution is to segment each bone separately and combine the results together. When merging segmentations into a global labeled volume, we resolve any labeling conflicts by treating conflicted voxels as unlabeled voxels again, and resolve the Laplacian problem again in a local region. Using our method, we were able to segment a 512x512x512 volume of the cancellous bones of the entire hand in about 3 hours (5-10 minutes per bone). For cortical bones, there is no existing work on segmentation without using training data (which is unavailable for hands) [106]. The Laplacian method does not work here because the cortical bone is dark in the MRI images and too similar to tendons. We segment it by the classical computer vision watershed method [85] in 3D. We deem the connected component closest to the cancellous bone as the cortical segmentation. Finally, we merge each cortical and cancellous pair into a single bone using dilation and/or random walk [47].

After the medical image segmentation, it is not difficult to build the implicit function for each bone from the labeled images. With such implicit functions, we can easily generate the explicit 3D geometry using an isosurface meshing algorithm [19].

We now describe how we register each bone's pose-varying meshes to the same number of vertices and triangle connectivity, and how we compute a rigid transformation between the bone in the neutral pose and all other poses. For the neutral pose, we perform complete segmentation of the bone as described in the previous paragraph. This gives us a mesh of the cancellous part of the bone, and a complete bone mesh combining both the cancellous and cortical parts of the bone, in the neutral pose. For all other poses, we only perform cancellous segmentation. This saves a lot of time and is all that is needed. Unlike the cancellous bone which is white and has good contrast, the cortical bone is dark and intensive to segment in terms of manual work; and its segmentation in multiple poses is redundant. We bypass the need for segmenting cortical bones in non-neutral poses by executing a rigid Iterative Closest Point (ICP) algorithm [18] to align the neutral *cancellous* bone mesh onto the cancellous mesh of each pose, extracting the translation $x \in \mathbb{R}^3$ and rotation matrix R for each pose. We then transform the entire neutral bone mesh using the rigid transformation (x, R). This produces the output of our segmentation: a mesh for each bone in each pose, with the same connectivity across all poses. We smooth the bone meshes using 2 levels of the Laplacian smoothing in MeshLab [30]. Our final skeleton has 155,450 triangles total for 23 bones. Using these methods, we successfully segmented 12 hand poses of our two subjects (Figure 3.1). Figure 4.1 shows a screenshot of our segmentation user interface. Next, we demonstrate how we can animate this skeleton mesh, by building a "skeleton rig."

4.2 Skeleton kinematic model

In computer animation, the motion of the human hand is typically modeled using a hierarchical joint structure; and we adopt this approach also in our work. Controlling motion by prescribing joint angles using commonly available inputs (mocap, IK, keyframing, etc.) is commonly done in

practice because it is easier (and admittedly less principled) than optimizing muscle actuators [73]. Given the vector of joint angles θ, a typical pipeline in computer animation industry today is **not** to compute any bone geometry, but instead to bypass bones and apply a skinning algorithm that computes the positions of the skin vertices, based on θ and some suitably defined skinning weights, optionally with sculpted pose-varying corrections. Such modeling is imprecise because the fat layer and the skin are sliding relative to the bones, and because real bones undergo general 6-DOF rigid body motion relative to one another. The rigid transformations between bones depend nonlinearly on θ, and are dictated by the anatomical constraints between the bones and/or muscles, such as ligaments and tendons, and muscle activations. Accurate anatomical modeling requires computing the bone geometry (mesh vertex positions) in arbitrary hand poses. We therefore depart from previous work and give a new kinematic model (a "bone rig") that computes the positions of all vertices in all bone meshes, based on θ. The input to our rig is θ, and the output is the anatomically based rigid transformation (translation and rotation) of each bone. We obtain our rig by fitting it to our segmented bone MRI meshes in the captured poses (Section 3). Our model can both interpolate and extrapolate the MRI-acquired poses. In later sections, we will use our bone rig to compute FEM simulations of the soft tissue of the hand.

We note that Kurihara et. al. [71] optimized bone rotation centers to data, but did not optimize joint rotation axes. Instead, they specified the relative rotation of each joint via three Euler angles corresponding to unoptimized axes that in general do not match biomechanical axes. For example, the finger DIP and PIP joints rotate mostly around one axis. With unoptimized axes, none of the elemental Euler rotations correspond to the main biomechanical axis of rotation. Hence, the animator has to either accept inaccurate motion, or animate all three Euler angles simultaneously. In contrast, we optimize rotation axes to the MRI-acquired bone motion. We give a rig where 1-DOF and 2-DOF joints can be animated by specifying only 1 and 2 angles, respectively; namely the angles of rotations around our optimized joint axes (Figure 4.4). Our rig drives the biomechanical joint axes directly, and as such requires fewer parameters to animate.

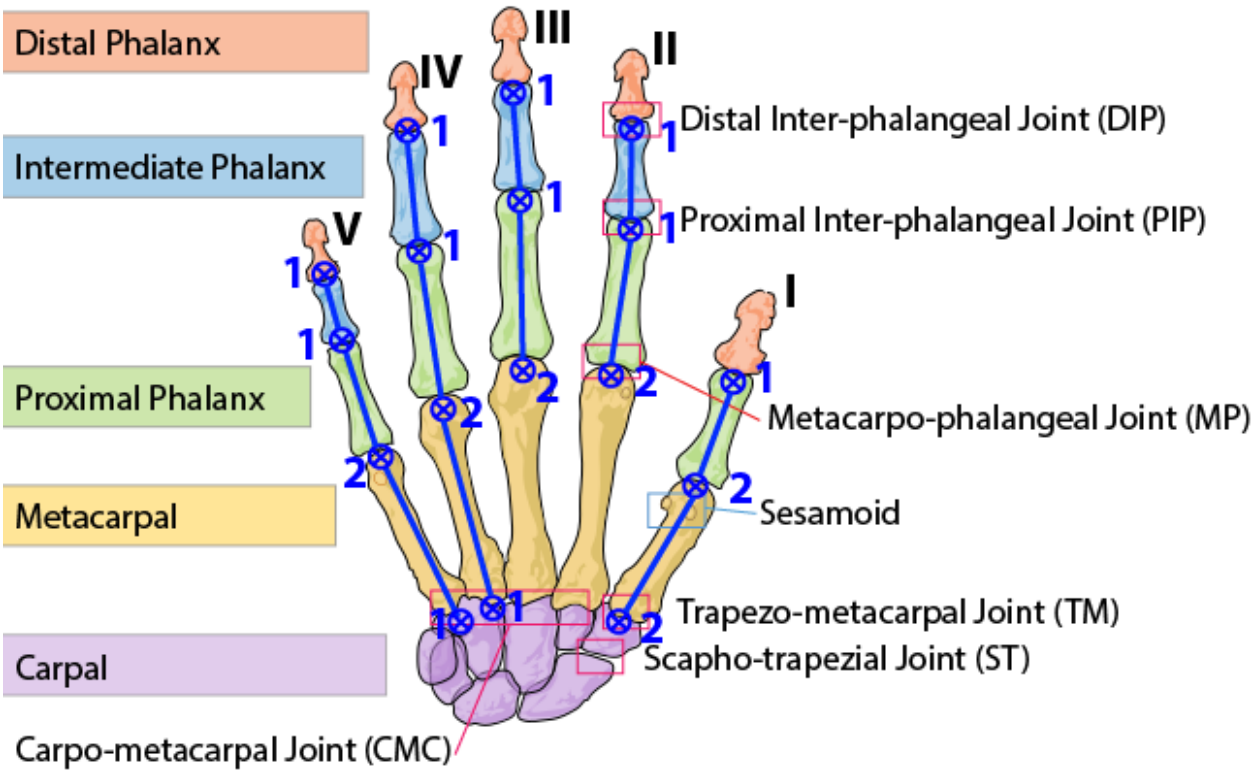


Figure 4.2: **The bones and joints of a human hand.** Our joint hierarchy is indicated with a superimposition in blue, with the number of joint DOFs indicated.

Hand kinematics Modeling the hand joints and bone motion is a complex problem. The human hand (see Figure 4.2) has 27 bones: 5 distal phalanges, 4 intermediate phalanges, 5 proximal phalanges, 5 metacarpals and 8 carpals, most of which are movable. The hand has 9 inter-phalangeal joints (DIPs and PIPs), 5 metacarpo-phalangeal joints (MP) and 5 carpo-metacarpal joints (CMC). The thumb is the only finger capable of opposing the other four fingers; therefore, it has a larger range of motion and greater complexity. The CMC joint of the thumb is usually called Trapezo-Metacarpal joint (TM). Researchers also study the inter-carpal joints, i.e., scapho-trapezial joint (ST) [62]. The employed models vary largely according to the specific setup and application. The number of modeled joints can range from 5 to 36 [143]. The number of DOFs of a single joint can range from 1 to 3. The motion of carpal bones is small and on (or slightly over) the boundary of our MRI scanning volume for our male subject; as commonly done [115], we do not model this motion. The IP and CMC joint are usually modeled to have one rotation axis (or zero for CMC

Table 4.1: **Bone rig errors for each joint.** We list them separately for rotations (R) and translations (x). Here, FM_R and FM_x are the absolute errors under our Full Model (i.e., quadratic fitting of $\hat{R}$ and $\hat{x}$), while PM_R and PM_x are the errors under a Partial Mode, in which $\hat{R} = I$ and $\hat{x} = 0$. The error is computed as the average difference between the output of our bone rig and the ground truth (segmented MRI data), across all 12 poses (male subject). It can be seen that the fitting of $\hat{R}$ and $\hat{x}$ decreases the error. The first and second halves of the table present 1-DOF and 2-DOF joints, respectively. Observe that the pinky finger has the largest error; this is because it is the smallest. Consequently, it is resolved with fewer voxels in the MRI scan.

Joint	FM_R [deg]	PM_R [deg]	FM_x [mm]	PM_x [mm]
DIP I	2.40	2.98	0.52	0.67
DIP II	2.10	3.25	0.31	0.40
DIP III	1.93	2.87	0.22	0.36
DIP IV	2.41	2.78	0.30	0.33
DIP V	2.84	3.30	0.35	0.73
PIP II	1.90	2.08	0.32	0.36
PIP III	1.58	2.19	0.27	0.50
PIP IV	1.32	2.81	0.34	0.46
PIP V	2.77	3.44	0.26	0.31
CMC IV	2.06	2.07	0.69	0.70
CMC V	2.74	2.79	1.07	1.15
MP I	0.59	1.03	0.44	0.80
MP II	1.58	1.82	0.55	0.76
MP III	1.76	1.87	0.37	0.65
MP IV	0.87	2.57	0.24	0.51
MP V	1.67	1.81	0.46	0.79
CMC I	0.79	1.94	0.90	2.26
Average	1.84	2.45	0.45	0.69

II and III), and MP and TM joints to have two rotation axes [62, 49]. We adopt these conventions in our work. There are 17 joints in total in our human hand model, of which 11 are single-axis and 6 are two-axis, for a total of 23 degrees of freedom (Figure 4.2). We assign no degrees of freedom to CMC II and III because it is well-known that these joints almost do no move [10]; in our dataset, maximum motion was under 2 degrees. The configuration of the entire joint hierarchy is then specified via the vector θ of 23 joint angles. These angles can be driven using any suitable computer animation method, such as keyframe animation, motion capture or even motion control.

Our bone rig We first describe our approach for 1-DOF joints. Consider one specific parent-child bone pair. During hand articulation, both the parent and child undergo rigid body motion. In this section, we conceptually undo the parent's transformation so that the parent is stationary in the world, and the child undergoes some rigid body motion relative to parent (Figure 4.3). All motions of the child bone in this section are to be understood in this way. Denote the angle of the 1-DOF rotation of the child bone by the scalar ϕ. Our MRI scan and its segmentation give us the rigid transformation (x_i, p_i) of the child bone in each pose i, for $i = 1,\ldots,N$, where N is the number of acquired poses. Here, $x_i \in \mathbb{R}^3$ is the translation vector and $p_i \in \mathbb{R}^4$ is the unit quaternion. In pose i, the child bone undergoes rigid transformation $X \mapsto R_i X + x_i$, where R_i is the rotation matrix corresponding to p_i, and $X \in \mathbb{R}^3$ is an arbitrary point on the child bone. Our task is to discover a unit rotation axis $a \in \mathbb{R}^3$, and angles $\phi_i \in \mathbb{R}$, such that p_i is as close as possible to a rotation by ϕ_i around axis a. This leads to a constrained optimization problem for a and angles ϕ_i:

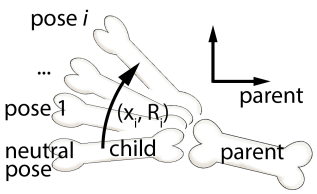


Figure 4.3: Transformation (x_i, R_i) is expressed in the parent's coordinate system. It transforms the neutral child bone into pose i.

$$\max_{\substack{a,\phi_1,\ldots,\phi_N \\ \text{s.t. } a^T a=1}} \sum_{i=1}^{N} \operatorname{dot}\Big(p_i,\ \cos(\frac{\phi_i}{2}) + \sin(\frac{\phi_i}{2})a\Big). \tag{4.1}$$

Here $\operatorname{dot}(p,q)$ denotes the dot product of quaternions p and q. We robustly solve this problem using the IPOPT constrained nonlinear optimizer [131] with ADOL-C automatic differentiation [132].

After a and ϕ_i are known, we construct our rig as follows. Given an arbitrary angle ϕ, we model the rigid transformation of the child bone relative to its parent as

$$X \mapsto \hat{R}(\phi)R(\phi)\big(X - C\big) + C + \hat{x}(\phi), \tag{4.2}$$

where $R(\phi)$ is the rotation around axis a by angle ϕ, and $\hat{R}$ and $\hat{x}$ are the residual rotation and translation, respectively (to be determined). Here, $C \in \mathbb{R}^3$ is a center of rotation, which we determine using least squares so that $\hat{x}(\phi)$ is minimized, assuming $\hat{R} = I$,

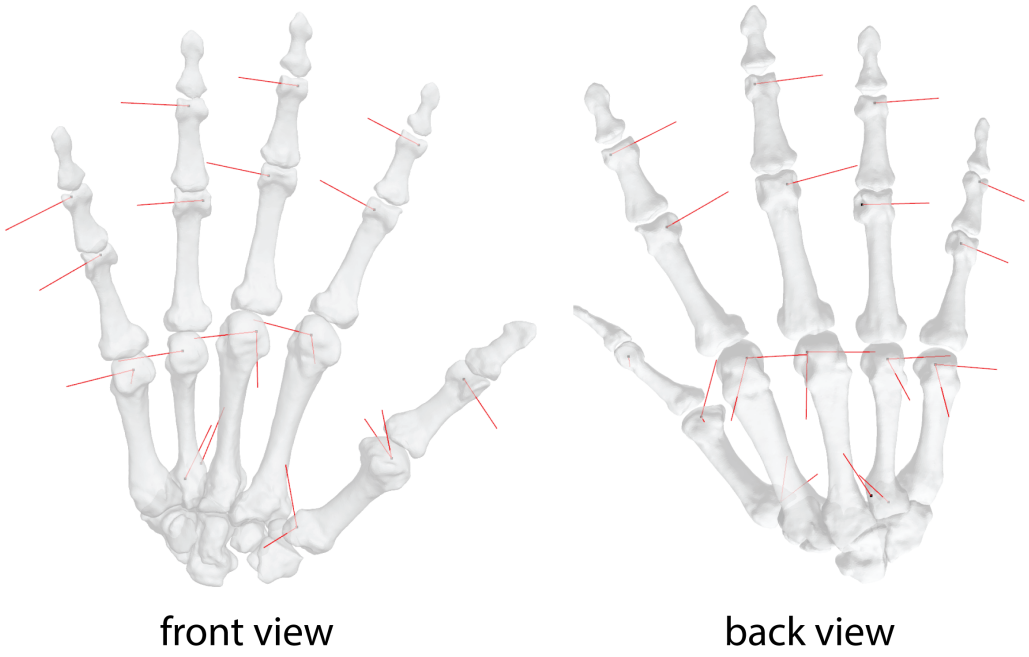


Figure 4.4: **The rotation axes and rotation centers of our joints.** The bone meshes are shown in a transparent style. The axes are shown as red lines. The centers are shown as black dots.

$$\min_{C} \sum_{i=1}^{N} ||x_i + (R_i - I)C||^2. \tag{4.3}$$

Finally, we compute the residual translations $\hat{x}(\phi_i) = x_i + (R_i - I)C$ and residual rotations $\hat{R}(\phi_i) = R_i R(\phi_i)^T$ at each sampled pose i. The only remaining task is to interpolate / extrapolate these residuals to arbitrary ϕ. We achieve this by representing $\hat{R}$ using Euler angles (note that $\hat{R}$ is a small rotation), and then interpolating / extrapolating each degree of freedom of translations and rotations by fitting a 1D quadratic function to samples $(\phi_i, \hat{x}_i(\phi_i))$ and $(\phi_i, \hat{R}_i(\phi_i))$, respectively. We initially employed natural cubic splines, and then also polyharmonic splines; however, this resulted in a lot of spurious wiggles in the rig; hence, we settled for quadratic functions which gave smooth residuals. We extrapolate the functions using the tangents at each end of the samples. We experimentally observed that the residuals $\hat{x}_i$ are small, on the order or smaller than the accuracy of our MRI scan

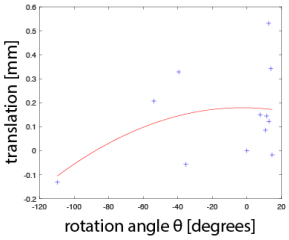


Figure 4.5: Fitting the residual $\hat{x}$.

(0.5 mm) (Figure 4.5), and the wiggles in $\hat{x}$ are therefore mostly just experimental noise. Our conclusion is that bones in real hands, to a large extent, do rotate around parent bones around a fixed center C. However, the deviation away from the "fixed-rotation-center" model is still substantial. Table 4.1 analyzes the accuracy of our bone rig. It demonstrates that using our quadratically interpolated offsets $\hat{R}$ and $\hat{x}$ improves the accuracy of the rig. Even if one accepts fixed centers, one still needs our method to compute the centers and the rotation axes; otherwise suboptimal shapes occur (Figures 4.10, 4.14).

Our approach for two-dimensional joints is similar, except that we now have to find two orthogonal unit rotation axes a and b and a pair of joint angles ϕ_i and ψ_i. Our kinematic model is that we first perform a rotation around a by ϕ_i, followed by a rotation around the rotated axis b by ψ_i. Therefore, we maximize

$$\max_{\substack{a,b,\phi_1,\ldots,\phi_N,\theta_1,\ldots,\theta_N \\ \text{s.t. } a^T a=1, b^T b=1, a^T b=0}} \sum_{i=1}^{N} \text{dot}\Big(p_i,\ r_i q_i\Big),\ \text{where} \tag{4.4}$$

$$q_i = \cos(\frac{\phi_i}{2}) + \sin(\frac{\phi_i}{2})a, \quad r_i = \cos(\frac{\theta_i}{2}) + \sin(\frac{\theta_i}{2})b_i', \tag{4.5}$$

and b_i' is obtained by rotating vector b using q_i. As in the 1D case, this optimization is greatly facilitated by the ADOL-C automatic differentiation library. The computation of the entire rig took 1 minute total for all poses and the entire skeleton. The rotation axes are shown in Figure 4.4.

With our bone rig, we can simulate the human hand by considering the rest of the human hand as a single soft tissue. Therefore, Our simulation model consists of bone geometry (Section 3), surrounded by an elastic soft tissue modeled as a tetrahedral mesh. The bone geometry is animated using our skeleton rig (Section 4.2). We now describe how we generate a stable tetrahedral mesh (Figure 4.6), assign its material properties, and how we constrain the tetrahedral mesh to the bones for FEM simulation.

4.3 Generation of the tet mesh outer surface

We start with the high-resolution (11M triangles) mesh of the hand in the neutral pose, obtained using optical scanning (Artec Spider) of the plastic neutral hand. Because the fingers are sufficiently spread in this pose, this mesh had no webbing or occlusion issues, and only tiny imperfections which we easily cleaned in Maya using smoothing. We then simplify this mesh to a smaller mesh (0.5M triangles) using MeshLab; this is done to make the rest of the pipeline more tractable. The average edge length in this mesh is 0.5mm, however the edge lengths are not uniform and some triangles are of bad quality. We then manually (using painting in Maya) separate the triangles in two groups: those on creases (where higher deformation precision is needed; both at the finger joints and on the palm), and those elsewhere ("non-crease"). We then constrain the crease triangles, and remesh the non-crease triangles using CGAL's isotropic_remeshing tool, with a 2.5mm target edge length. Next, we constrain the non-crease triangles and remesh the crease triangles using the same tool, at 0.5mm. We note that such a 2-stage remeshing process is necessary; otherwise, artefacts appear (Figure 4.7, bottom-left). This procedure gives us a good isotropic manifold triangle mesh $\hat{\mathscr{T}}_{\text{outer}}$ (31,986 triangles) of the hand's neutral pose surface, with smaller edge lengths in the crease areas. We need to correct this mesh, as described next.

4.4 Resolving discrepancies due to a small pose change of the subject in the mold prior to MRI scanning

Note that $\hat{\mathscr{T}}_{\text{outer}}$ encloses the surface of the plastic hand, which is the negative image of the MRI mold of the neutral pose. However, when the hand is placed into the rubber-like mold for MRI scanning, for ergonomics reasons, the hand slightly changes its pose as the subject settles it into the mold prior to scanning. Also, the rubber-like mold somewhat gently squeezes the hand, which slightly alters its volume. We resolve these discrepancies by constructing the hand surface geometry (the skin) from the MRI scan. This is feasible for the neutral shape because the fingers are

well-separated. Note that due to limited MRI scanning resolution, this mesh has significantly lower quality than $\hat{\mathscr{T}}_{\text{outer}}$, and cannot serve directly as an input mesh for tetrahedral meshing. Instead, we align $\hat{\mathscr{T}}_{\text{outer}}$ onto the MRI-scanned surface mesh, using a nonlinear Iterative Closest Point (ICP) tool Wrap3 [142]. Here, we set the parameters in Wrap3 so that the wrap is nearly rigid and only adjusts the pose, preserving the local surface detail in $\hat{\mathscr{T}}_{\text{outer}}$. This is a slight, but important adjustment, to compensate for subject's ergonomic settling. The result is a triangle mesh $\mathscr{T}_{\text{outer}}$ of the surface of the neutral hand that has good quality and local surface detail, does not suffer from a loss of volume, and is positioned consistently with the MRI-scanned skeleton (see Figure 4.6, top). This mesh will serve as the outer mesh for our simulation tet mesh.

4.5 Generation of the tet mesh inner boundary

We now describe how we generate a triangle mesh $\mathscr{T}_{\text{inner}}$ which will serve as the inner boundary for our tet mesh. This mesh has to conform to the bones; however, it should *not* mesh the space in between the bones at the joints; otherwise, those tets will be immediately pinched as the joints articulate, leading to tet inversions and simulation instabilities. We start with the skeleton rig mesh $\mathscr{B}$ (155,450 triangles; animated in Section 4.2), and use CGAL's isotropic_remeshing function (at 2.5mm) to produce a mesh $\mathscr{B}_{\text{coarse}}$ with 10,408 triangles for the entire skeleton. This mesh cannot serve as $\mathscr{T}_{\text{inner}}$ because the mesh simplification and remeshing sometimes introduce collisions between the bone meshes. To remove collisions, we use the TetWild algorithm [52] to create a tetrahedral mesh of the union of the volumes enclosed by bones in $\mathscr{B}_{\text{coarse}}$. The surface of this tet mesh is our inner mesh $\mathscr{T}_{\text{inner}}$. It is a good-quality isotropic mesh (10,682 triangles) whose surface closely matches $\mathscr{B}_{\text{coarse}}$, except that $\mathscr{T}_{\text{inner}}$ joins ("welds") bone meshes in the collision areas. Note that the vertices of $\mathscr{T}_{\text{inner}}$ are different to those of $\mathscr{B}_{\text{coarse}}$. Here, we considered an alternative: compute mesh boolean union of the bone meshes in $\mathscr{B}_{\text{coarse}}$, and then use CGAL's isotropic_remeshing tool; however, this destroyed some geometric detail and introduced new collisions, so we abandoned this approach.

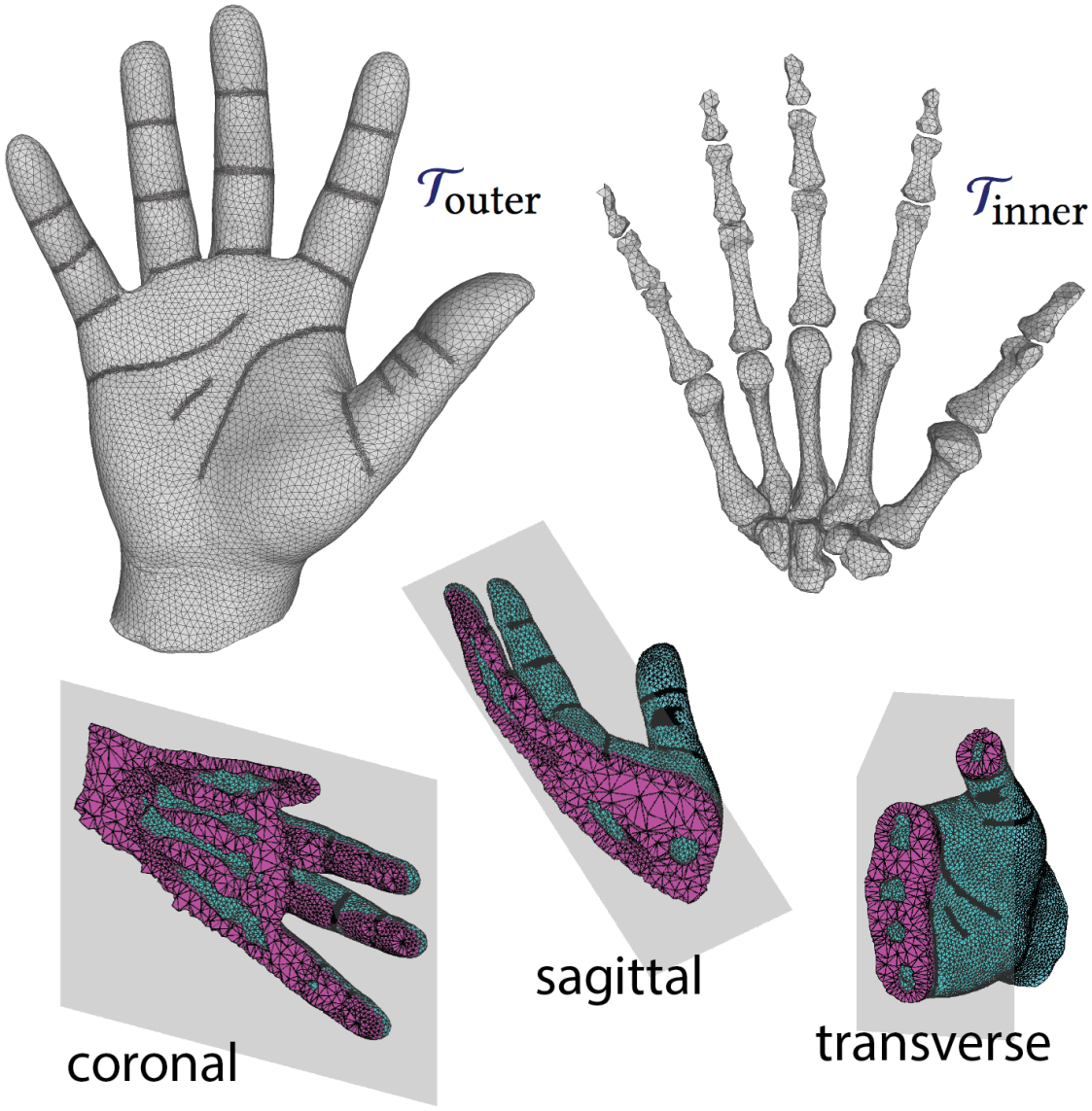


Figure 4.6: **Simulation tet mesh.** Top row: outer and inner tet surface mesh. In $\mathcal{T}_{\text{outer}}$, note the greater resolution at the folds. Bottom: three cutaway views of the FEM mesh.

Next, we execute constrained Delaunay tetrahedralization with refinement (we use TetGen [51]) on input $\mathcal{T}_{\text{outer}} \cup \mathcal{T}_{\text{inner}}$. We set TetGen's parameters so that $\mathcal{T}_{\text{outer}}$ and $\mathcal{T}_{\text{inner}}$ are not refined, but the interior tetrahedra are refined. The running time of TetGen was under 1 second. The resulting tet mesh is not our final tet mesh because we need to remove tetrahedra located at the joints between two bones. For each bone, we manually select the triangles of $\mathcal{B}_{\text{coarse}}$ on the proximal epiphysis (the head closer to the parent bone), roughly corresponding to the region of articular cartilage (smooth white tissue where bone forms a joint to its parent bone); call this set $\mathcal{S}_{\text{child}}$. Next,

we execute a procedure that we call "sweeping": for each vertex of a triangle from $\mathscr{S}_{\text{child}}$, find the closest triangle t on the parent bone. If the distance is less than a threshold (we use 2.5mm), and the normals are approximately opposite (dot product is smaller than -0.7), we add t to the swept set $\mathscr{S}_{\text{parent}}$. We then form the convex hull (using CGAL's convex_hull_3) of $\mathscr{S}_{\text{child}} \cup \mathscr{S}_{\text{parent}}$. We then remove from the tet mesh every tet whose center is inside the convex hull (see Figure 4.7, top-middle). This gives us our simulation tet mesh (101,941 tets, 29,583 vertices). Note that this mesh has an external surface and an internal surface (i.e., there are internal voids corresponding to the bones).

4.6 Constraining the tet mesh to the bones

Our tet mesh is attached to the bones using soft constraints. We constrain a subset of the vertices of the internal tet mesh surface. Constrained vertices are determined by sweeping (in the sense of Section 4.5) the child bone against the parent bone, across the entire range of motion of the joint. Note that hand joints may have 1-DOF or 2-DOFs. Our sweep resolution is 1 degree for each joint degree of freedom, both for 1-DOF and 2-DOF joints. For each value of the joint degrees of freedom, we compute $\mathscr{S}_{\text{parent}}$. We form the union $\mathscr{U}$ of $\mathscr{S}_{\text{parent}}$ across the entire range of motion (Figure 4.7, bottom-right). The sweeping takes 10 minutes total for the entire skeleton. We then traverse all tet mesh vertices that are on the internal surface. If the closest triangle on $\mathscr{B}_{\text{coarse}}$ is further than 0.01mm, or is in $\mathscr{U}$, we do not constrain this vertex, otherwise we constrain it to the closest location on $\mathscr{B}_{\text{coarse}}$, using a spring of stiffness 10 N/mm. This avoids constraining $\mathscr{U}$, and the region on the tet mesh adjacent to the removed tets in between the bones. The constraint springs are integrated using implicit integration by computing their force gradients, for simulation stability. We compute the motion of the coarse bone mesh $\mathscr{B}_{\text{coarse}}$ by skinning it to the output mesh $\mathscr{B}$ of our bone rig; each vertex of $\mathscr{B}_{\text{coarse}}$ is skinned to the closest bone in $\mathscr{B}$. Because nails are rigid, we need to also constrain nail vertices. We do this manually, by painting the nail triangles in Maya. The nail vertices are constrained to follow the transformation of the distal phalanx finger bones. In addition

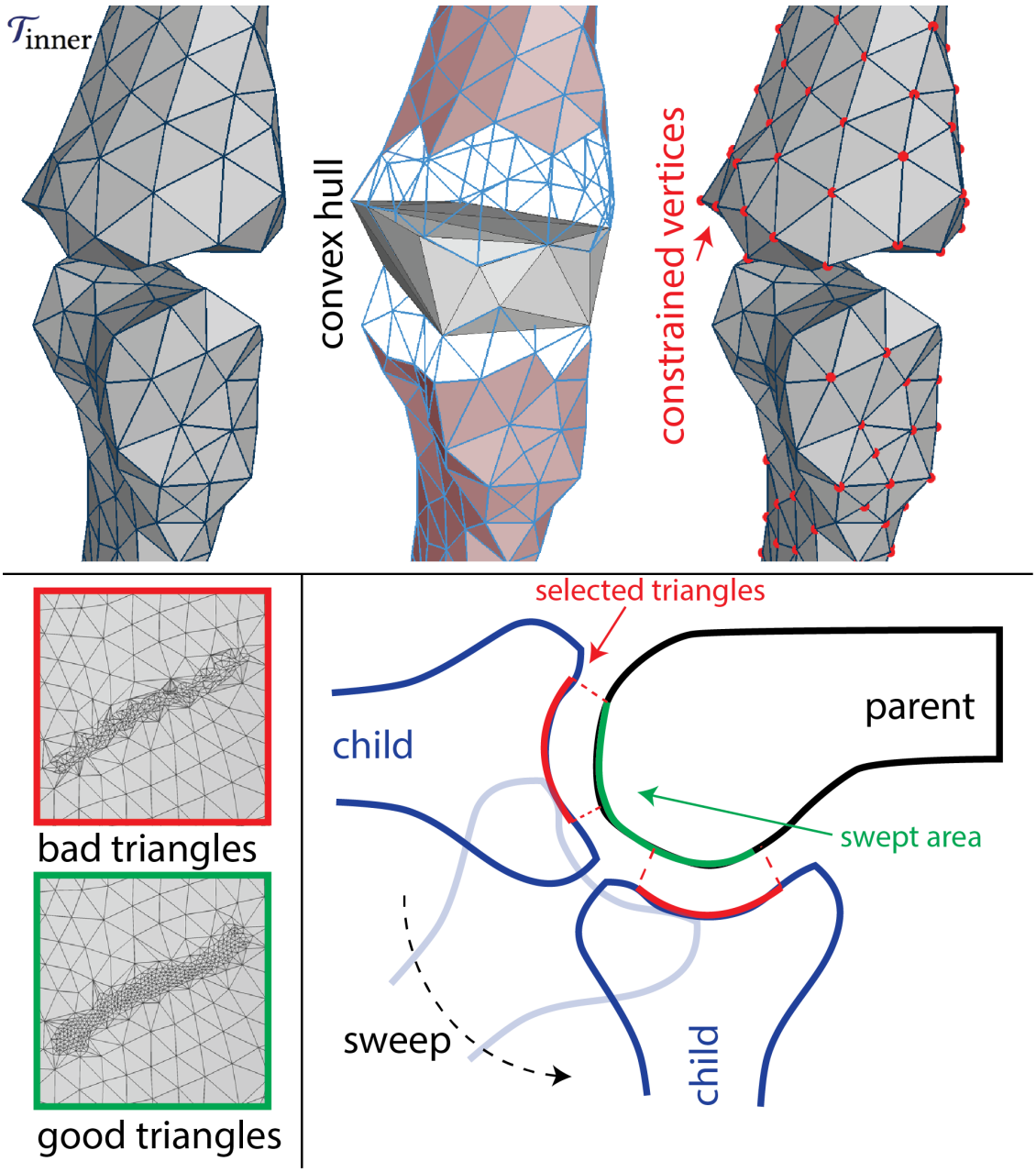


Figure 4.7: **Tet mesh creation.** Top: mesh $\mathcal{T}_{\text{inner}}$; removing tets at the joints; tet mesh constrained vertices. Bottom-left: the two-step remeshing process at the folds produces good triangles and is necessary: under a single-step process, bad triangles appear. Bottom-right: illustration of sweeping in 2D.

to the constraints, we also apply collision response between the bones and the soft tissue, and soft tissue self-collision response. For the former, we use the technique presented in [13], whereas for the latter, we detect pairs of non-neighboring overlapping triangles and push them apart using penalty springs. In one of our examples (connect-all-poses), we observed small spurious wrinkles

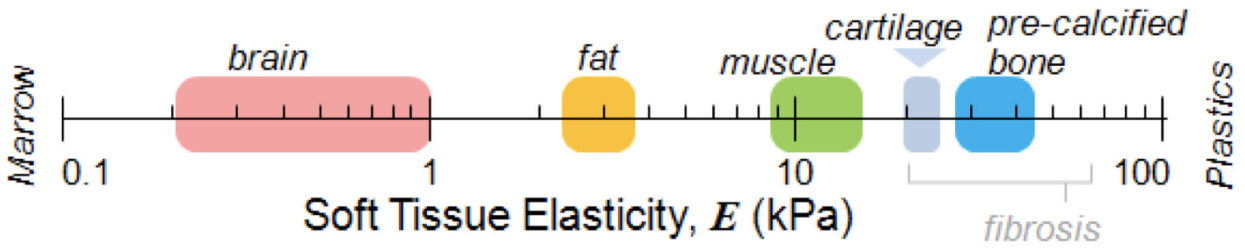


Figure 4.8: **Young's modulus of various human musculoskeletal tissues.** Note the unit $kPa = 1000N/m^2$. From [36]. Reprinted with permission from the AAAS.

on the skin at the dorsal (i.e., back) side of the PIP joint, when the PIP joint is bent to its extreme. We attribute this to the fact that the skin is pre-folded in that region in the neutral configuration, and "unfolds" as PIP articulates; pre-folding is not modeled by our system. We remedy this issue by post-processing the small local region using the "Delta Mush" deformer [80].

4.7 Material properties

For elastic material properties, we use the Saint-Venant Kirchhoff material. We add a volume preservation term which prevents large compressions and successfully guards against collapse. Specifically, we model volume preservation by adding a term $\lambda(J-1)^2/2$ to the elastic strain energy density function. Here, λ is the first Lamé parameter, and J is the local volume growth factor (product of the three principal stretches [57]). The combination of StVK and volume preservation produced organic hand shapes that visually match photographs of the subject's hand (Figure 4.13). We model our tissue as a nearly incompressible isotropic hyperelastic solid, and therefore use a Poisson's ratio of 0.495. Although biological tissues are not completely incompressible, incompressibility and isotropy is appropriate for many biological materials [139], because human tissue largely consists of an incompressible liquid (water). Our choice of Young's modulus is motivated by experimental measurements available in literature, such as Figure 4.8 reproduced from [36], which the authors assembled from various experimental studies. As can be seen, fat (2500-4000N/m^2) is softer than muscles (9000-15000N/m^2). Because we do not model fat and

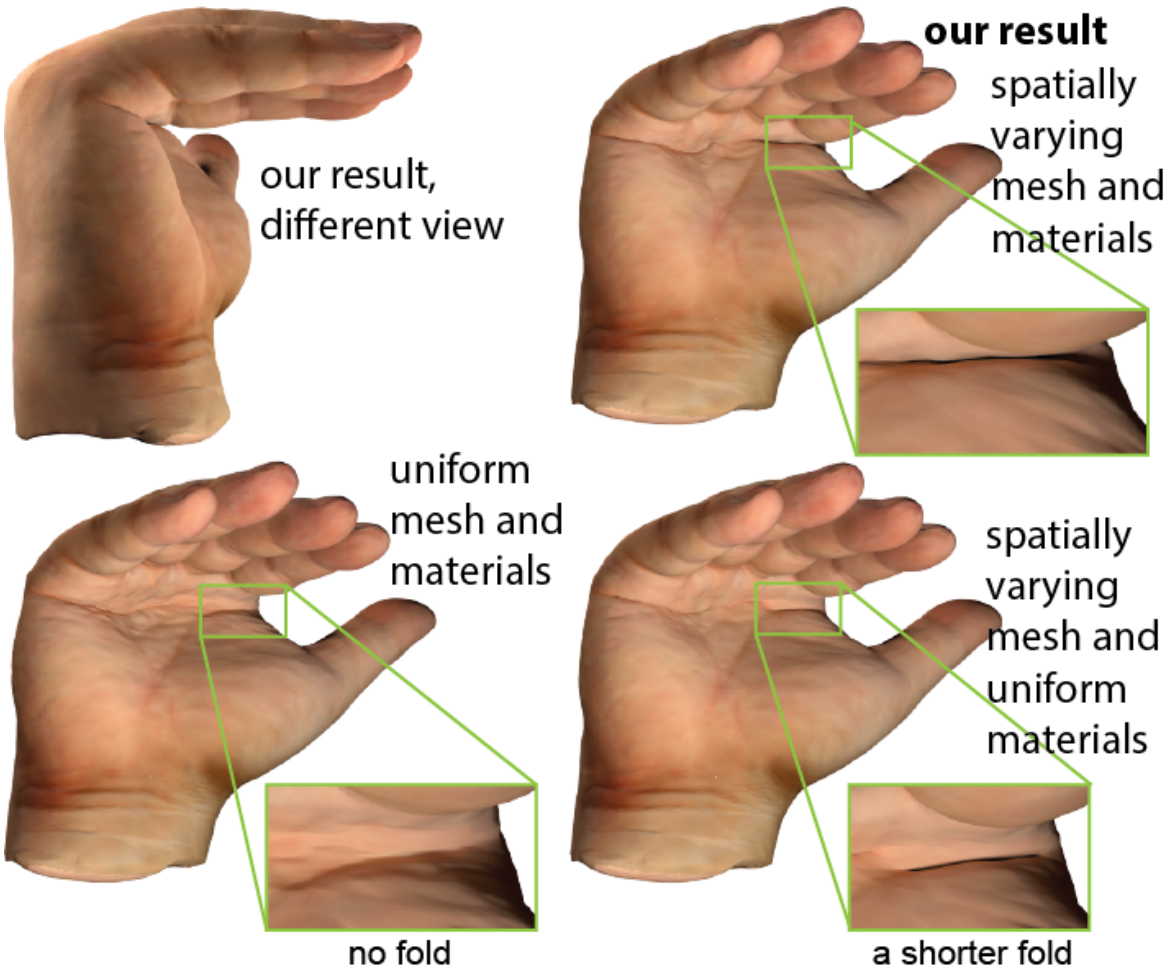


Figure 4.9: **Spatially varying mesh resolution and materials.** Fold formation is improved by increasing the tet mesh resolution under the finger joints (via $\mathscr{T}_{\text{outer}}$; Sections 4.3 and 4.4), and by making the material in the same region 6x softer (Section 4.7). Our method produces a quality horizontal fold across the entire length of the palm. Other methods only produce a partial fold, or no fold at all.

muscles as separate objects, we set our Young's modulus to a single value of $12000 N/m^2$. In locations where there are visible hand lines, such as on the palm and at finger joins, we decrease Young's modulus to $2000 N/m^2$, and Poisson's ratio to 0.45, which helps with crease formation. Because there are no muscles on the dorsal side of the hand, we decrease Young's modulus in that region to $6000 N/m^2$, including on the back side of the fingers. For mass density, we choose the density of water ($1000 kg/m^3$). This is because the reported densities of muscles and fat are $1060 kg/m^3$ and $920 kg/m^3$, respectively [39].

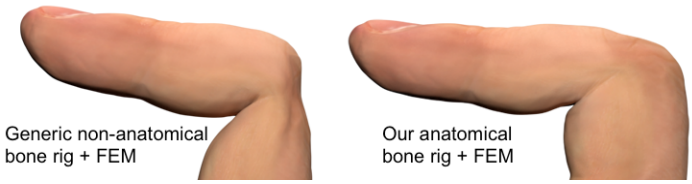


Figure 4.10: **Anatomical bone rig stabilizes FEM soft tissue simulations.** Left: non-anatomical rig, created by pivoting bones around the center of the parent bone's head. Bones pinch elastic material, causing artefacts. Right: our rig produces stable soft tissue simulations. Both results use FEM.

4.8 Embedding of surface triangle meshes into tet mesh

We deform any of our surface triangle meshes by embedding it into the tet simulation mesh. Sometimes, some vertices of these surface triangle meshes in the neutral pose may be slightly outside of the tet mesh due to coarsening or discretization errors; in which case we simply deform the vertex using the closest tetrahedron. There are two surface meshes that we deform in this way and that appear in our rendering results: the high-resolution mesh obtained by scanning the neutral plastic hand using Artec Spider (11M triangles), and a low-resolution version with 15,232 triangles, obtained by simplifying the high-resolution mesh using MeshLab.

4.9 Results

We used a 3 Tesla GE MRI scanner with a PD CUBE (3D fast spin echo) sequence with the following parameters: NEX (number of excitations) 1; FA (flip angle) 90 degrees; TR (time to repetition) 1500 ms; TE (time to echo) 25.93 ms; slice thickness 1 mm; slice spacing 0.5 mm; matrix 256 x 256 pixels; and FOV (field of view) 28.3 x 25.6 cm. We scanned, processed and simulated two subjects: a male (age: late 20s; Figure 3.1), and a female (age: late 40s; Figure 4.12).

We tested our bone rig and FEM simulation on 4 motion sequences. Two sequences were recorded using a LeapMotion system [72], one was created using Inverse Kinematics (IK) to perform the opposition of the thumb to the other fingers, and one connects all of our scanned poses into one continuous sequence, also using IK. We implemented a simple inverse kinematics engine,

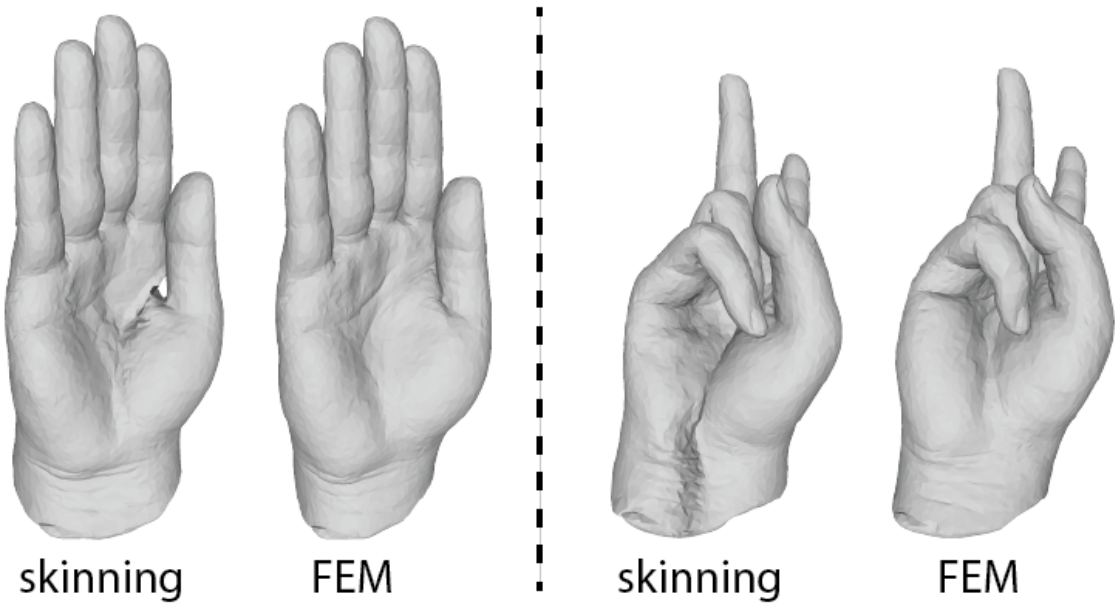


Figure 4.11: **Comparison of FEM simulation to skinning.** Skinning weights were computed using Maya's geodesic voxel method. Other Maya skinning weight methods yield even worse results. Both methods use our bone rig.

using the ADOL-C symbolic differentiation library. In our IK, we can constrain both positions and orientations of the end effectors; and we did so for the fingers subject to biological limits. The running time of our FEM simulations is approximately 1 sec / timestep, on an Intel i7 6950X PC (manufactured in 2016, 1 processors x 10 cores at 3.00 GHz), 128GB of RAM. We are performing 60 timesteps per frame, at 25 FPS; and therefore, the computational cost is 1 minute per frame, or 25 minutes per second of motion. We do not attempt to simulate accurate tissue dynamics, only static shapes under the animated bone meshes. Hence mass, damping and timestep properties only matter for simulation stability. The four sequences (recorded motions 1, 2; thumb opposition, connect-all-poses) are 13, 16, 10 and 16 seconds long, and took 5.4, 6.7, 4.2 and 6.7 hours to compute, respectively.

Our pipeline produces a quality un-colored surface mesh consistent with internal anatomy. In order to add a color texture, we took photos of the subject's hand from approximately 60 different angles using a DSLR camera (Canon EOS 7). The photos were taken rapidly while the subject held the hand firmly against a table. The subject then reversed the hand and another set of 60 photos was taken. We then constructed two separate textured surfaces from these photographs

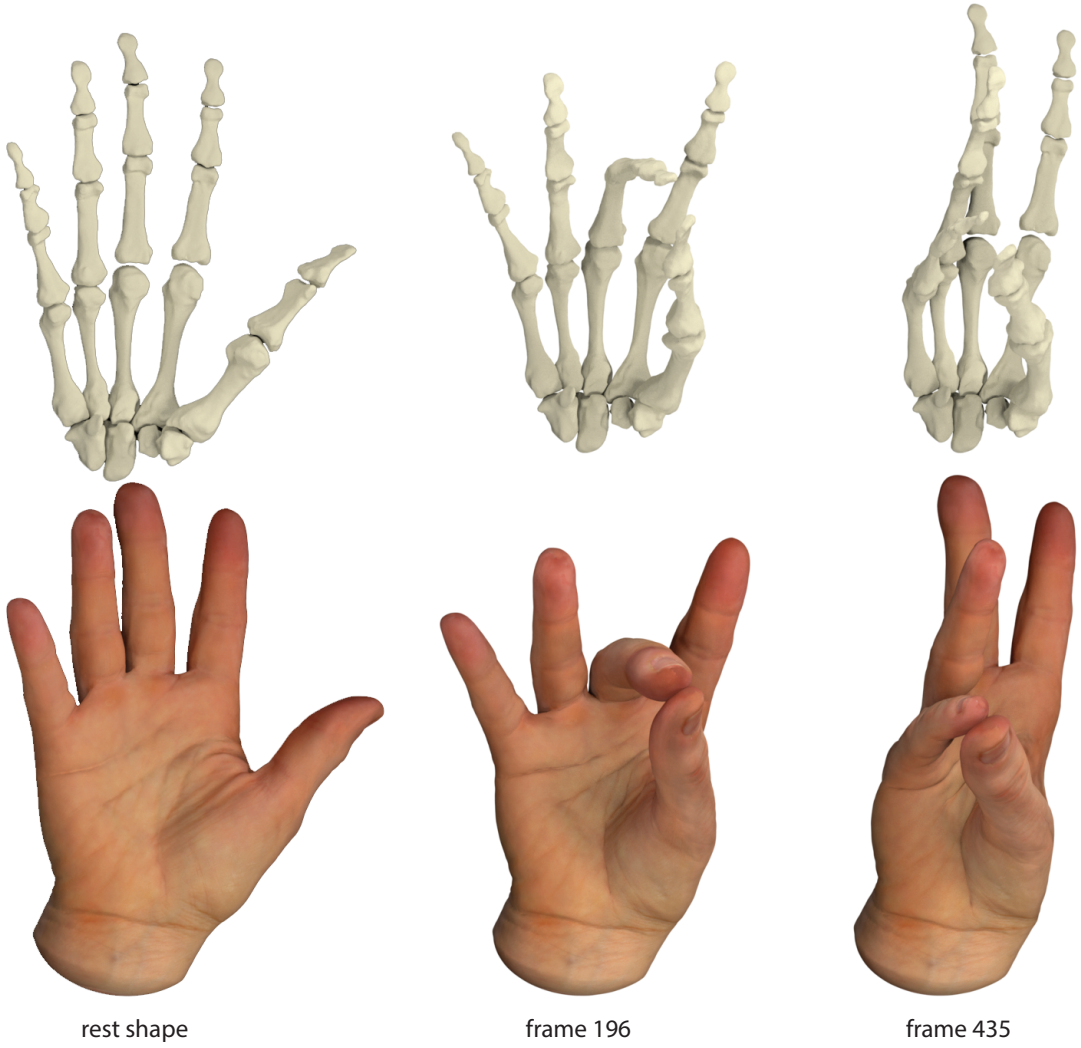


Figure 4.12: **The opposition of the thumb for our female subject (late 40s).** The bone and FEM geometry of three animation frames.

using the Photoscan software [2]. We then separately wrapped each of these two surfaces onto our un-colored low-resolution surface mesh, using Wrap3. At the seams, the surfaces and colors overlap because each of the two half-scans covers slightly more than one half of the hand. We adjusted and smoothed the seams using Photoshop and Maya. We then transferred the texture onto the high-resolution mesh, using Maya. This produced a good quality color texture map for both low-resolution and high-resolution surface meshes, which we then use for rendering (Figure 4.14). In this project, we discovered that it is actually not easy to obtain a complete hand static model with a high-resolution geometry and texture. Our Photoscan approach benefited from the fact that

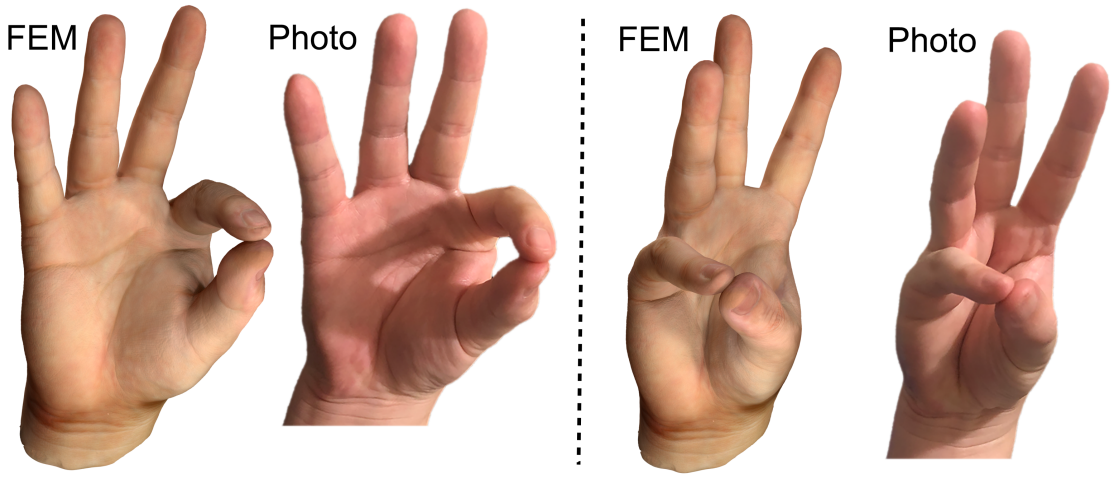


Figure 4.13: **Comparison of our FEM results to photographs**, on two "in-between" poses (i.e., not a part of the acquired 12 pose-dataset). The FEM results are rendered using our high-resolution mesh (11M triangles) with subsurface scattering; individual pores are visible as geometric features. The image can be zoomed in on. Generally, our FEM method produces a good qualitative visual match to the photographs: the finger and palm appear "organic", and the two main palmar lines are in the correct location and are produced by FEM as geometric mesh features (folds), as opposed to only a texture map illusion. This figure also demonstrates the limitations of our method. We do not model the wrinkling of the skin, which is absent in the FEM result, but visible in photographs. We also do not simulate muscle activation and, hence, the soft tissue at the root of the thumb is flatter in FEM vs. in photographs.

we already obtained high-precision geometry by scanning the (perfectly still) plastic hand using Artec Spider, and only needed to add color texture. Before using Photoscan, we consulted a local professional photogrammetry studio who attempted to create both geometry and color texture. They failed to produce a good scan, despite taking over 60 simultaneous photographs of the subject's hand with high-quality cameras from multiple angles. Because a live hand is moving, Artec Spider also failed to simultaneously create geometry and color texture, despite repeated scanning attempts.

We compare our anatomy-driven bone rig to a non-anatomical bone rig created in Maya, by placing the joint pivots at the end of each bone, and then running FEM soft tissue simulation. Our rig clearly outperforms the non-anatomical rig, where bone collisions artificially squeeze the elastic material, causing instabilities in FEM simulations (Figure 4.10). We also compare our FEM method to skinning (Figure 4.11). Both FEM and skinning used our bone rig. The comparison

Figure 4.14: **Our bone rig also improves skinning.** Left: skinning with a non-anatomical bone rig, created manually by placing the joints at the center of the parent bone's bonehead. Right: skinning with our anatomical rig. No FEM simulation was used in this example, only skinning. It can be seen that skinning works better when it uses our anatomical bone rig (see the dent at the joint). Both results use Maya's "geodesic voxel" skinning weights.

shows that FEM produces higher quality shapes. Figure 4.9 demonstrates that our spatially varying materials enable the generation of quality folds near the finger joints. In Figure 4.13, we compare our FEM result to photographs of the hand of the same subject. Figure 4.14 demonstrates that our bone rig also improves the quality of skinning (i.e., without using FEM). Table 4.1 analyzed the accuracy of our bone rig.

4.10 Discussion and conclusion

We demonstrated how to acquire highly accurate human skeleton geometry in multiple poses using MRI. We achieved this using a novel molding procedure to stabilize hands during MRI scanning. We registered all poses to the same bones mesh connectivity, and built a skeleton mesh kinematic model ("skeleton rig") to interpolate or extrapolate the acquired pose to the entire hand range of motion. We demonstrated that our skeleton rig can be used to drive soft-tissue FEM simulation to produce anatomically plausible organic hand shapes that qualitatively match photographs. Our accurate hand model can potentially benefit virtual hands in games / film / VR, robotic hands, grasping, and medical education, such as visualizations of internal hand anatomy motion.

We performed 12 scans for each subject, which is sufficient to demonstrate stable and precise common hand motions, including opposition of the thumb to the other four fingers. Accuracy would be improved with more scans. An interesting research questions is how to automatically select the next best pose to scan, to maximize coverage of the hand's range of motion. Some

complex hand poses are challenging for our technique, for example, all fingertips touching at a single point. This is because we cut our molds in two pieces manually using a knife. It would be interesting to explore how to cut the molds into 3 or more pieces, to improve the ergonomics of the insertion of the hand into the mold. Although we produced and scanned 12 plastic shapes, we only used the neutral plastic shape in our simulation pipeline; the other 11 serve to evaluate the accuracy of our FEM rig (Figure 3.1). It is challenging to register the scanned plastic meshes to consistent complete mesh connectivity because many of these optical scans are incomplete due to palm / finger occlusions, and webbing of fingers in close proximity.

We only acquired bone geometry so far; muscles, tendons and subcatenous fat could potentially extracted from MRI too. While our FEM simulations (driven by our skeleton rig) produce anatomically plausible shapes, accuracy would be improved by modeling muscles and subcatenous fat. These structures are available in our MRI scans and could be in principle segmented and animated; These problems are addressed in the later chapters. We use spatially-varying tissue elastic properties to improve the generation of folds at the joints. Results could be further improved by optimizing spatially varying material properties. Physically based grasping would also benefit from good material properties. We do not compute the joint hierarchy motion; but instead rely on the animator to provide us with the animations of the bone's joints. Such an approach is standard in industry today [121], but requires the animator to be careful not to cause self-collisions. A better, more principled approach would be for motion to originate from hand muscle activations, with bones and the rest of the anatomy animating based on a physically based simulation. This poses interesting neuromechanical control problems, due to the well-known muscle redundancy, and the specific nature of hands which are mostly controlled by long muscle tendons originating in the human arm. The understanding of motion of internal hand anatomy as proposed here represents one step closer to lifelike robots with hands that mimic biological hands. Researchers at Columbia Univ. recently demonstrated how to make artificial muscles [87]. With recent advances in 3D printing technology, our work may contribute to the physical replication of anatomically realistic hands and improved hand prosthetics, in the not so distant future.

Chapter 5

Extracting Anatomy using Plastic Strains

In contrast to hand bones, the boundaries of other organs are often blurry. For example, in an MRI of a human hand, the muscles often "blend" into each other and into fat without clear boundaries; a CT scan has even less contrast. Thus, the approach used for segmenting the bones is not helpful. In this chapter, we want to tackle the problem of how to extract those tissues from volumetric 3D medical imaging (such as MRI or CT scan). To represent such blurry boundaries, we therefore manually select as many reliable points ("markers") as possible on the boundary of the organ in the medical image; some with correspondence ("landmark constraints") to the same anatomical landmark in the template organ, and some without ("ICP constraints"). Given a template volumetric mesh of an organ of a generic individual, a medical image of the same organ of a new individual, and a set of landmark and ICP (Iterative Closest Point) constraints, our method asks how to deform the template mesh to match the medical image.

Given a template tet mesh of a soft tissue organ for a generic individual, as well as known optional attachments of the organ to other objects, our goal is to deform the tet mesh to match a medical image of the organ of a new individual. We use the term "medical image" everywhere in this thesis because this is standard terminology; this does not refer to an actual 2D image, but to the 3D medical volume. We now describe how we mathematically model the attachments and medical image constraints.

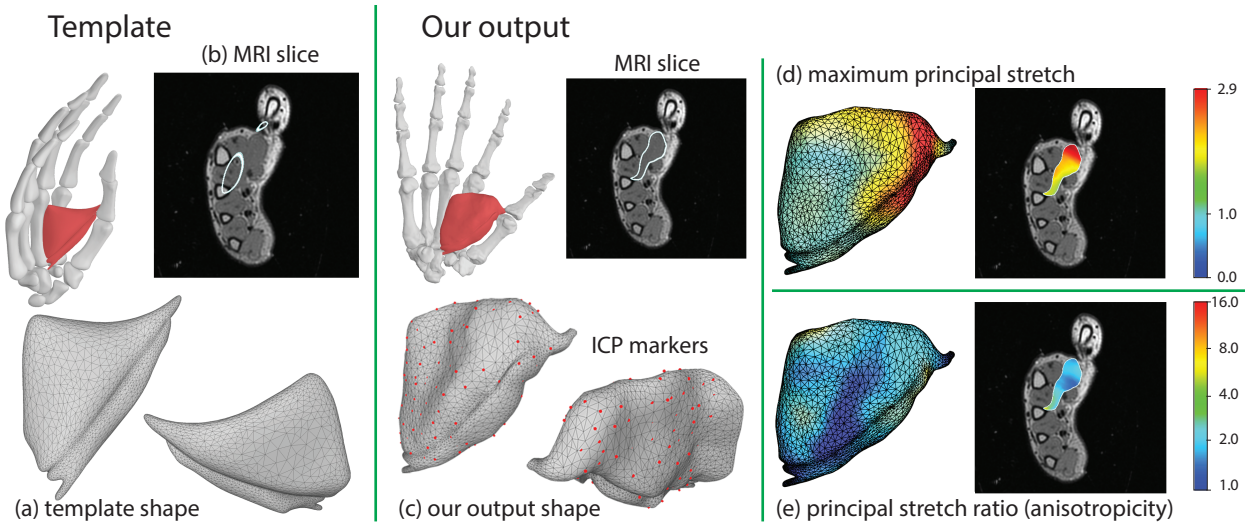


Figure 5.1: **We optimized the shape of this hand muscle (Adductor Pollicis) to match the MRI scan.** (a) Template shape [26]; (b) representative MRI slice [134]; we rigidly aligned the template mesh onto the markers in the MRI scan, producing the white contour that is obviously incorrect (deeply penetrates the bone and extends out of the volume of the MRI-scanned hand); (c) our output shape, optimized to the MRI scan; the white contour now matches the scan; (d, e) large anisotropic spatially varying strains accommodated by our method (d: maximum principal stretch; e: ratio between maximum and minimum principal stretch).

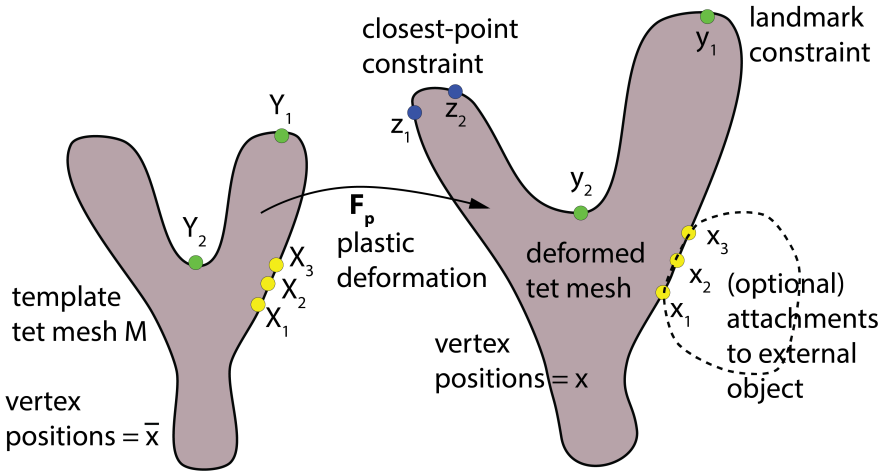


Figure 5.2: **Our shape deformation setup.** Our optimization discovers plastic strains F_p and vertex positions x so that the model is in elastic equilibrium under the attachments while meeting the medical image landmark and closest point constraints as closely as possible. The presence of attachments is optional; our work address both attached and unattached objects.

5.1 Attachments and medical image constraints

We start with a template organ tet mesh $\mathscr{M}$. We denote its vertex positions by $\bar{\mathbf{x}} \in \mathbb{R}^{3n}$, where n is the number of tet mesh vertices. In this thesis, we use **bold** text to represent the quantities for the entire mesh and non-bold text to represent the quantities for a single vertex or element in the FEM mesh. We would like to discover vertex positions $\mathbf{x} \in \mathbb{R}^{3n}$ such that the organ shape obeys the attachments to other organs (if they exist), and of course the medical image. The attachments are modeled by known material positions $X_i \in \mathscr{M}$, $i = 1, \ldots, t$ that have to be positioned at known world-coordinate positions $x_i \in \mathbb{R}^3, i = 1, \ldots t$ (Figure 5.2). The medical image constraints come in two flavors. First, there are *landmark constraints* whereby a point on a template organ is manually corresponded to a point in the medical image, based on anatomy knowledge. Namely, landmark constraints are modeled as material positions $Y_i \in \mathscr{M}$, $i = 1, \ldots, q$, that are located at known world-coordinate positions $y_i \in \mathbb{R}^3$ in the medical image. Observe that landmark constraints are mathematically similar to attachments. However, they have a different physical origin: attachments are a physical constraint that is pulling the real-world organ to a known location on another (fixed, un-optimized) object; for example, a muscle is attached to a bone. With landmarks, there is no such physical force in the real-world; namely, landmarks (and also closest-point constraints) are just medical image observations.

The second type of medical image constraints are *closest-point constraints* ("ICP markers"). They are given by known world-coordinate positions $z_i \in \mathbb{R}^3, i = 1, \ldots r$ that have to lie on the surface of the deformed tet mesh. Locations z_i are easier to select in the medical image than the landmarks because there is no need to give any correspondence. As such, they require little or no medical knowledge, and can be easily selected in large numbers. We went through the medical image slices and selected clear representative points on the organ boundary. We then visually compared the template and the target shape inferred by the ICP marker cloud. This guided our positioning of the landmarks, which we place on anatomically "equal" positions in the template and the medical image. We consulted a medical doctor to help us interpret medical images, such

as identifying muscles in the scan, clarify ambiguous muscle boundaries, placing attachments, and disambiguating tendons.

5.2 Plastic deformation gradients

We model shape deformation using plastic deformation gradients, combined with a (small) amount of elastic deformation. In solid mechanics, plasticity is the tool to model large shape variations of objects, making it very suitable for shape deformation with large strains. Unlike using the elastic energy directly (without plasticity), plastic deformations have the advantage that they can arbitrarily and spatially non-uniformly and anisotropically scale the object. There is also no mathematical requirement that they need to respect volume preservation constraints. This makes plastic deformations a powerful tool to model shapes. Our key idea is to find a plastic deformation gradient F_p at each tet of $\mathscr{M}$, such that the FEM equilibrium shape under $\mathbf{F_p}$ and any attachments matches the medical image observations. Figure 5.2 illustrates our shape deformation setting. In order to do so, we need to discuss the elastic energy and forces in the presence of plastic deformations, which we do next.

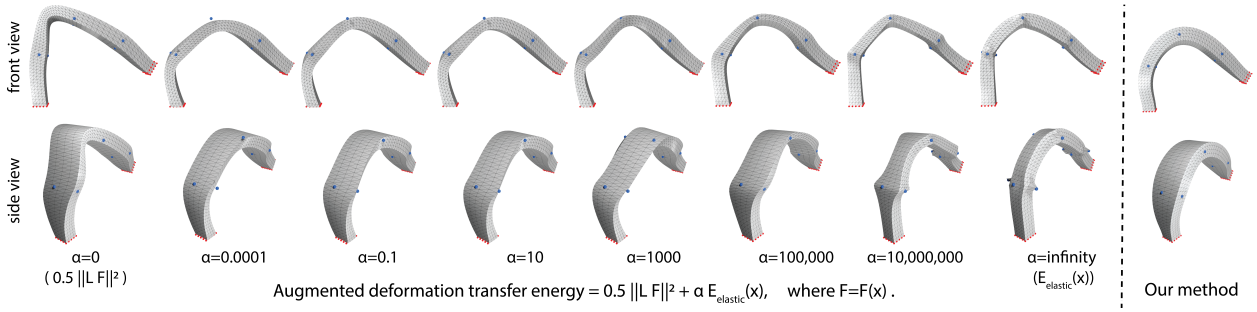


Figure 5.3: **Comparison to augmented deformation transfer.** The beam's attachments (red) cause the beam to bend, whereas the ICP markers (blue) cause it to stretch 2x in one of the two transverse directions. Our method can easily recover such a shape deformation, whereas deformation transfer [118] cannot, even if augmented with an elastic energy.

Plastic strain is given by a 3×3 matrix F_p at each tetrahedron of $\mathscr{M}$. For each specific deformed shape $\mathbf{x} \in \mathbb{R}^{3n}$, one can define and compute the deformation gradient F between $\overline{\mathbf{x}}$ and $\mathbf{x}$

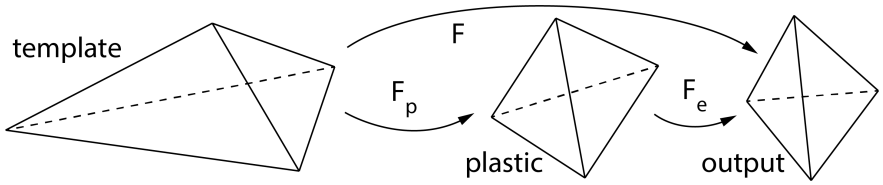


Figure 5.4: **Plastic and elastic deformation gradient for a single tet.**

at each tet [89]. The elastic deformation gradient F_e can then be defined as $F_e = FF_p^{-1}$ [15] (see Figure 5.4). Observe that for any shape $\mathbf{x}$, there exists a corresponding plastic deformation gradient $\mathbf{F_p}$ such that $\mathbf{x}$ is the elastic equilibrium under $\mathbf{F_p}$; namely $\mathbf{F_p} = \mathbf{F}$. This means that the space of all plastic deformation gradients $\mathbf{F_p}$ is expressive enough to capture all shapes $\mathbf{x}$. The elastic energy of a single tet is defined as

$$\mathscr{E}(F_p, x) = V(F_p)\psi(x, F_p) = V(F_p)\psi\big(F(x)F_p^{-1}\big), \tag{5.1}$$

where V is the rest volume of the tet under the plastic deformation F_p, and ψ is the elastic energy density function. We have $V = |F_p|V_0$, where V_0 is the tet's volume in $\mathscr{M}$, and $|F_p|$ is the determinant of the matrix F_p. Elastic forces equal $\mathbf{f}_e(\mathbf{F_p}, \mathbf{x}) = d\mathscr{E}(\mathbf{F_p}, \mathbf{x})/d\mathbf{x}$. When solving our optimization problem to compute $\mathbf{F_p}$ in Section 5.4, we will need the first and second derivatives of $\mathscr{E}(F_p, x)$ with respect to x and F_p. We provide their complete derivation in Appendix B.

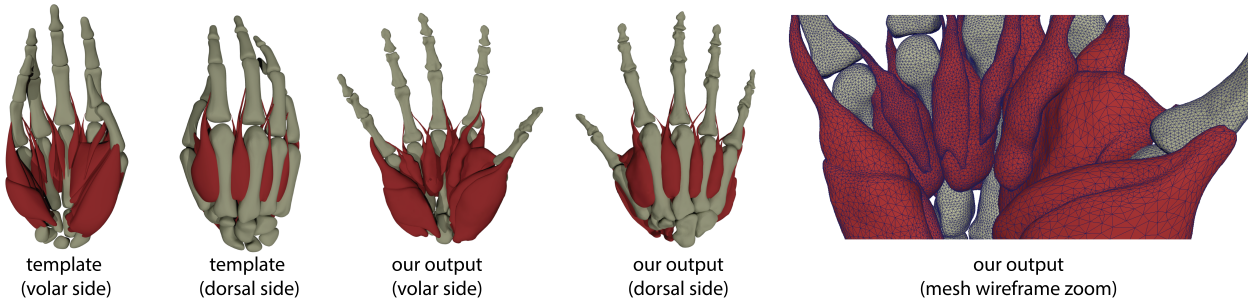


Figure 5.5: **Seventeen muscles of the human hand extracted from MRI.** Observe that the template hand is larger than the scanned hand. The pose is also different. Our method solves this using bone attachments.

Our method supports any isotropic hyperelastic energy density function ψ. In our examples, we use the isotropic stable neo-Hookean elastic energy [110], because we found it to be stable and sufficient for our examples. Note that we do model anisotropic plastic strains (and this is crucial for our method), so that our models can stretch by different amounts in different directions. Observe that plastic strains are only determined up to a rotation. Namely, let F_p be a plastic strain (we assume $\det(F_p) > 0$; i.e., no mesh inversions), and $F_p = QS$ be the polar decomposition where Q is a rotation and S a 3×3 *symmetric* matrix. Then, F_p and S are the "same" plastic strain: the resulting elastic deformation gradients differ only by a rotation, and hence, due to isotropy of ψ, produce the same elastic energy and elastic forces. Note that it is not required that rotations Q match in any way at adjacent tets. We do not need to even guarantee that F_p globally correspond to any specific "rest shape", i.e., the F_p are independent of each other and may be inconsistent. This gives plastic deformation gradient modeling a lot of flexibility. Hence, it is sufficient to model plastic strains as symmetric 3×3 matrices. We can therefore model F_p as a symmetric matrix and parameterize it using a vector $s \in \mathbb{R}^6$,

$$F_p = \begin{bmatrix} s_1 & s_2 & s_3 \\ s_2 & s_4 & s_5 \\ s_3 & s_5 & s_6 \end{bmatrix}. \tag{5.2}$$

We model plasticity globally using a vector $\mathbf{s} \in \mathbb{R}^{6m}$, where m is the number of tets in $\mathscr{M}$. We note that Ichim et al. [55] used such a 6-dimensional parameterization to model facial muscle activations. In our work, we use it for general large-strain shape modeling. Our application and optimization energies are different, e.g, Ichim et al. [55] causes muscle shapes to follow a prescribed muscle firing field, and biases principal stretches to be close to 1. Furthermore, we address the singularities arising with unattached objects.

5.3 Shape deformation of attached objects

We now formulate our shape deformation problem. We first do so for attached objects. An object is "attached" if there are sufficient attachment forces to remove all six rigid degrees of freedom, which is generally satisfied if there are at least three attached non-colinear vertices. We find the organ's shape that matches the attachment and the medical image constraints by finding a plastic strain F_p at each tet, as well as static equilibrium tet mesh vertex positions $\mathbf{x}$ under the attachments and plastic strain $\mathbf{F_p}$, so that the medical image observations are met as closely as possible,

$$\underset{\mathbf{s},\mathbf{x}}{\arg\min} \quad ||\mathbf{L}\mathbf{s}||^2 + \alpha \mathscr{E}_{\mathrm{MI}}(\mathbf{x}) + \beta \mathscr{E}_{\mathrm{a}}(\mathbf{x}), \tag{5.3}$$

$$\text{subject to:} \quad \mathbf{f}_{\mathrm{e}}\big(\mathbf{F_p}(\mathbf{s}), \mathbf{x}\big) + \mathbf{f}_{\mathrm{a}}(\mathbf{x}) = 0, \tag{5.4}$$

where $\alpha \geq 0$ and $\beta \geq 0$ are scalar trade-off weights, and $\mathbf{L}$ is the *plastic strain Laplacian*. We define $\mathbf{L}$ as essentially the tet-mesh Laplacian operator on the tets, 6-expanded to be able to operate on entries of s at each tet (precise definition is in Appendix A). The Laplacian term enforces the smoothness of $\mathbf{F_p}$, i.e., F_p in adjacent tets should be similar to each other. The second equation enforces the elastic equilibrium of the model under plastic strains $\mathbf{F_p}$ and under the attachment forces $\mathbf{f_a}$. Attachments pull a material point embedded into a tet to a fixed world-coordinate location using spring forces; precise definitions of the attachment energy $\mathscr{E}_{\mathrm{a}}$ and attachment forces $\mathbf{f_a}$ are in Appendix F. It is important to emphasize that our output shapes are always in static equilibrium under the plastic strains $\mathbf{F_p}$, and both this equilibrium shape $\mathbf{x}$ and $\mathbf{F_p}$ are optimized together; this is the key aspect of our work. The first equation contains the smoothness and the medical image (MI) observations; we discuss the rationale for using the attachment energy $\mathscr{E}_{\mathrm{a}}$ in Equation 5.3 in the next paragraph. The medical image energy measures how closely $\mathbf{x}$ matches the medical image constraints,

$$\mathscr{E}_{\mathrm{MI}}(\mathbf{x}) = \sum_{i=1}^{q} ||S\mathbf{x} - y_i||^2 + \sum_{i=1}^{r} ||z_i - \mathrm{closestPoint}(\mathbf{x}, z_i)||^2, \tag{5.5}$$

where S is the interpolation matrix that selects Y_i, namely $S\bar{x} = Y$. The function closestPoint$(\mathbf{x}, z_i) \in \mathbb{R}^3$ computes the closest point to $z_i \in \mathbb{R}^3$ on the surface of the tet mesh with vertex positions $\mathbf{x}$.

Our treatment of attachments in Equations 5.3 and 5.4 deserves a special notice. Equation 5.4 is consistent with our setup: we are trying to explain the medical images by saying that the organ has undergone a plastic deformation due to the variation between the template and captured individual. The shape observed in the medical image is due to this plastic deformation and the attachments. We formulate attachment forces in Equation 5.4 as a "soft" constraint, i.e., $\mathbf{f_a}(\mathbf{x})$ is modeled as (relatively stiff) springs pulling the attached organ points to their position on the external object. This soft constraint could in principle be replaced for a hard constraint where the attached positions are enforced exactly. We use soft constraints in our examples because they provide additional control to balance attachments against medical image landmarks and ICP markers. These inputs are always somewhat inconsistent because it is impossible to place them at perfectly correct anatomical locations, due to medical imaging errors. Hence, it is useful to have some leeway in adjusting the trade-off between satisfying each constraint type. With soft constraints, it is important to keep the spring coefficient in $\mathbf{f_a}(\mathbf{x})$ high so that constraints are met very closely (under 0.5 mm error in our examples). We were able to do this using some small amount of spring coefficient tuning.

As per the attachment energy $\mathscr{E}_\text{a}$, we initially tried solving the optimization problem of Equations 5.3 and 5.4 without it. This seems natural, but actually did not work. Namely, without $\mathscr{E}_\text{a}$, there is nothing in Equations 5.3 and 5.4 that forces the plastic strains to reasonable values. The optimizer is free to set $\mathbf{F_p}$ to arbitrarily extreme values, and then find a static equilibrium $\mathbf{x}$ under the attachment forces. In our outputs, we would see smooth nearly tet-collapsing plastic strains that result in a static equilibrium $\mathbf{x}$ whereby the medical image constraints were nearly perfectly satisfied. Obviously, this is not a desired outcome. Our first idea was to add a term that penalizes the elastic energy $\mathscr{E}(\mathbf{F_p}, \mathbf{x})$ to Equation 5.3. Although this worked in simple cases, it makes the expression in Equation 5.3 generally nonlinear. Instead, we opted for a simpler and more easily computable alternative, namely add the elastic spring energy of all attachments, $\mathscr{E}_\text{a}$. This keeps the expression in Equation 5.3 quadratic in $\mathbf{x}$ and $\mathbf{F_p}$, which we exploit in Section 5.4 for speed.

Observe that $\mathscr{E}_{\text{a}}$ behaves similarly to the elastic energy: if the plastic strain causes a rest shape that is far from the attachment targets, then both $\mathscr{E}_{\text{a}}$ and the elastic energy will need to "work" to bring the shape x to its target attachments. Similarly, if the plastic strain already did most of the work and brought the organ close to its target, then neither $\mathscr{E}_{\text{a}}$ nor the elastic energy will need to activate much.

Because our units are meters and we aim to satisfy constraints closely, we typically use weights close to $\alpha = 10^9$ and $\beta = 10^8$ in our examples. The weights α and β permit adjusting the trade-off between three desiderata: make plastic strains smooth, meet medical image observations, and avoid using too much elastic energy (i.e., prefer to resolve shapes with plastic strains).

Finally, we note that our formulation is different to approaches that optimize the deformation gradient F directly (i.e., without an intermediary quantity such as the plastic deformation gradient). In Figures 5.3 and 5.6, we compare to two such approaches: deformation transfer [118] and variational shape modeling [21]. We demonstrate that our method better captures shapes defined using our inputs (landmarks, ICP markers, large spatially varying strains). Among all compared approaches, the variational method in Figure 5.6 came closest to meeting our constraints, but there is still a visual difference to our method. We provide a further comparison to variational methods in Section 5.9.5.

5.4 Solving the optimization problem for attached objects

We adapt the Gauss-Newton method [109] to efficiently solve the optimization problem of Equations 5.3 and 5.4 (example output shown in Figure 5.5). Although the Gauss-Newton method is commonly used to solve nonlinear optimization problems, the dimension of our optimization space is several hundred thousands; contrast this to recent related work in computer graphics [109, 86] where the dimension is usually less than 50. In our case, a direct application of the Gauss-Newton method will result in large dense matrices that are costly to compute and store, causing the method to fail on complex examples. Below, we demonstrate how to avoid these issues, producing a robust

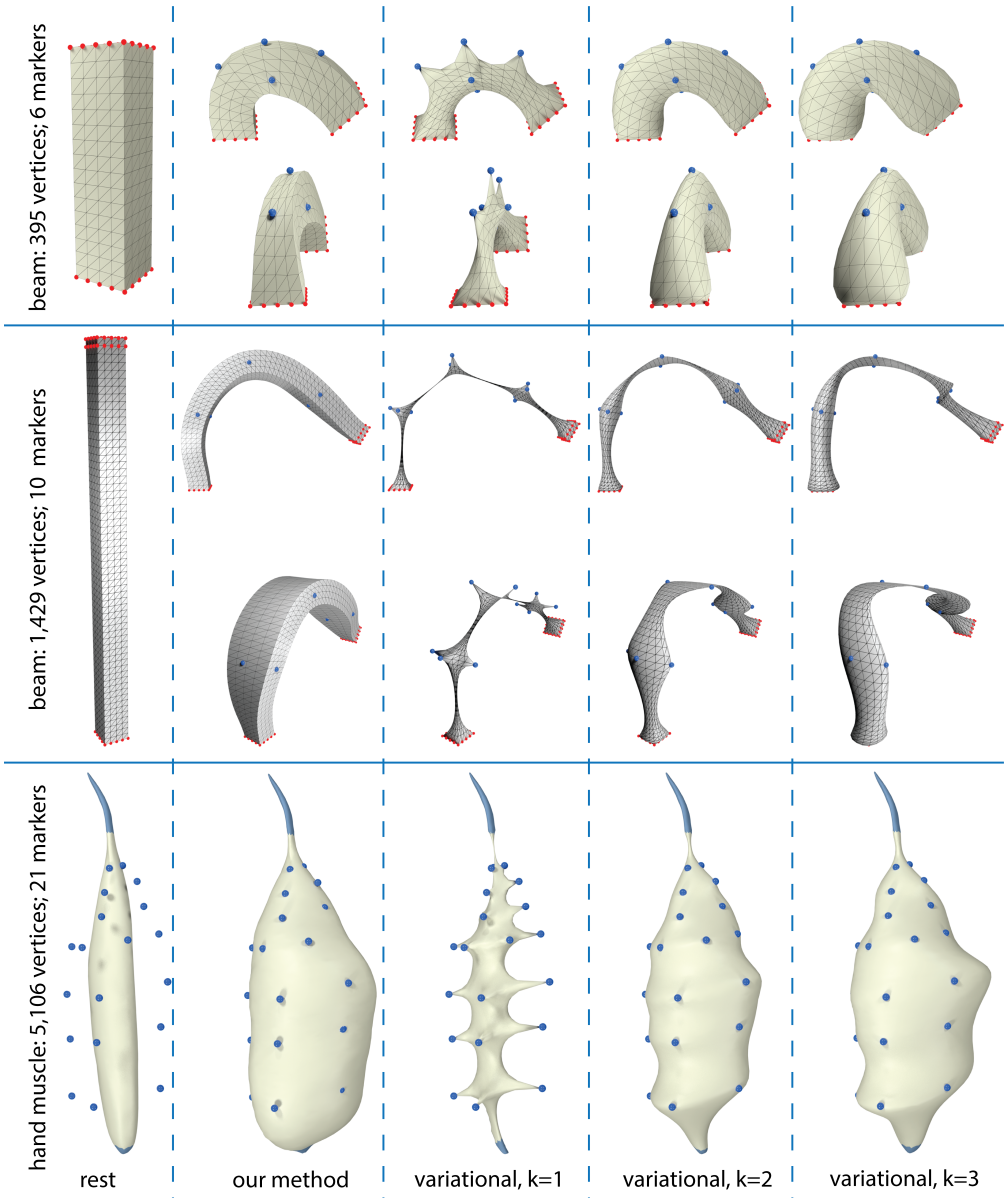


Figure 5.6: **Comparison to variational shape modeling.** In a variational method [21], the wiggles increase if one imposes a stricter constraint satisfaction. First row: under a small number of landmarks, variational methods with $k = 2, 3$ produce a smooth and reasonable result, albeit somewhat smoothening the rest shape. Middle row: under more landmarks, it becomes more difficult for variational methods to meet the landmarks while avoiding the wiggles. Bottom row: variational methods produce wavy results. Our method meets the landmarks and produces fewer wiggles. This is because the plastic deformation field can arbitrarily rotate and non-uniformly scale to adapt to the inputs; the elastic energy then finally irons out the kinks.

method capable of handling geometrically complex examples involving complex spatially varying plastic strains. Before settling on our specific Gauss-Newton approach, we attempted to use the

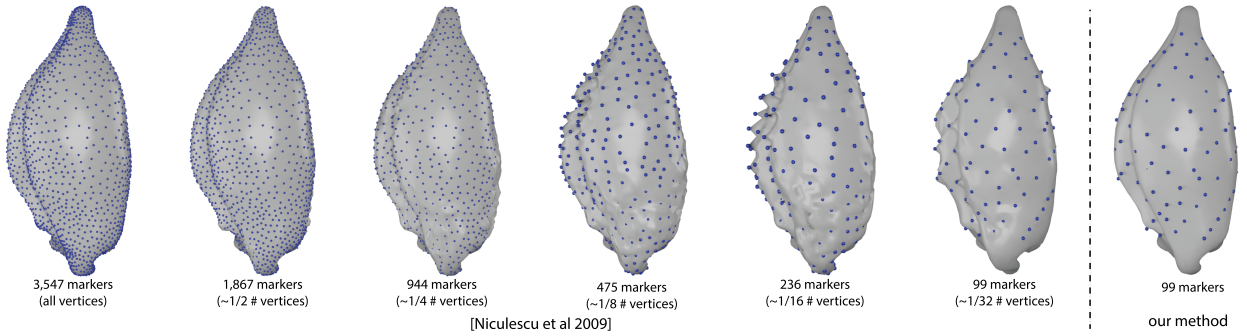


Figure 5.7: **Elastic energy methods only work with dense markers.** In this figure, we compare to a state-of-the art medical imaging technique [91], whereby the output shape is calculated by minimizing an elastic energy of a template shape, subject to dense medical image markers. With dense markers, elastic energy methods work well (left). As the constraints sparsify, elastic energy produces artefacts (middle). Our plasticity method (right) produces a good shape, even with sparse markers.

interior-point optimizer available in the state-of-the-art Knitro optimization library [9]. This did not work well because our problem is highly nonlinear. The interior-point method (IPM) worked well on simple examples, but was slow and not convergent on complex examples. IPM fails because it requires the constraint Hessian, which is not easily available (because it involves the third derivative of the elastic energy). When we approximated it, IPM generated intermediate states too far from the constraints, and failed. The strength of our Gauss-Newton approach is that we only need constraint gradients. Our method inherits the convergence properties of the Gauss-Newton method. While not guaranteed to be locally convergent, Gauss-Newton is widely used because its convergence can approach quadratic when close to the solution.

We note that our method is designed for sparse medical image landmarks and ICP markers. In Figure 5.7, we give a comparison to a related method from medical imaging which used an elastic energy, but with dense correspondences. Our method can produce a quality shape even under sparse inputs, and can consequently work even with coarser MRI scans (such as our hip bone example; Figure 5.12). The ability to work with sparse markers also translates to lower manual processing time to select the markers in the medical image.

Because the object is attached, Equation 5.4 implicitly defines $\mathbf{x}$ as a function of $\mathbf{s}$. The Gauss-Newton method uses the Jacobian $\mathbf{J} = d\mathbf{x}/d\mathbf{s}$, which models the change in the static equilibrium $\mathbf{x}$ as one changes the plastic deformation gradient. It eventually relies on the derivative of elastic forces with respect to the plastic deformation gradient, which we give in Appendix B. Although closestPoint$(\mathbf{x}, z_i)$ is a nonlinear function of $\mathbf{x}$, we can treat the barycentric coordinates of the closest point to z_i as fixed during one ICP iteration, and therefore closestPoint$(\mathbf{x}, z_i)$ becomes a linear function of $\mathbf{x}$. Therefore, $\mathscr{E}_{\text{MI}}(\mathbf{x})$ can be seen as a quadratic function of $\mathbf{x}$, and so is $\mathscr{E}_{\text{a}}(\mathbf{x})$. We can thus rewrite Equations 5.3 and 5.4 as

$$\arg\min_{\mathbf{x},\mathbf{s}} \frac{1}{2}||\mathbf{L}\mathbf{s}||^2 + \sum_{k=1}^{q+r+t} \frac{\mathbf{c_k}}{2}||\mathbf{A_k}\mathbf{x} + \mathbf{b_k}||^2, \tag{5.6}$$

$$\text{s.t.} \quad \mathbf{f}_{\text{net}}(\mathbf{s}, \mathbf{x}) = \mathbf{0}, \tag{5.7}$$

where $\mathbf{f}_{\text{net}}(\mathbf{s}, \mathbf{x}) = \mathbf{f}_{\text{e}}\big(\mathbf{F_p}(\mathbf{s}), \mathbf{x}\big) + \mathbf{f}_{\text{a}}(\mathbf{x})$ is the net force on the mesh, and constant matrices, vectors and scalars $\mathbf{A_k}, \mathbf{b_k}, \mathbf{c_k}$ are independent of $\mathbf{s}$ and $\mathbf{x}$ (we give them in Appendix F). The integer t denotes the number of attachments. We now re-write Equations 5.6 and 5.7 so that the plastic strains are expressed as $\mathbf{s} + \Delta\mathbf{s}$, and the equilibrium $\mathbf{x}$ as $\mathbf{x} + \Delta\mathbf{x}$, where $\Delta\mathbf{x} = \mathbf{J}\Delta\mathbf{s}$. At iteration i of our Gauss-Newton method, given the previous iterates $\mathbf{x}^i$ and $\mathbf{s}^i$, we minimize a nonlinearly constrained problem,

$$\arg\min_{\mathbf{x}^{i+1}, \Delta\mathbf{s}^i} \frac{1}{2}||\mathbf{L}(\mathbf{s}^i + \Delta\mathbf{s}^i)||^2 + \sum_{k=1}^{q+r+t} \frac{\mathbf{c_k}}{2}||\mathbf{A_k}\big(\mathbf{x}^i + \mathbf{J}\Delta\mathbf{s}^i\big) + \mathbf{b_k}||^2, \tag{5.8}$$

$$\text{s.t.} \quad \mathbf{f}_{\text{net}}(\mathbf{s}^i + \Delta\mathbf{s}^i, \mathbf{x}^{i+1}) = \mathbf{0}. \tag{5.9}$$

After each iteration, we update $\mathbf{s}^{i+1} = \mathbf{s}^i + \Delta\mathbf{s}^i$. Observe that Equation 5.8 does not depend on $\mathbf{x}^{i+1}$, and that the constraint of Equation 5.9 is already differentially "baked" into Equation 5.8 via $\Delta\mathbf{x} = \mathbf{J}\Delta\mathbf{s}$. We therefore first minimize Equation 5.8 for $\Delta\mathbf{s}^i$, using unconstrained minimization; call the solution $\overline{\Delta\mathbf{s}^i}$. A naive minimization requires solving a large dense linear system of equations, which we avoid using the technique presented at the end of this section. We regularize $\overline{\Delta\mathbf{s}^i}$ so

that the corresponding F_p is always positive-definite for each tet; we do this by performing eigen-decomposition of the symmetric matrix F_p at each tet, and clamping any negative eigenvalues to a small positive value (we use 0.01). Our method typically did not need to perform clamping in practice, and in fact such clamping is usually a sign that the method is numerically diverging, and should be restarted with better parameter values.

We then minimize the optimization problem of Equations 5.8 and 5.9 using a 1D line search, using the search direction $\overline{\Delta \mathbf{s}^i}$. Specifically, for $\eta \geq 0$, we first solve Equation 5.7 with $\mathbf{s}(\eta) := \mathbf{s}^i + \eta \overline{\Delta \mathbf{s}^i}$ for $\mathbf{x} = \mathbf{x}(\eta)$ using the Knitro library [9]. Direct solutions using a Newton-Raphson solver also worked, but we found Knitro to be faster. We then evaluate the objective of Equation 5.6 at $\mathbf{x} = \mathbf{x}(\eta)$ and $\mathbf{s} = \mathbf{s}(\eta)$. We perform the 1D line search for the optimal η using the gradient-free 1D Brent's method [94].

Initial guess: We solve our optimization problem by first assuming a constant s at each tet, starting from the template mesh as the initial guess. This roughly positions, rotates and globally scales the template mesh to match the medical image. We use the output as the initial guess for our full optimization as described above.

Optimization stages and stopping criteria: We first do the optimization with attachments only. Upon convergence, we add the landmarks, ignoring any ICP markers. This is because initially, the mesh is far away from the target and the ICP closest locations are unreliable. After convergence, we disable the landmarks and enable the ICP markers and continue optimizing. After this optimization meets a stopping criterium, we are done. Our output is therefore computed with ICP markers only; landmarks only serve to guide the optimizer. This is because landmarks require a correct correspondence, and it is harder to mark this correspondence reliably in the scan than to simply select an ICP marker on the boundary of an organ. We recompute the closest locations to ICP markers after each Gauss-Newton iteration. We stop the optimization if either of the following three criteria is satisfied: (i) reached the user-specified maximal number of iterations (typically 20; but was as high as 80 in the liver example), (ii) maximum error at ICP markers is less than

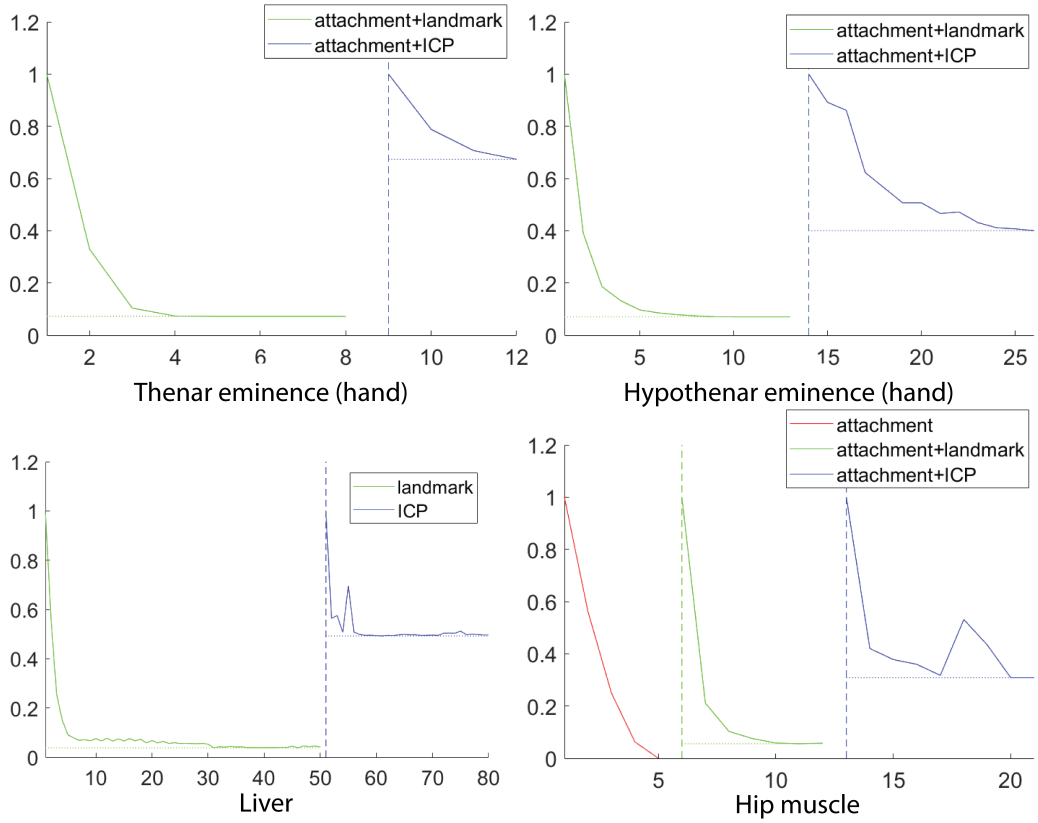


Figure 5.8: **Convergence plots.** The iterations are presented on the X-axis, and the optimization energy is marked on the Y-axis. The initial optimization energies are normalized to 1.0.

a user-specified value (1mm for hand muscles), (iii) the progress in each iteration is too small, determined by checking if η is under a user-defined threshold (we use 0.01). Figure 5.8 shows the convergence of our optimization.

Avoiding dense large linear systems Because the object is attached, $\frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{x}}$ is square and invertible. Therefore, one can obtain a formula for $\mathbf{J}$ by differentiating Equation 5.7 with respect to $\mathbf{s}$,

$$\mathbf{J} = -\left(\frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{x}}\right)^{-1} \frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{s}}. \tag{5.10}$$

The matrix $\mathbf{J}$ is dense (dimensions $3n \times 6m$). Observe that because Equation 5.8 is quadratic in $\Delta\mathbf{s}^i$, minimizing it as done above to determine the search direction $\overline{\Delta\mathbf{s}^i}$ is equivalent to solving a linear

system with the system matrix $\mathbf{H}$, where $\mathbf{H}$ is the second derivative (Hessian matrix; dimension $6m \times 6m$) of Equation 5.8 with respect to $\Delta\mathbf{s}^i$. Because $\mathbf{J}$ is dense, $\mathbf{H}$ is likewise a dense matrix,

$$\mathbf{H} = \mathbf{L}^2 + \mathbf{J}^T\left(\sum_k \mathbf{c_k}\mathbf{A_k}^T\mathbf{A_k}\right)\mathbf{J} = \mathbf{L}^2 + \mathbf{Z}^T\mathbf{Z}, \quad \text{where} \tag{5.11}$$

$$\mathbf{Z} = \begin{bmatrix} \sqrt{\mathbf{c_1}}\mathbf{A_1} \\ \sqrt{\mathbf{c_2}}\mathbf{A_2} \\ \vdots \\ \sqrt{\mathbf{c_{q+r+t}}}\mathbf{A_{q+r+t}} \end{bmatrix} \mathbf{J} = -\begin{bmatrix} \sqrt{\mathbf{c_1}}\mathbf{A_1} \\ \sqrt{\mathbf{c_2}}\mathbf{A_2} \\ \vdots \\ \sqrt{\mathbf{c_{q+r+t}}}\mathbf{A_{q+r+t}} \end{bmatrix} \left(\frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{x}}\right)^{-1} \frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{s}}. \tag{5.12}$$

Therefore, when the number of tet mesh elements m is large, it is not practically possible to compute $\mathbf{H}$, store it explicitly in memory or solve linear systems with it. To avoid this problem, we first tried solving the system of equations using the Conjugate Gradient (CG) method. This worked, but was very slow (Table 5.1). The matrix $\mathbf{Z} \in \mathbb{R}^{3(q+r+t)\times 6m}$ is dense. In our complex examples, the number of medical image constraints $q+r+t$ is small (typically 10 - 800) compared to the dimension of $\mathbf{s}$ ($6m$; typically ˜200,000). Our idea is to efficiently compute the solution to a system $\mathbf{H}y = h$ for any right-hand side h using the Woodbury matrix identity [141], where we view $\mathbf{L}^2$ as a "base" matrix and $\mathbf{Z}^T\mathbf{Z}$ a low-rank perturbation. Before we can apply Woodbury's identity, we need to ensure that the base matrix is invertible. As we prove in Appendix A, the plastic strain Laplacian $\mathbf{L}$ is singular with six orthonormal vectors $\boldsymbol{\psi_i}$ in its nullspace (assuming that $\mathscr{M}$ is connected). Each $\boldsymbol{\psi_i}$ is a vector of all ones in component i of s, $i = 1,\ldots,6$ and all zeros elsewhere, divided by $\sqrt{m}$ for normalization. It follows from the Singular Lemma (i) (Section 5.5) that $\mathbf{L}^2$ is also singular with the same nullspace vectors. Therefore, we decompose

$$\mathbf{H} = \left(\mathbf{L}^2 - \sum_{i=1}^{6} \boldsymbol{\psi_i}\boldsymbol{\psi_i}^T\right) + \left(\mathbf{Z}^T\mathbf{Z} + \sum_{i=1}^{6} \boldsymbol{\psi_i}\boldsymbol{\psi_i}^T\right) = \mathbf{B} + \hat{\mathbf{Z}}^T\hat{\mathbf{Z}}, \tag{5.13}$$

Table 5.1: **Solving a single linear system of equations with H,** using the conjugate gradients (CG) and our method. The naive direct solver failed in all cases. Note that $\mathbf{H}$ is a dense $6m \times 6m$ matrix. The column t_{prep} gives a common pre-processing time for both CG and our method.

Example	$6m$	$3(q+r+t)$	t_{prep}	CG	Ours
Hand muscle	237,954	1,143	17.5s	897.5s	9.5s
Hip bone	172,440	1,497	11.4s	408.9s	7.2s
Liver	259,326	1,272	24.4s	1486.7s	10.0s

where $\mathbf{B} = \mathbf{L}^2 - \sum_{i=1}^{6} \psi_\mathbf{i} \psi_\mathbf{i}^T$ and $\hat{\mathbf{Z}}$ is matrix $\mathbf{Z}$ with an additional added 6 added rows $\psi_\mathbf{i}^T$. By the Singular Lemma (iii) (Section 5.5), $\mathbf{B}$ is now invertible, and we can use Woodbury's identity to solve

$$y = \mathbf{H}^{-1} h = \left(\mathbf{B}^{-1} - \mathbf{B}^{-1} \hat{\mathbf{Z}}^T \left(I + \hat{\mathbf{Z}} \mathbf{B}^{-1} \hat{\mathbf{Z}}^T \right)^{-1} \hat{\mathbf{Z}} \mathbf{B}^{-1} \right) h. \tag{5.14}$$

We rapidly compute $\mathbf{Z}$, without ever computing or forming $\mathbf{J}$, by solving sparse systems $\frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{x}} \tilde{z}_k = -\sqrt{\mathbf{c_k}} \mathbf{A_k}^T$, and $z_k = \tilde{z}_k^T \frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{s}}$, for $k = 1, \ldots, (q+r+t)$, where $z_k \in \mathbb{R}^{3 \times 6m}$ is the k-th 3-row block of $\mathbf{Z}$. Observe that this sparse system matrix is symmetric and the same for all k. We factor it once using the Pardiso solver and then solve the multiple right-hand sides in parallel. Once $\tilde{z}_k^T$ has been computed, $z_k = \tilde{z}_k^T \frac{\partial \mathbf{f}_{\text{net}}}{\partial \mathbf{s}}$ is easy to compute because it is a multiplication of a thin dense matrix and a sparse matrix. The matrix $\mathbf{B}$ is constant, and we only need to factor it once for the entire optimization. Finally, the matrix $I + \hat{\mathbf{Z}} \mathbf{B}^{-1} \hat{\mathbf{Z}}^T \in \mathbb{R}^{3(q+r+t) \times 3(q+r+t)}$ is small, and so inverting it is fast. We analyze the performance of our algorithm in Table 5.1.

5.5 Singular lemma

In this thesis, there are two occasions where we have to solve a singular sparse linear system with known nullspace vectors. Such systems occur often in modeling of unattached objects, e.g., finding static equilibria, solving Laplace equations on the object's mesh, animating with rotation-strain coordinates [53], or computing modal derivatives [12]. Previous work solved such systems ad-hoc, and the underlying theory has not been stated or developed in any great detail; or it only

addressed non-singular systems. For example, in order to constrain vertices in cloth simulation, Boxerman [25] modified the linear system $Ax = b$ to $SAx = Sb$, for a properly chosen filtering matrix S. However, A is non-singular to begin with; there is no discussion of singularity in Boxerman's work.

We hereby state and prove a lemma that comprehensively surveys the common situations arising with singular systems in computer animation and simulation, and back the lemma with a mathematical proof (Appendix C). Recall that the *nullspace* of a matrix $A \in \mathbb{R}^{p\times p}$ is $\mathcal{N}(A) = \{x \in \mathbb{R}^p\,;\, Ax = 0\}$, and the *range* of A is $\mathcal{R}(A) = \{Ax; x \in \mathbb{R}^p\}$. Both are linear vector subspaces of $\mathbb{R}^p$.

Singular Lemma: Let the square symmetric matrix $A \in \mathbb{R}^{p\times p}$ be singular with a known nullspace spanned by k linearly independent vectors $\psi_1, \ldots, \psi_k$. Then the following statements hold:

(i) $\mathcal{N}(A)$ and $\mathcal{R}(A)$ are orthogonal. Every vector $x \in \mathbb{R}^p$ can be uniquely expressed as $x = n + r$, where $n \in \mathcal{N}(A)$ and $r \in \mathcal{R}(A)$. Vector r is orthogonal to n and to ψ_i for all $i = 1, \ldots, k$ (Figure 5.9).

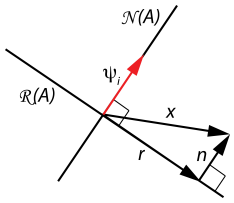


Figure 5.9: **Illustration of the Singular Lemma.**

(ii) Let $b \in \mathcal{R}(A)$. Then, the singular system $Ax = b$ has a unique solution x with the property that x is orthogonal to ψ_i for all $i = 1, \ldots, k$. This solution can be found by solving the *non-singular* linear system

$$\begin{bmatrix} A & \psi_1 & \ldots & \psi_k \\ \psi_1^T & 0 & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ \psi_k^T & 0 & \ldots & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda_1 \\ \vdots \\ \lambda_k \end{bmatrix} = \begin{bmatrix} b \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{5.15}$$

All other solutions equal $x + \sum_{i=1}^{k} \mu_i \psi_i$ for some scalars $\mu_i \in \mathbb{R}$.

(iii) For any scalars $\alpha_i \neq 0$, the matrix $B = A + \sum_{i=1}^{k} \alpha_i \psi_i \psi_i^T$ is invertible. If ψ_i are orthonormal vectors, then the solution to $By = h$ equals $y = x + \sum_{i=1}^{k} \frac{\lambda_i}{\alpha_i} \psi_i$, where x and λ_i are solutions to

Equation 5.15 with $b = \text{proj}_{\mathscr{R}(A)} h = h - \sum_{i=1}^{k} (\psi_i^T h)\psi_i$. We give the proof of the singular lemma in Appendix C.

5.6 Unattached objects

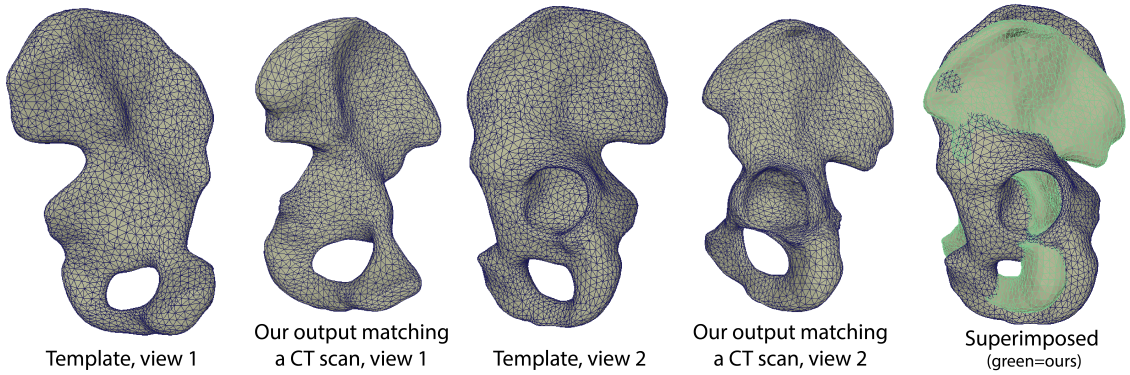


Figure 5.10: **Unattached optimization of a hip bone shape to a CT scan.** The scanned bone is smaller and has a substantially different shape from the template.

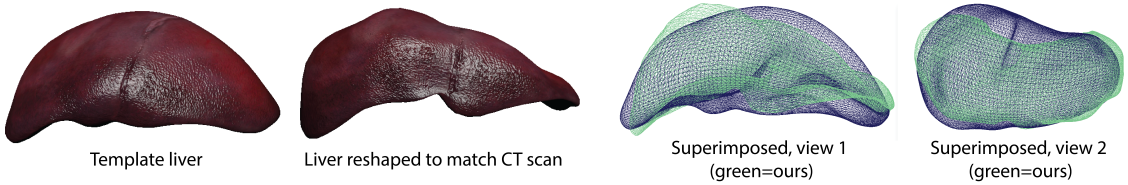


Figure 5.11: **Unattached optimization of a liver shape to a CT scan.** Our method successfully captures the large shape variation between the template and the scan. This figure also demonstrates that our method makes it possible to transfer the rendering textures and UV coordinates from the template onto the output.

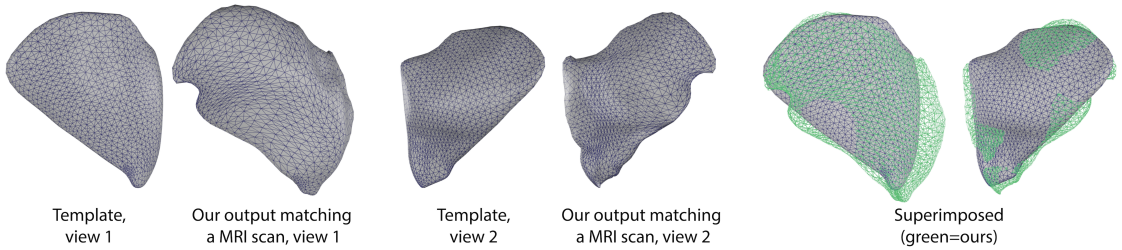


Figure 5.12: **Attached optimization of a hip muscle (gluteus medius) to an MRI scan.** Our method successfully captures the large shape variation between the template and the scan.

The difficulty with unattached objects is that we now have $\mathbf{f}_{\text{net}} = \mathbf{f}_{\text{e}}$, and the equation $\mathbf{f}_{\text{e}}\big(\mathbf{F}_{\mathbf{p}}(\mathbf{s}), \mathbf{x}\big) = 0$ no longer has a unique solution $\mathbf{x}$ for a fixed plastic state $\mathbf{s}$. This can be intuitively easily understood: one can arbitrarily translate and rotate any elastic equilibrium shape $\mathbf{x}$ under the given plastic state $\mathbf{s}$; doing so produces another elastic equilibrium shape. The space of solutions $\mathbf{x}$ is 6-dimensional. This means that we can no longer uniquely solve Equation 5.7 for $\mathbf{x}$ during our line search of Section 5.4. Furthermore, the square tangent stiffness matrix

$$\mathbf{K}(\mathbf{F}_{\mathbf{p}}(\mathbf{s}), \mathbf{x}) = \frac{\partial \mathbf{f}_{\text{e}}\big(\mathbf{F}_{\mathbf{p}}(\mathbf{s}), \mathbf{x}\big)}{\partial \mathbf{x}} \tag{5.16}$$

is no longer full rank. In order to address this, we now state and prove the following Nullspace Lemma.

Nullspace Lemma: The nullspace of the tangent stiffness matrix of an elastoplastic deformable object in *static equilibrium* $\mathbf{x}$ under plasticity, is 6-dimensional. The six nullspace vectors are $\psi_i := [e_i, e_i, \ldots, e_i]$, where $e_i \in \mathbb{R}^3$ is the i-th standard basis vector, and $\psi_{3+i} := [e_i \times x_1, e_i \times x_2, \ldots, e_i \times x_n]$, for $i = 1, 2, 3$.

To the best of our knowledge, this fact of elasto-plasto-statics has not been stated or proven in prior work. It is very useful when modeling large-deformation elastoplasticity, as real objects are often unattached, or attachments cannot be easily modeled. We give a proof in Appendix D. To accommodate unattached objects, it is therefore necessary to stabilize the translation and rotation. For translations, this could be achieved easily by fixing the position of any chosen vertex. Matters are not so easy for rotations, however. Our idea is to constrain the centroid of all tet mesh vertices to a specific given position t, and to constrain the "average rotation" of the model to a specific given rotation R. We achieve this using the familiar "shape-matching" [90], by imposing that the

rotation in the polar decomposition of the global covariance matrix must be R. We therefore solve the following optimization problem,

$$\underset{\mathbf{x},\mathbf{s},R,t}{\arg\min}\ ||\mathbf{L}\mathbf{s}||^2 + \alpha\mathscr{E}_{\mathrm{MI}}(\mathbf{x}) \tag{5.17}$$

$$\text{s.t.}\quad \mathbf{f}_{\mathrm{e}}\big(\mathbf{F}_{\mathbf{p}}(\mathbf{s}),\mathbf{x}\big) = \mathbf{0}, \tag{5.18}$$

$$\left(\sum_{j\in D} w_j x_j\right) - \sum_{j\in D} w_j X_j = t, \tag{5.19}$$

$$\mathrm{Polar}\left(\sum_{j\in D} w_j\Big(x_j - t\Big)\Big(X_j - \big(\sum_{k\in D} w_k X_k\big)\Big)^T\right) = R, \tag{5.20}$$

where D is the set of points on the mesh surface where we have either a landmark or an ICP constraint, w_j is the weight of a point, X_j is the position of vertex j in $\mathscr{M}$ and $\mathrm{Polar}(F)$ is the polar decomposition function that extracts the rotational part of a matrix F. We set all weights equal, i.e., $w_j = 1/|D|$. We choose the set D as opposed to all mesh vertices so that we can easily perform optimization with respect to R and t (next paragraph). We assume that our argument matrices F to Polar are not inversions, i.e., $\det(F) > 0$, which establishes that $\mathrm{Polar}(F)$ is always a rotation and not a mirror. This requirement was easily satisfied in our examples, and is essentially determined by the medical imaging constraints; the case $\det(F) < 0$ would correspond to an inverted (or mirror) medical image, which we exclude.

We solve the optimization problem of Equations 5.17, 5.18, 5.19 and 5.20 using a block-coordinate descent, by iteratively optimizing $\mathbf{x},\mathbf{s}$ while keeping R,t fixed and vice-versa (Figure 5.10, 5.11). Rigid transformations do not affect smoothness of $\mathbf{s}$ so we do not need to consider it when optimizing R,t. We need to perform two modifications to our Gauss-Newton iteration of Section 5.4. The first modification is that we need to simultaneously solve Equations 5.18, 5.19 and 5.20 when determining the static equilibrium in the current plastic state $\mathbf{s}$. As with attached objects, we do this using the Knitro optimizer. In order to do this, we need to compute the first and second derivatives of the stabilization constraints in Equations 5.19 and 5.20 (Section 5.7). The second modification is needed because the tangent stiffness matrix $\mathbf{K}(\mathbf{F}_{\mathbf{p}}(\mathbf{s}),\mathbf{x})$, as explained

above, is now singular with a known 6-dimensional nullspace. In order to compute the Jacobian matrix **J** using Equation 5.10, we use our Singular Lemma (ii) (Section 5.5). Note that the right-hand side is automatically in the range of **K** because Equation 5.10 was obtained by differentiating a valid equation, hence Equation 5.10 must also be consistent.

5.7 Gradient and Hessian of $\mathrm{Polar}(F)$

Previous work computed first and second-order *time* derivatives of the rotation matrix in polar decomposition [14], or first derivative with respect to each individual entry of F [124, 28]. In our work, we need the first *and* second derivatives of R with respect to each individual entry of F. We found an elegant approach to compute them using Sylvester's equation, as follows. Observe that $\mathrm{Polar}(F) = FS^{-1}$, where S is the symmetric matrix in the polar decomposition. Because $\det(F) > 0$, S is positive-definite and uniquely defined as $S = \sqrt{F^T F}$. To compute the first-order derivatives, we start from $F = RS$, and differentiate,

$$\frac{\partial F}{\partial F_i} = \frac{\partial R}{\partial F_i} S + R \frac{\partial S}{\partial F_i}, \quad \text{hence} \quad \frac{\partial R}{\partial F_i} = \left(\frac{\partial F}{\partial F_i} - R \frac{\partial S}{\partial F_i} \right) S^{-1}, \tag{5.21}$$

Therefore, we need to compute $\partial S / \partial F_i$. We have

$$F^T F = S^2, \quad \text{and thus} \quad \frac{\partial F^T F}{\partial F_i} = \frac{\partial S}{\partial F_i} S + S \frac{\partial S}{\partial F_i}, \tag{5.22}$$

i.e., this is the classic Sylvester equation for the unknown matrix $\frac{\partial S}{\partial F_i}$ [119]. The Sylvester equation $AX + XB = C$ can be solved as

$$\left(B^T \oplus A \right)^{-1} \mathrm{vec}(X) = \mathrm{vec}(C), \tag{5.23}$$

where $\oplus$ is the Kronecker sum of two matrices. In our case,

$$\mathrm{vec}\left(\frac{\partial S}{\partial F_i}\right) = (S \oplus S)^{-1} \mathrm{vec}\left(\frac{\partial F^T F}{\partial F_i}\right). \tag{5.24}$$

The computation of second-order derivatives follows the same recipe: differentiate the polar decomposition and solve a Sylvester equation. We give it in Appendix E.

We can now compute the gradient and Hessian of our stabilization constraints. The translational constraint is linear in $\mathbf{x}$ and can be expressed as $W_1\mathbf{x} - d_1 = 0$, where W_1 is a $3 \times 3n$ sparse matrix. Although Polar is not linear, the argument of Polar is linear in $\mathbf{x}$. The rotational constraint can be expressed as $\text{Polar}(W_2\mathbf{x} - d_2) - \bar{R} = 0$, where W_2 is a $9 \times 3n$ sparse matrix. The Jacobian of the translational constraint is W_1, and the Hessian is zero. For the rotational constraint, the Jacobian is $\frac{\partial R}{\partial F} : W_2 \in \mathbb{R}^{9\times 3n}$ and the Hessian is $(W_2^T : \frac{\partial^2 R}{\partial F^2} : W_2) \in \mathbb{R}^{9\times 3n\times 3n}$, where : denotes tensor contraction.

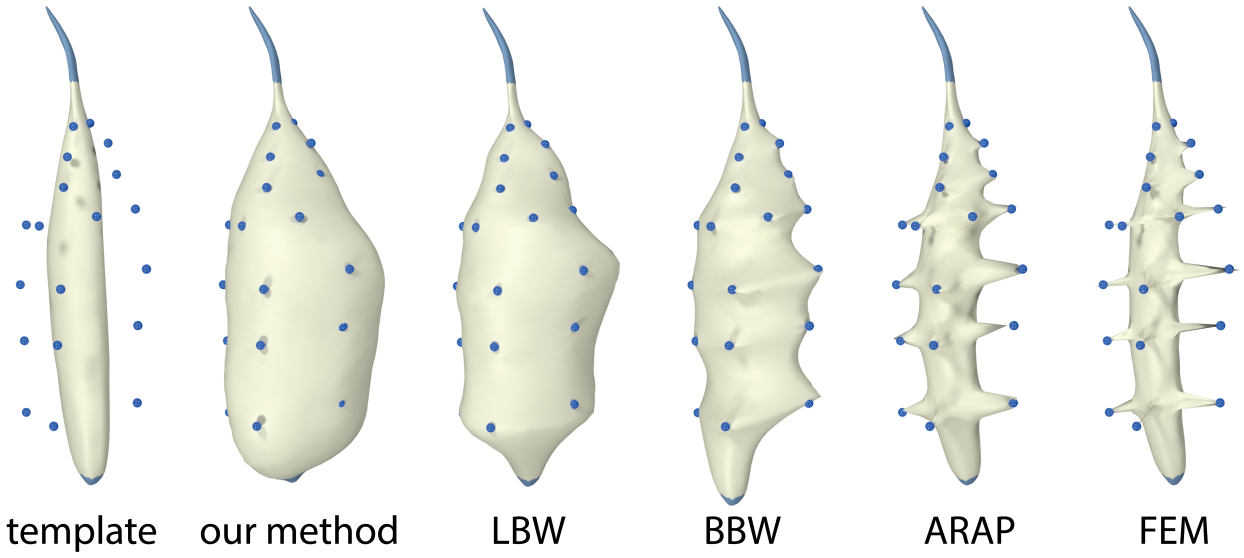


Figure 5.13: **Standard volume-based shape deformation methods result in wiggly and spiky artefacts.** The displayed hand Palmar interossei muscle has a tendon on one side and no tendon on the other; both ends are attached to a bone (light blue). The acronyms are defined in Section 5.8. The MRI landmark constraints are shown in dark blue. The shape deformation between the template muscle and the shape in the MRI consists of large and spatially varying stretches. Our method successfully models this deformation. We note that spikes cannot be avoided by using, say, a spherical region for the constraints as opposed to a point; the non-smoothness merely moves to the spherical region boundary.

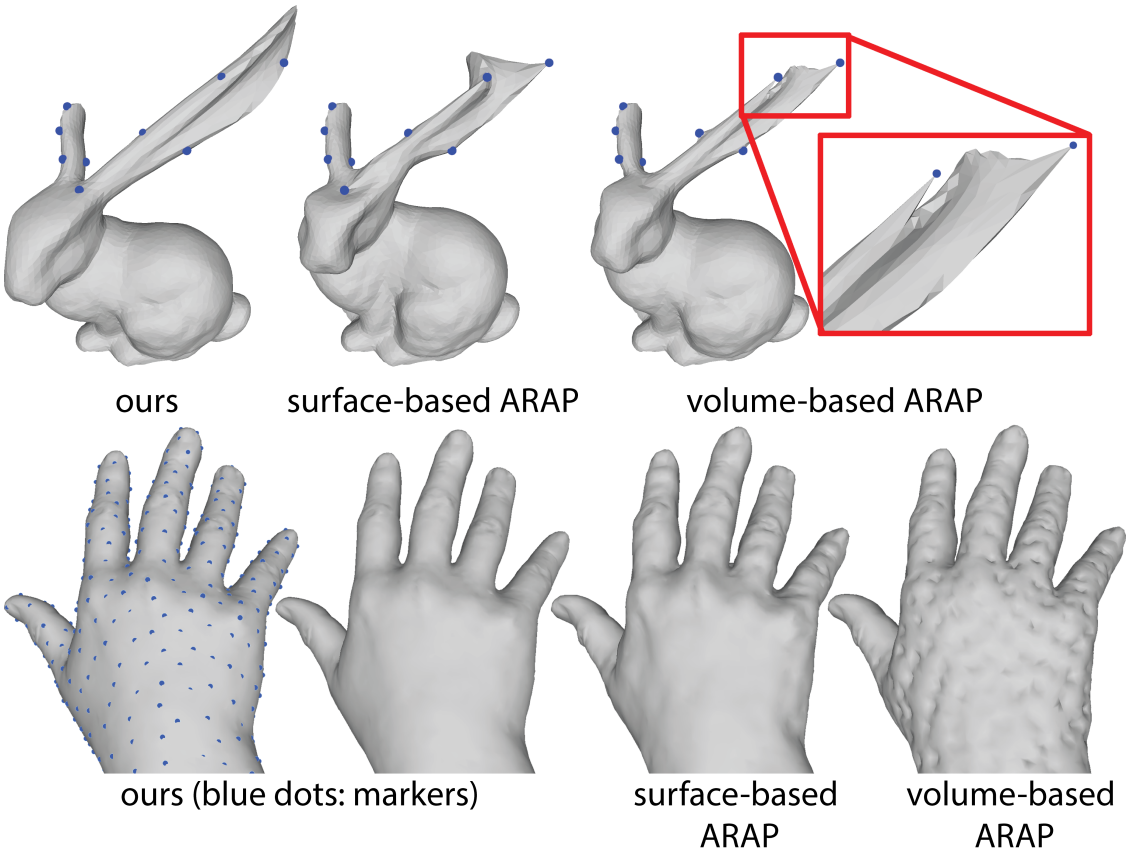


Figure 5.14: **Comparison between surface energy and volumetric energy.** In both examples (bunny and hand), we performed non-rigid iterative closest point alignment between a template triangle mesh and a collection of target ICP markers (13 for bunny and 456 for the hand). For the bunny, we manually placed the markers to greatly enlarge the bunny's left ear. For the hand, we placed the markers on a pre-existing target hand surface mesh that has different geometric proportions and mesh topology as the template. The template is a man and target is a woman. We then solved the ICP problem using the surface-based ARAP energy, volume-based ARAP energy, and our volumetric plastic strains. Our method produces smooth artefact-free outputs.

5.8 Comparison to standard shape deformation methods

Our shape deformation setup is similar to standard shape deformation problems in computer graphics. In fact, we first attempted to solve the shape deformation problem with as-rigid-as-possible energy (ARAP) [113], bounded biharmonic weights (BBW) [59], biharmonic weights with linear precision (LBW) [137], and a Finite Element Method static solver (FEM) [11]. Unfortunately, none of the methods worked well. Figures 5.13 and 5.14 demonstrate that these methods produce non-smooth shapes with spikes (ARAP, BBW, FEM), or wiggles (LBW).

Mathematically, the reason for the spikes in ARAP, BBW and FEM is that point constraints for second-order methods are inherently flawed. As one refines the solution by adding more tetrahedra, the solution approaches a spiky point function at each point constraint, which is obviously not desirable. This mathematical issue is exposed in our work because our shape deformation consists of large spatially varying stretches. Often, the template mesh needs to be stretched $\sim$2x or more along some (or several) coordinate axes. The medical image constraints are distributed all around the muscle, pulling in different directions and essentially requesting the object to undergo a spatially non-uniform and anisotropic scale. This exacerbates the spikiness for second-order methods. We note that these problems cannot be avoided simply by using an elastic energy that permits volume growth. Namely, Drucker's stability condition [37] requires a monotonic elastic energy increase with increase in strain. An elastic energy therefore must penalize strain increases if it is to be stable; and this impedes large-strain modeling in methods that rely purely on an elastic energy. Our plasticity method does not penalize large strains and thus avoids this problem. Spikes can be avoided by using a higher-order variational method such as LBW. However, our experiments indicate that such methods suffer from wiggles when applied to medical imaging problems (see also Figures 5.6 and 5.22).

5.9 Results

We extracted muscles of the human hand and the hip muscle from an MRI scan, and a hip bone and a liver from a CT scan. We analyze the performance of our method in Table 5.2. In Figure 5.15, we give histograms of the magnitude of the difference between the positions of the medical image markers and their output positions. It can be seen that our method produces shapes that generally match the medical image constraints to 0.5mm or better. In Figure 5.16, we demonstrate that the output quality of our tetrahedra is still good; if needed, this could be further improved by remeshing [15]. Figures 5.17 and 5.18 superimpose our output meshes on the CT and MRI scans, respectively. In Figure 5.20, we compare to a recent implicit point set surface reconstruction

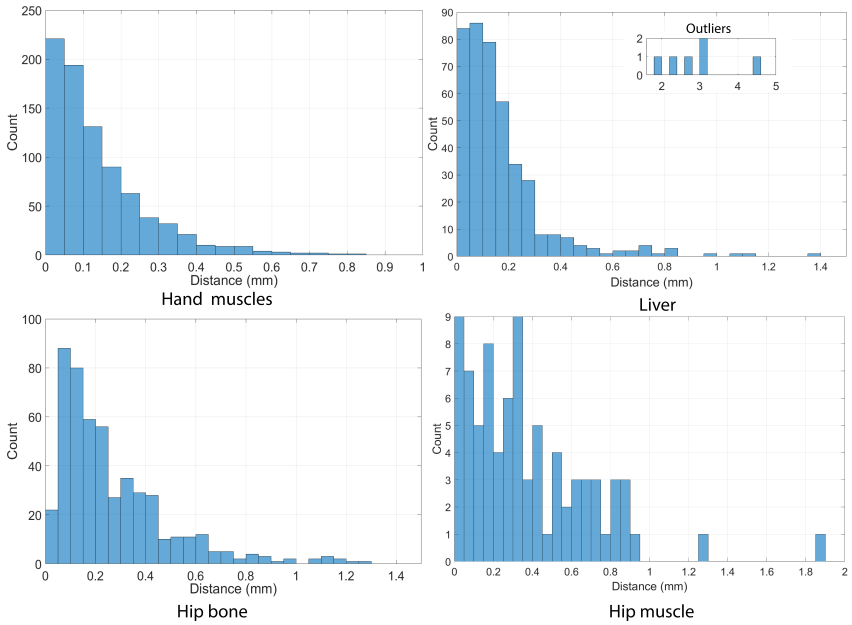


Figure 5.15: **Output ICP error histograms.** Each medical image marker contributes one entry to the histogram. The hand muscles histogram is a combined histogram for all the 17 hand muscles.

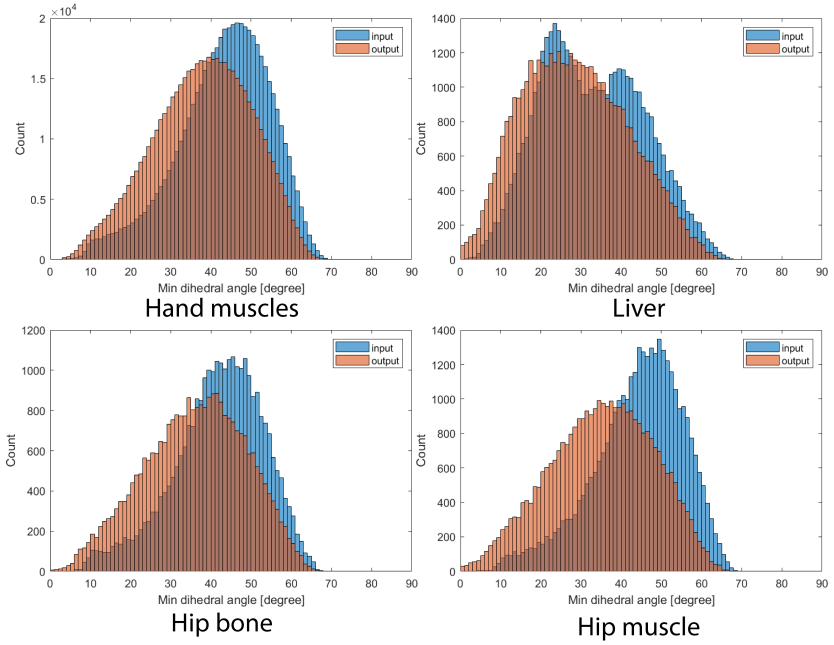


Figure 5.16: **Minimum tetrahedral dihedral angles before and after our optimization.** It can be seen that the output angles are still relatively large. As expected, the output angles are somewhat worse than the initial ones, as the object has undergone a plastic deformation.

Table 5.2: **The statistics for our examples:** #vtx = number of vertices; #ele = number of tetrahedra in $\mathscr{M}$; #iter = number of ICP iterations; "time" = total computation time to compute the output shape; "attached" indicates whether the object is attached; e_{init} = error between our template mesh and the ICP constraints; e_{final} = error between our result and the ICP markers. The first and second reported error numbers are the average and maximum errors, respectively. In the hand example, there are 17 groups of muscles; "min", "med" and "max" refer to the smallest, representative median, and largest muscle groups; "max-m" is the example with the largest number of ICP constraints. Observe that our markers are sparse; the ratio between the number of markers and the number of vertices is 0.3%–7.3% in our examples.

Example	# vtx	# ele	# markers	# iter	time [min]	attached	$e_{\text{init}}[mm]$	$e_{\text{final}}[mm]$
Hand muscle (min)	4,912	20,630	15	8	14.2	yes	0.53 / 2.22	0.06 / 0.14
Hand muscle (med)	6,260	31,737	32	12	12.8	yes	0.62 / 1.56	0.11 / 0.55
Hand muscle (max)	8,951	42,969	96	11	14.2	yes	3.35 / 12.82	0.11 / 0.34
Hand muscle (max-m)	7,552	34,966	151	18	20.3	yes	3.28 / 9.11	0.16 / 0.47
Hip muscle (Fig 5.12)	6,793	34,118	82	21	28.3	yes	7.41 / 21.27	0.39 / 1.85
Hip bone	6,796	28,740	499	34	49.2	no	4.12 / 14.27	0.25 / 1.30
Liver	11,392	43,221	424	80	128.3	no	9.00 / 33.68	0.21 / 4.81
Hand surface (Fig 5.14)	11,829	49,751	456	31	43.8	no	4.87 / 16.78	0.07 / 0.86

method [54]. In Figure 5.19, we evaluate our method in the presence of known ground truth plastic deformation gradients. Figures 5.21, 5.23, 5.24, 5.25 provide comparisons to surface-based methods, in addition to Figures 5.6,5.14. In all comparisons, our method outperformed surface-based methods.

5.9.1 Hand muscles

In our muscle hand example, we extracted 17 hand muscles from an MRI scan (Figure 5.5). We obtained the scan and the already extracted bone meshes from [134]; scan resolution is 0.5mm x 0.5mm x 0.5mm . We considered two "templates", the first one from the Centre for Anatomy and Human Identification at the University of Dundee, Scotland [26], and the second one from Zygote [147]. We used the first one (Figure 5.5, left) because we found it to be more medically accurate (muscles insert to correct bones). Muscle anatomy of a human hand is challenging (Figure 5.5). We model all muscle groups of the hand, namely the thenar eminence (thumb), hypothenar eminence (below little finger), interossei muscles (palmar and dorsal) (between metacarpal bones),

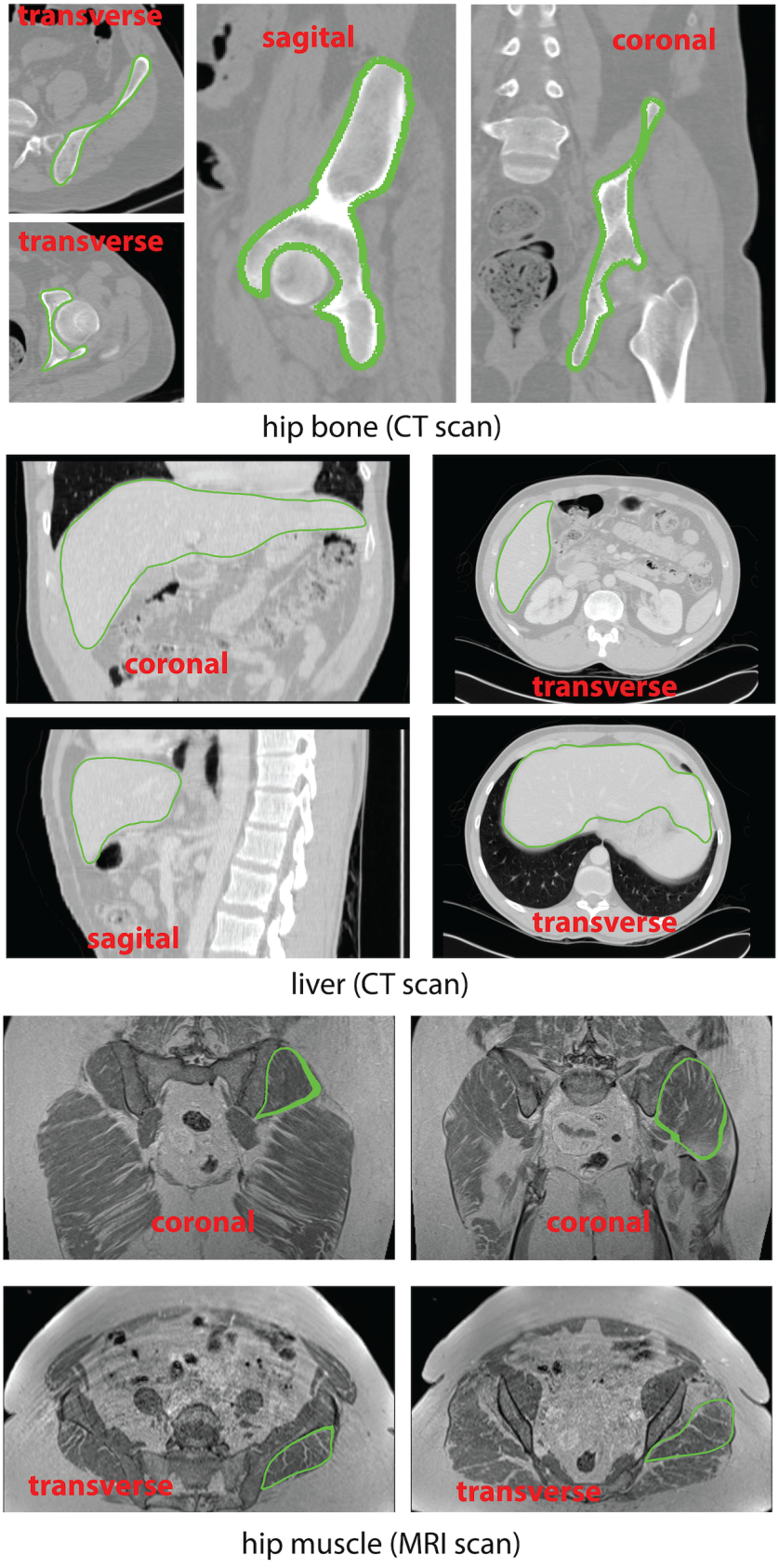


Figure 5.17: **Our extracted organs match the medical image.** The intersection of the output mesh with the medical image slices is shown in green.

adductor pollicis (soft tissue next to the thumb, actuating thumb motion), and lumbricals (on the side of the fingers at the fingers base). Our template models the correct number and general location of the muscles, but there are large muscle shape differences between the template subject and the scanned subject (Figure 5.1). We solve the optimization problem of Equations 5.3 and 5.4 separately for each muscle, starting from the template mesh as the initial guess. In our results, this produces muscles that match the attachments and medical image constraints markers at 0.5 mm or better, which is at, or better than, the accuracy of the MRI scanner.

5.9.1.1 Marking the muscles in MRI scans

During pre-processing, we manually mark as many reliable points as possible on the boundary of each muscle ($\sim 10-20$ landmarks and $\sim 50-100$ ICP markers per muscle) in the MRI scans. This process took approximately 5 minutes per muscle.

5.9.1.2 Attachments to bones

The template muscles are modeled as triangle meshes. We build a tetrahedral mesh for each muscle. Our tet meshes conform to the muscle's surface triangle mesh; this requirement could be relaxed. For each muscle in the template, we attach its tet mesh to the bones using soft constraints. We do this by marking where on one or multiple bones this muscle inserts; to do so, we consulted a medical doctor with specific expertise in anatomy. For each bone triangle mesh vertex that participates in the insertion, we determine the material coordinates (i.e., tet barycentric coordinates) in the closest muscle tet. We then form a soft constraint whereby this muscle material point is linked to the bone vertex position using a spring.

5.9.1.3 Direct attempt using segmentation:

We note that we have also attempted to model the muscle shapes directly using segmentation, simply from an MRI scan. Recent work has demonstrated that this can be done for hand bones [134], and we attempted a similar segmentation approach for muscles. However, given that the muscles

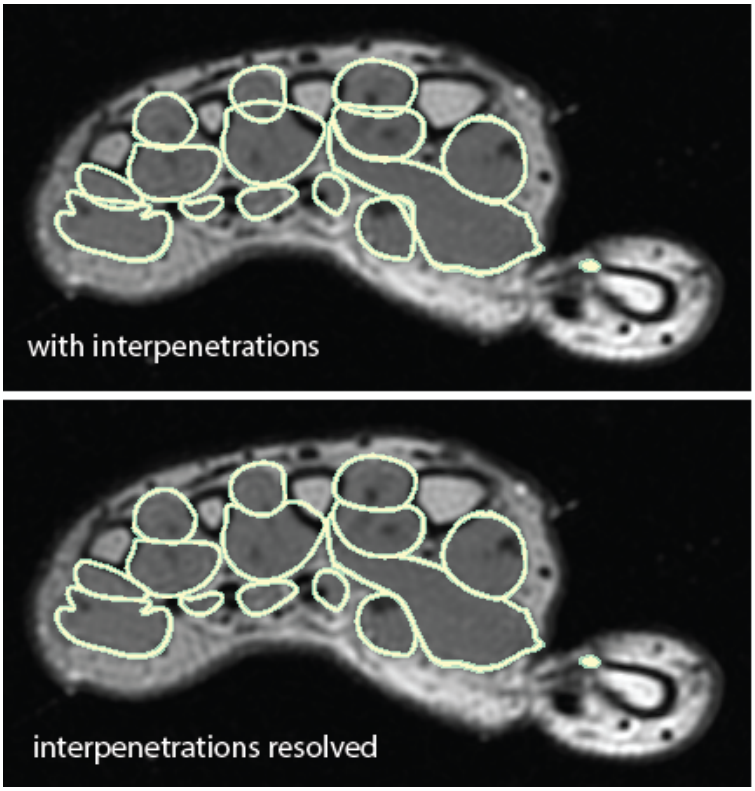


Figure 5.18: **Interpenetration removal.** The yellow lines are muscle cross-sections in this representative MRI slice of the hand. It is clear that our interpenetration removal method successfully removes penetrations without modifying the interpenetration-free sections of the boundaries of muscles.

touch each other in many places (unlike bones), the contrast in the MRI scan was simply not sufficient to discern the individual muscles. Our conclusion is that a segmentation approach is not feasible for hand muscles, and one must use a pre-existing anatomically accurate template as in our work.

5.9.1.4 Removing interpenetrations of muscles

Many hand muscles are in close proximity to one another and several are in continuous contact. One strategy to resolve contact would be to incorporate contact into our optimization (Equations 5.3 and 5.4). This approach is not very practical, as unilateral constraints are very difficult to optimize. Furthermore, such an approach couples all (or most) muscles, and requires one to solve an optimization problem with a much larger number of degrees of freedom. It is extremely

slow at our resolution; it overpowers our machine. Instead, we optimize each muscle separately. Of course, the different muscles interpenetrate each other, which we resolve as follows. For each muscle, we already positioned the MRI constraints so that they are at the muscle boundary. Therefore, observe that if our marker positioning and the solution to the optimization problem were perfect, interpenetration would be minimal or non-existent. The marker placement is relatively straightforward at the boundary between a muscle and another tissue (bone, fat, etc.) due to good contrast. However, placing markers at the boundary between two continuously colliding muscles is less precise, due to a lower MRI contrast between adjacent muscles. This is the main cause of the interpenetrations. We remove interpenetrations with FEM simulation because it produces smooth organic shape changes; alternatively, one could use geometric approaches [105]. Specifically in our work, for a pair of interpenetrating muscles, we run collision detection to determine the set of triangles of each muscle that are inside the volume of the other muscle. On each muscle, we then determine the set of tetrahedra that are adjacent to the collision area. We then slightly enlarge this set, by including any tet within a 5-ring of tets. We then run an FEM contact simulation just on these two tetrahedral sets on the two muscles. FEM simulation pushes the muscle surface boundaries apart, without displacing the rest of the muscle (Figure 5.18). We handle contact islands of multiple muscles by running the above procedure on two muscles, then for a third muscle against the first two muscles, then the fourth against the first three, and so on. Although in principle this procedure is order-dependent, the input penetrations were shallow in our examples, making the procedure relatively unaffected by the processing order.

5.9.2 Hip bone

In our second anatomical example (Figure 5.10), we apply our method to a CT scan of the human hip bone (pelvis). We obtained the template from the human anatomy model of Ziva Dynamics [146], and the CT scan from the "KidneyFullBody" medical image repository [114]. The template and the scanned hip bone differ substantially in shape, and this is successfully captured by our method.

5.9.3 Liver

In our third anatomical example, we apply our method to a CT scan of the human liver (Figure 5.11). We purchased a textured liver triangle mesh on TurboSquid [122]. We subdivided it and created a tet mesh using TetGen [51]. This serves as our "template." We used a liver CT scan from the "CHAOS" medical image repository [65]. We then executed our method to reshape the template tet mesh to match the CT scan. Much like with the hip bone, our method successfully models large differences between the template and the scanned liver. Finally, we embedded the TurboSquid triangle mesh into the template tet mesh, and transformed it with the shape deformation of the tet mesh. This produced a textured liver mesh (Figure 5.11) that matches the CT scan.

5.9.4 Hip muscle

In our fourth anatomical example, we apply our method to a MRI scan of a female human hip muscle (gluteus medius) (Figure 5.12). We obtained the data from The Cancer Imaging Archive (TCIA) [31]. The image resolution is $384 \times 384 \times 240$ with voxel spacing of 1mm, which is 2x coarser to the hand MRI dataset. We use the template mesh from the human anatomy model of Ziva Dynamics [146]. We subdivided it and created a tet mesh for it using TetGen [51]. Because the muscle is attached to the hip bone and the leg bone, we needed to first extract the bones from the MRI scan; we followed the method described in [134]. Note that the subject in the Ziva Dynamics template is male. The template and the scanned hip muscle differ substantially in shape, and this is successfully captured by our method.

5.9.5 Comparison to variational methods

We compare our method to variational shape modeling methods on an illustrative 1D example. Note that Figure 5.6 gave a comparison on 3D muscle geometry. Consider an elastic 1D line segment whose neutral shape is the interval $[0, 1]$, and study its longitudinal 1D deformation under

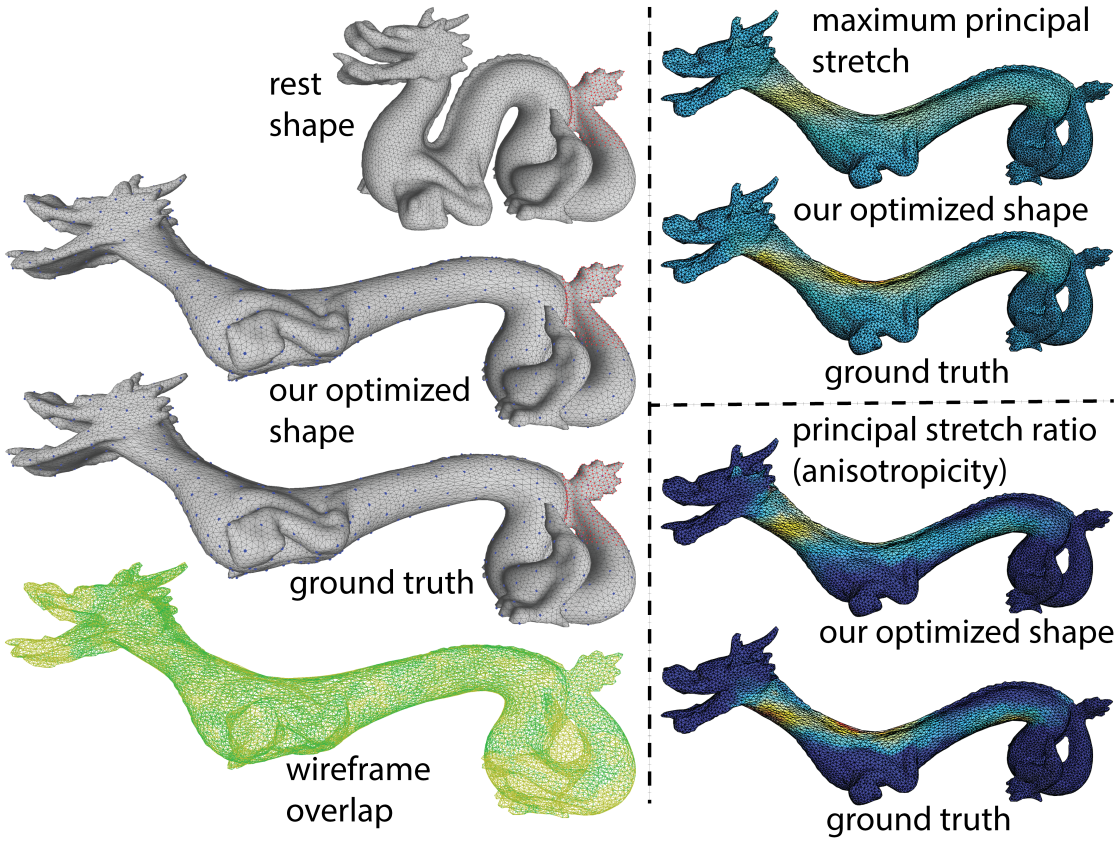


Figure 5.19: **Quantitative evaluation on ground truth.** We first performed a dynamic FEM simulation with plasticity [57], whereby the back of the dragon is fixed (red), and the head was pulled to the left with uniform force. This produced our ground truth plastic deformation gradients. We then selected 488 sparse triangle mesh vertices as landmarks and ICP markers, and ran our method to compute $\mathbf{F_p}$ and shape $\mathbf{x}$. It is evident that F_p and x closely match the ground truth.

the following setup. Let us prescribe hard attachments whereby we attach endpoint 0 to position 0, and endpoint 1 to position 2. Furthermore, assume landmarks whereby point $1/4$ is at $1/4$, and point $1/2$ is at $3/2$. Effectively in this setup, we are specifying that the subintervals $[0,1/4]$ and $[1/2,1]$ do not stretch, whereas the subinterval $[1/4,1/2]$ stretches from its original length of $1/4$ to $5/4$, (5x stretch). A variational formulation of order r is,

$$\min_{x(t)\in\mathscr{C}^r} \int_0^1 \left(\frac{d^r x}{dt}\right)^2 dt + \alpha\left(\left(x(\frac{1}{4})-\frac{1}{4}\right)^2 + \left(x(\frac{1}{2})-\frac{3}{2}\right)^2\right), \tag{5.25}$$

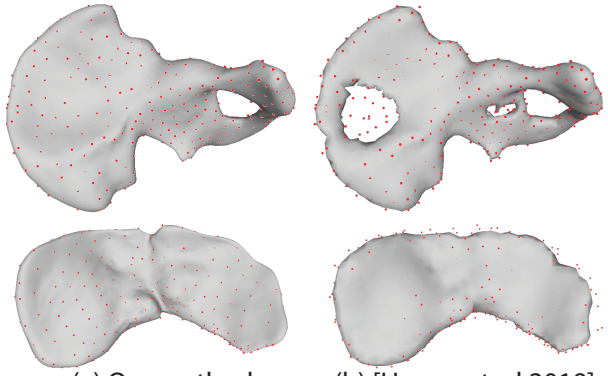


Figure 5.20: **Comparison to [54].** Top row: hip bone. Bottom row: liver. Red points are the markers from the CT scan. We used the publicly available implementation of [54] to compute the normals, followed by screened Poisson surface reconstruction using the points and computed normals [67]. We used this combination because it produced better results than running [54] directly. Our method produces shapes that more closely match the ground truth data.

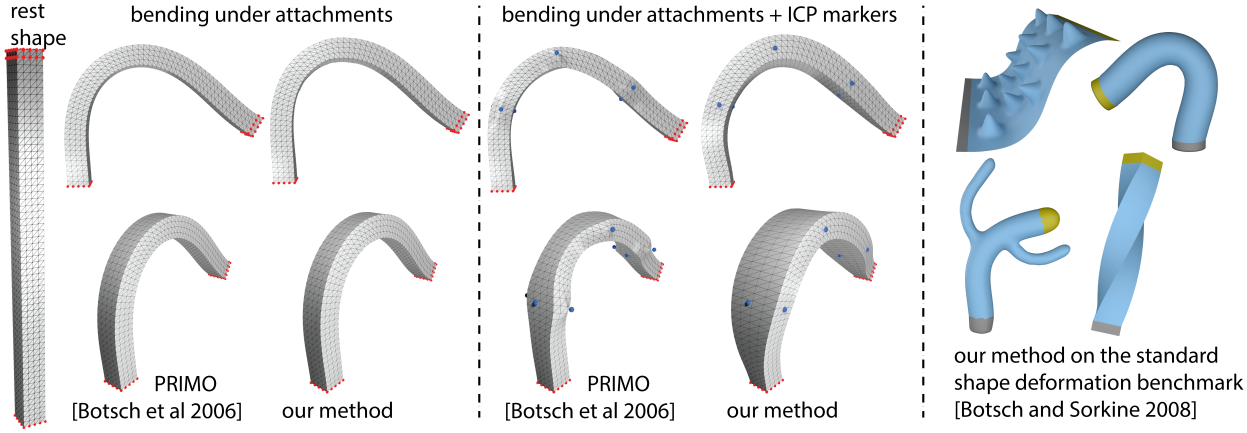


Figure 5.21: **Comparison to surface-based methods.** We compare to the "PRIMO" method [22] because this method is a good representative choice. It also has fewest artefacts in Figure 10 of the shape deformation survey [20]. Our method produces a clearly superior result in the challenging scenario where the ICP markers were placed to anisotropically stretch the beam in one transverse direction. Our method also passes the standard shape deformation benchmark [20].

where $\mathscr{C}^r$ denotes all functions $[0,1] \to \mathbb{R}$ whose derivatives exist and are continuous up to order r.

We solved these problems analytically in Mathematica for $r = 1,2,3$, each time for 3 representative values α, and compared them (Figure 5.22) to our method in 1D,

$$\min_{f_p, x(t) \in \mathscr{C}^r} \int_0^1 \ddot{f_p}^2 dt + \alpha\Big(\big(x(\frac{1}{4}) - \frac{1}{4}\big)^2 + \big(x(\frac{1}{2}) - \frac{3}{2}\big)^2\Big) + \beta \int_0^1 \big(\frac{\dot{x}}{f_p} - 1\big)^2 dt \tag{5.26}$$

$$\text{s. t. } x(t) = \arg\min_{x(t) \in \mathscr{C}^r} \int_0^1 \big(\frac{\dot{x}}{f_p} - 1\big)^2 dt. \tag{5.27}$$

Observe that, in the same vein as in 3D, we can decompose $\dot{x} = f_e f_p$, whereby scalar functions f_e and f_p and the β term are the equivalents of the elastic and plastic deformation gradients, and the elastic energy, respectively. It can be seen that our method produces a better fit to the data and a significantly less wiggly solution, compared to variational methods (Figure 5.22).

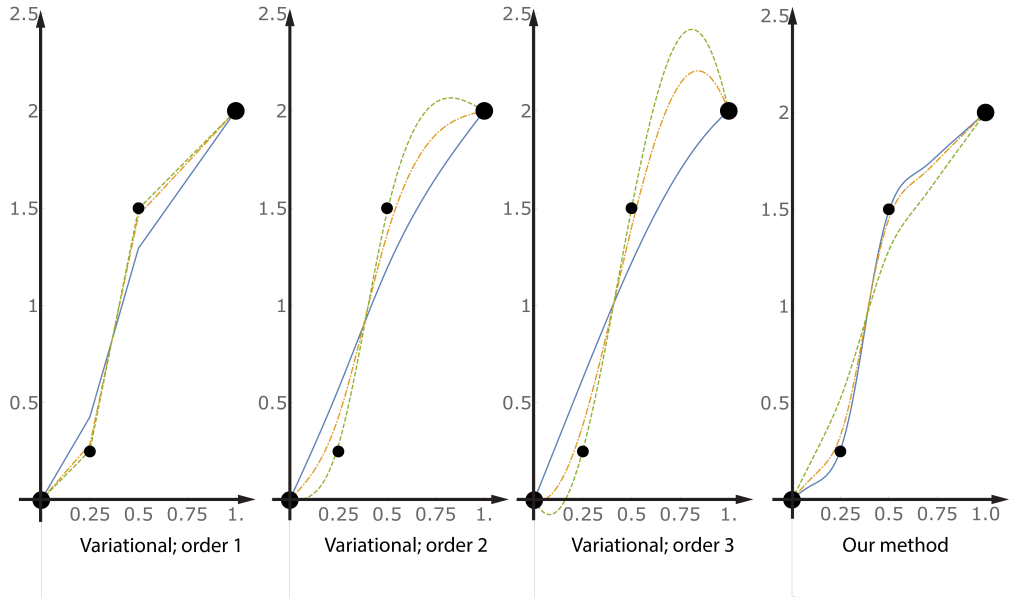


Figure 5.22: **Comparison to variational methods.** A 1D string of length 1 is attached on both ends. The left attachment is fixed. The right attachment is moved to coordinate 2. Thus, the spring stretches longitudinally while attempting to obey the two landmarks at $t = 1/4$ and $t = 1/2$. Y-axis gives the deformed 1D position of the corresponding material point on the X-axis. Big and small dots denote the attachments and landmarks, respectively. For each method, we plot the result under a few representative parameters. The blue, red, and green curves represent the different strengths of the landmarks. With variational methods, one has to either give up meeting the landmarks, or the curve becomes wiggly. Similar behavior can also be observed in 3D variational results (Figure 5.6). Our method produces a deformation profile whereby the slope (total 1D deformation gradient) on all three subintervals $[0, 1/4], [1/4, 1/2], [1/2, 1]$ approximately matches the slope implied by the attachments and landmarks (1, 5, 1, respectively). Note that the shown output curves are only $\mathscr{C}^{r-1}$ and not in $\mathscr{C}^r$ at the two juncture points $1/4, 1/2$; however, their r-th derivative is integrable and they are the optimal Cauchy limit of a score-decreasing sequence of curves in $\mathscr{C}^r$.

5.9.6 Comparison to iterative closest point methods

Penalizing the differences of affine transformations between neighboring vertices is one of the commonly used smoothness terms in "classical" ICP algorithms [5, 6, 77]. Namely, these methods

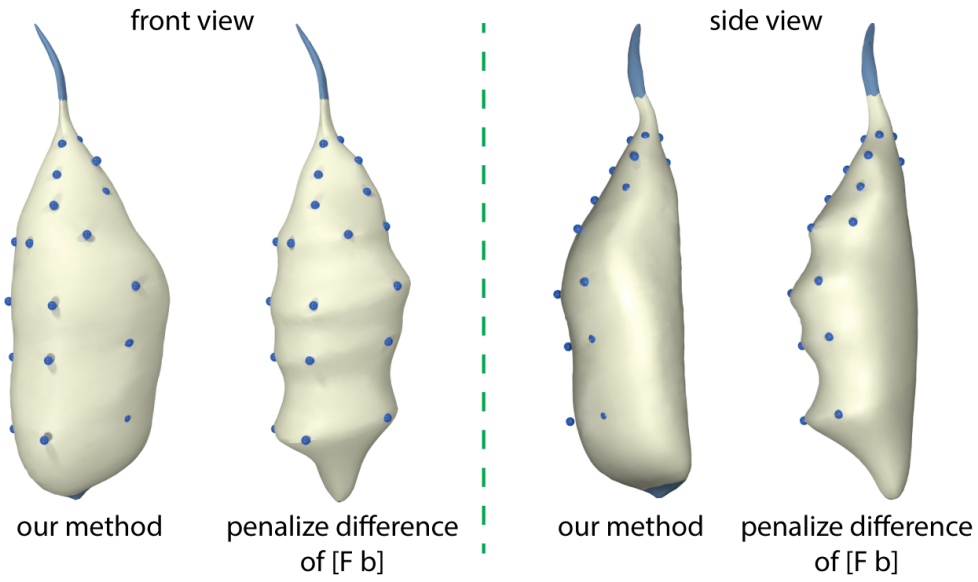


Figure 5.23: **Comparison to methods that penalize the difference of neighboring affine transformations** $[F\ b]$. These methods produce a result similar to the variational method and suffer from wiggle artefects.

optimize for a 3×4 affine transformation $A = [F\ b]$ ($x \rightarrow Fx + b$) at each vertex, and penalize an energy that is the sum of the differences in the $[F\ b]$ matrices between adjacent vertices, plus the ICP energy. This is similar to variational methods [21, 118], which penalize the difference in F (as opposed to $[F\ b]$). We provide a comparison on our hand muscle example; results are similar to variational methods (Figure 5.23). Note that "classical" ICP algorithms assume dense correspondences; they produce artefacts under sparse correspondences (Figure 5.23).

5.9.7 Comparison to ShapeOP

As shown by previous work, variational methods suffer from artefacts under large rotations [20]. One can enhance variational methods by adding local rotations to each vertex, producing the energy

$$E_{\text{bend}} = \frac{1}{\sum_i A_i} \sum_i A_i \left\| \Delta \mathbf{x_i} - \mathbf{R_i} \Delta \bar{\mathbf{x}}_\mathbf{i} \right\|^2, \tag{5.28}$$

used by [24, 1, 42]. Here, A_i is the local Voronoi area of vertex i. We used the publicly available ShapeOp library [23, 24] to implement E_{bend}. We then use E_{bend} for shape deformation by adding

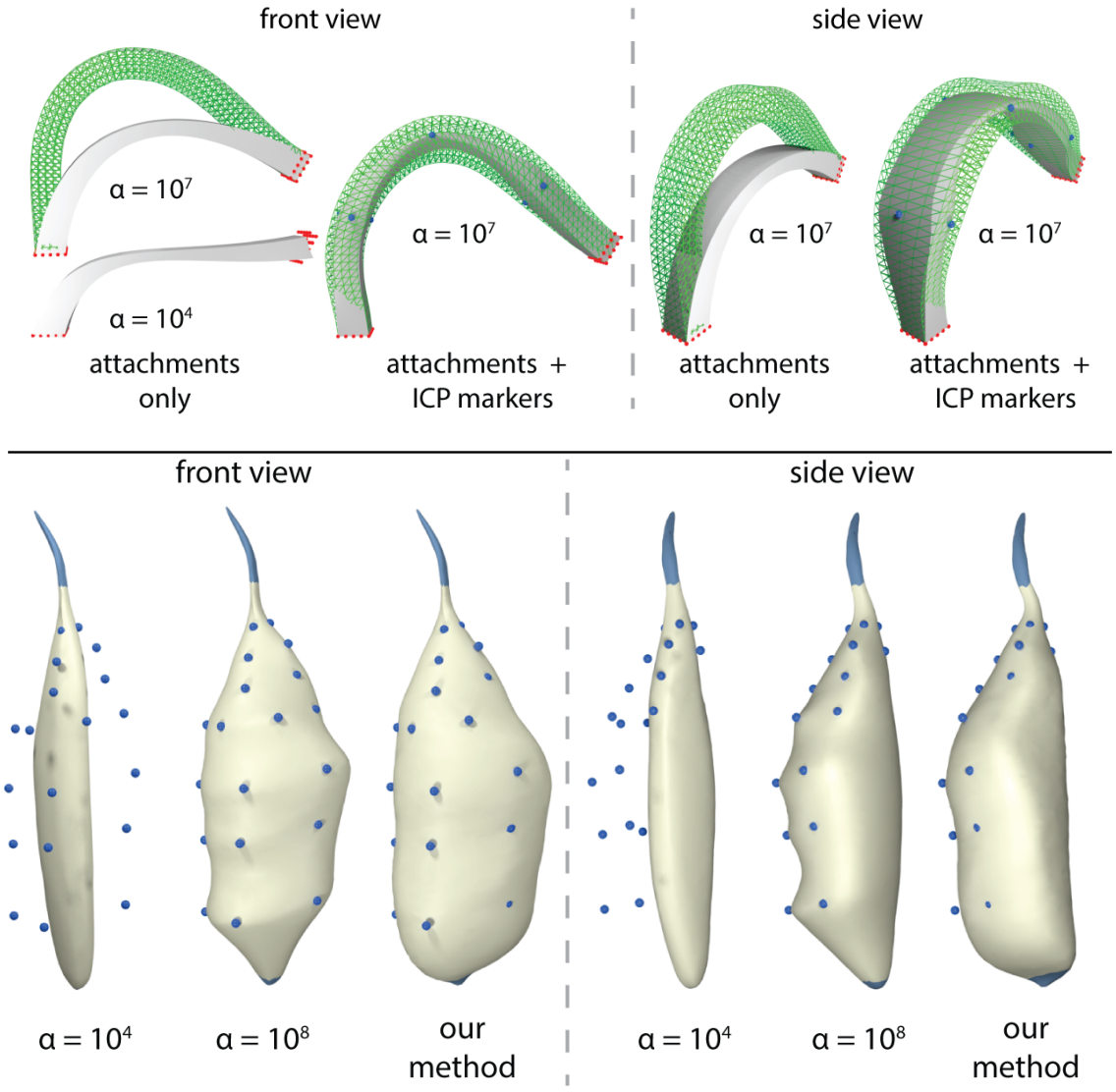


Figure 5.24: **Comparison to E_{bend}.** We deformed the surface using the ShapeOp library. Here, α is the strength of the marker constraints (attachments and ICP markers) relative to the bending energy. In the beam example, our method is presented in green wireframe and E_{bend} is shown as solid. Low values of α (such as, for example, $\alpha = 10^4$) cannot satisfy the constrains; therefore, we use $\alpha = 10^7$ and $\alpha = 10^8$ to meet the constraints in the beam and muscle examples, respectively. It can be seen that the E_{bend} method suffers from volume loss in the beam example. In addition, it fails to grow the beam (the middle is collapsed). Wiggle artefacts can be observed in the E_{bend} method on the muscle.

the ICP energy to it, and compare it to our method (Figure 5.24). The E_{bend} energy produces visible artefacts: when the beam is bent, there is a visible volume loss. When the beam is bent and forced to grow by the ICP markers, E_{bend} cannot grow the volume properly (the middle of the beam collapses). Similarly, wiggling artefacts can be observed in our hand muscle example. This

is not surprising as rotations are relatively small in this example, and therefore E_{bend} becomes the variational biharmonic energy.

5.9.8 Comparison to incremental plasticity

One idea explored in prior work is to incrementally convert "elastic" deformations into "plasticity" after each ICP iteration [101, 42]. However, doing so introduces an unwanted hysteresis-like effect. To demonstrate this, we compare our method to [42] who used two different energies combined with the "incremental plasticity" strategy. For the skull, they used an energy that penalizes volume change (Equation 1 in [42]). In the beam example, we can see that as one incrementally converts "elastic" deformation into "plastic" deformation, the bending beam undergoes a large spurious volume change. When imposing constraints that should cause the beam to inflate in its central region, the incremental plasticity method produces spikes (Figure 5.25, top left). This is because the constraints require volume to grow, but the energy penalizes volume growth. For the skin, Gietzen et al. used an energy similar to Equation 5.28 (Equation 5 in [42]), combined with "incremental plasticity." On the bending beam, the method suffers from a loss of volume and still causes the middle of the beam to collapse (Figure 5.25, top right). In the muscle example (Figure 5.25, bottom), the wiggle artefacts can be reduced by applying "incremental plasticity." However, the method introduces a vertical sharp edge at the middle of the muscle (clearly seen in the side view); our method has no such artefacts. The key difference to our method is that in the incremental plasticity method, the elasticity is explicitly converted to plasticity without any controlling criteria. Our method, however, finds **optimal** plastic strains, via optimization. For example, this makes it possible to employ minimal plastic strains when elastic energy can already do a good job.

We also compare to ShapeOp and "incremental plasticity" on the benchmark examples of [20] (Figure 5.26). The ShapeOp cylinder suffers from the same volume loss problem as the beam. In addition, E_{bend} causes self-collisions on the cactus example. "Incremental plasticity" improves the results somewhat, but it still cannot pass the benchmark.

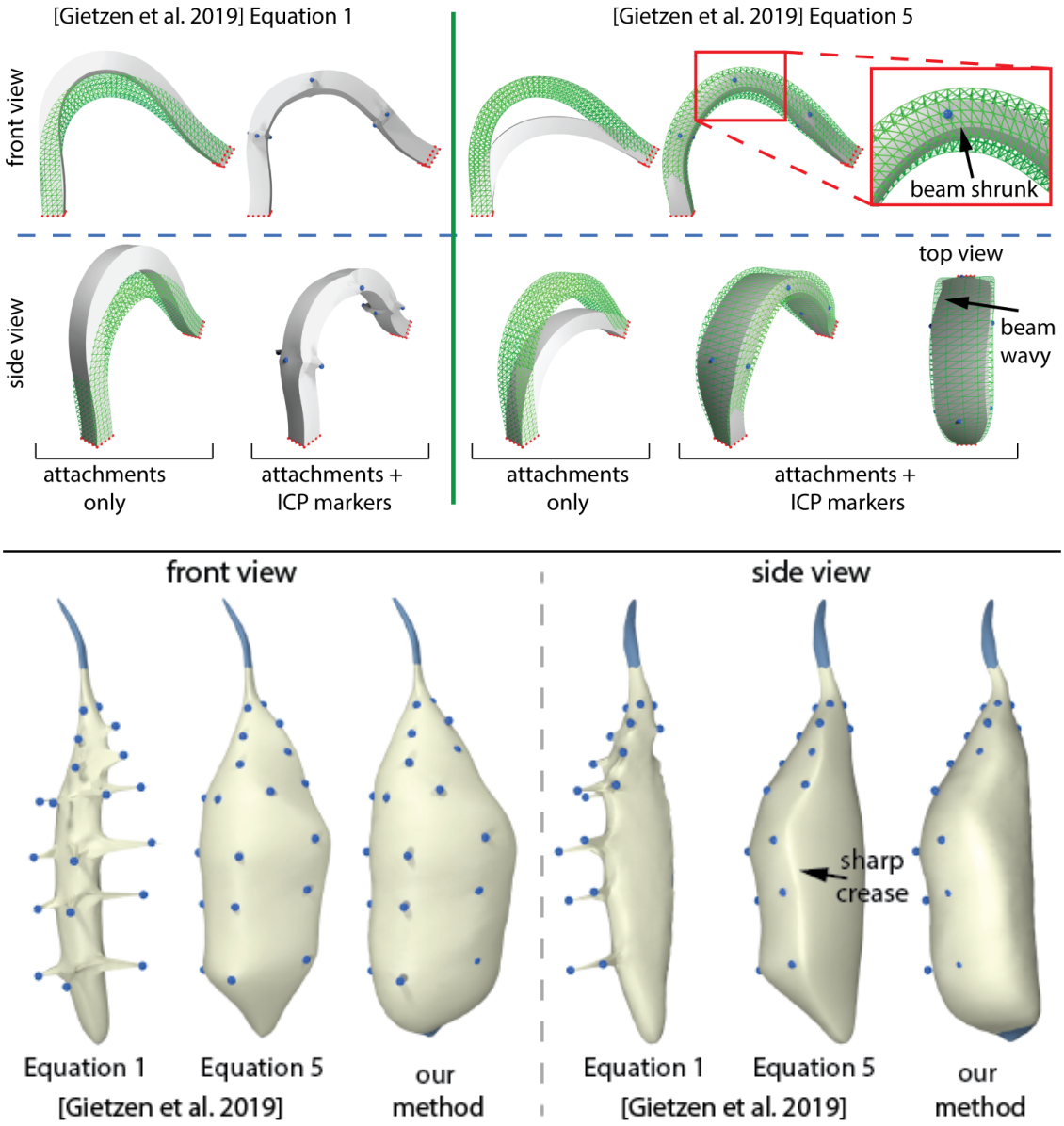


Figure 5.25: **Comparison to [42].** We compare our method to two different deformation energies used in [42]. Equation 1 penalizes the volume change relative to the previous ICP iteration. Equation 5 penalizes E_{bend} relative to the previous ICP iteration ("incremental plasticity"). On the beam example, our method is depicted via a green wireframe, whereas the compared method is depicted as a solid. It can be seen that the "incremental plasticity" method suffers from similar artefacts as other prior methods (but to a lesser degree). However, "incremental plasticity" introduces its own problems, namely volume loss and waviness (observable during bending the beam) and sharp creases (that occur while growing the muscle). To eliminate any effects of incorrect correspondence from the experiment, both examples use landmark constraints (depicted as blue spheres); the results are even more favorable to our method when using ICP constraints.

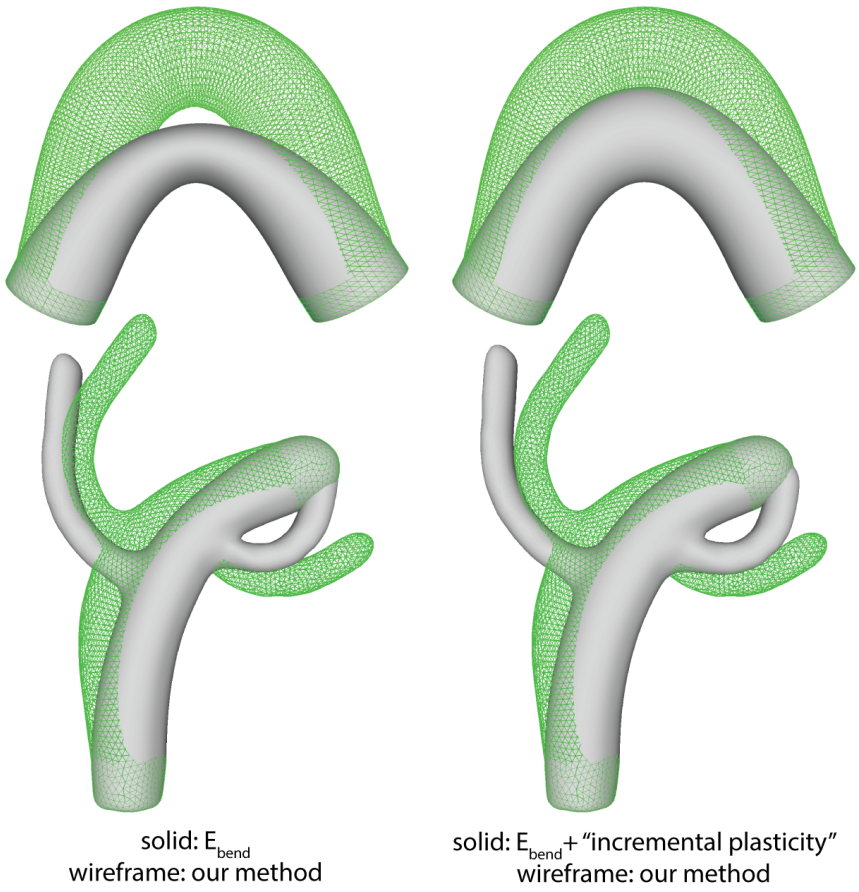


Figure 5.26: **Comparison to $E_{\mathbf{bend}}$ on the shape benchmark of [20].**

5.10 Discussion and conclusion

We gave a shape deformation method that can model objects undergoing large spatially varying strains. Our method works by computing a plastic deformation gradient at each tet, such that the mesh deformed by these plastic deformation gradients matches the provided sparse landmarks and closest-point constraints. We support both constrained and unconstrained objects; the latter are supported by giving a numerical method to solve large sparse linear systems of equations with a known nullspace. We applied our method to the extraction of shapes of organs from medical images. Our method has been designed to extract as much information as possible from MRI images, despite the soft boundaries between the different organs. We extracted hand muscles, a liver, a hip bone and a hip muscle.

Our method does not require a dense target mesh; only a sparse set of observations is needed. If a dense target mesh is available, the problem becomes somewhat easier, as one can then use standard ICP algorithms. However, medical images contain many ambiguities and regions where there is not a sufficient contrast to clearly disambiguate two adjacent medical organs; making it impractical to extract dense target meshes. We apply our method to solid objects, but our plastic strain shape deformation method could also be used for shells (cloth). Doing so would require formulating the elastic energy of a plastically deformed FEM cloth, and computing the energy gradients and Hessians with respect to the plastic parameters. The size of the small square dense matrix that we need to invert in our incremental solve is three times the number of markers. While we easily employed up to a thousand of markers in our work, our method will slow down under a large number of markers. We do not re-mesh our tet meshes during the optimization. If the plastic strain causes some tetrahedra to collapse or nearly collapse, this will introduce numerical instabilities. Although not a problem in our examples (see Figure 5.16), such situations could be handled by re-meshing the tet mesh during the optimization [15]. Our method requires a plausible template with a non-degenerate tet mesh. Re-meshing is important future work as it could extend the reach of our method, enabling one to start the optimization simply from a sphere tet mesh.

Chapter 6

Comprehensive Modeling of the Human Hand

Using MRI techniques, we were able to capture the interior of the hand. As can be seen from the MRI presented in Figure 6.1, a human hand contains multiple different organs. We cannot only find the bones, but also other organs such as muscles, tendons, and fat. We were able to model all soft tissues in the hand as a single soft tissue in the previous chapter. However, both functionalities and mechanical properties of these organs are rather different. For example. a fat tissue is very soft and is a passive tissue. On the contrary, the muscle tissue is much stiffer compared to the fat tissue and is an active tissue. Therefore, to improve the accuracy of our hand model, we model different organs separately. In addition, we have captured MRIs in multiple example poses. By using this data, we can improve the accuracy of our hand model.

We model tendons, fat, muscles, bones, and ligaments separately. As presented in Figure 6.2, we model each organ differently. In our simulation hierarchy, we first animate the bones. The method to animate bones is described in Chapter 4. On top of the bones, we perform bone fascia simulation to fill out the valleys between the bones. The simulation is achieved using a cloth solver, which is described in Section 6.3. Thereafter, we perform tendon simulation based on the bones, which is explained in Section 6.2, and we run muscle simulation, which is explained in Section 6.1. After the simulation, we treat muscles, tendons, and the bone fascia as kinematic objects. Similarly to the bone fascia, we do muscle fascia simulation. Then, we perform the joint ligament simulation. The ligaments are treated the same as muscles, which is controlled by plastic strains. Finally, we perform fat simulation. When the fat is being simulated, all other objects are

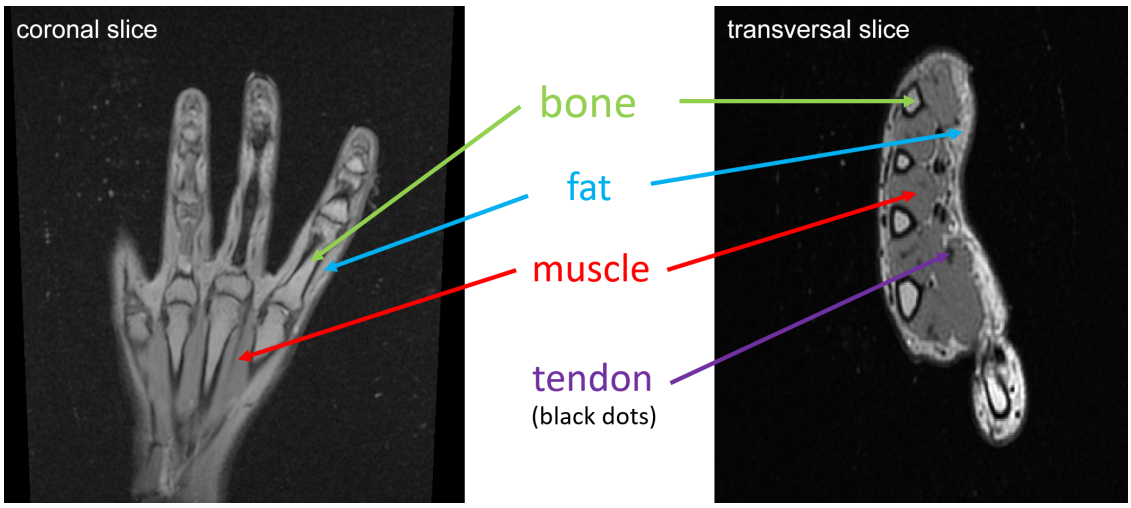


Figure 6.1: **MRI slices in the coronal plane and transversal plane.** Here, we show two MRI slices of human hand. It is evident that there are several different tissues inside the hand represented by different intensities in the MRI image.

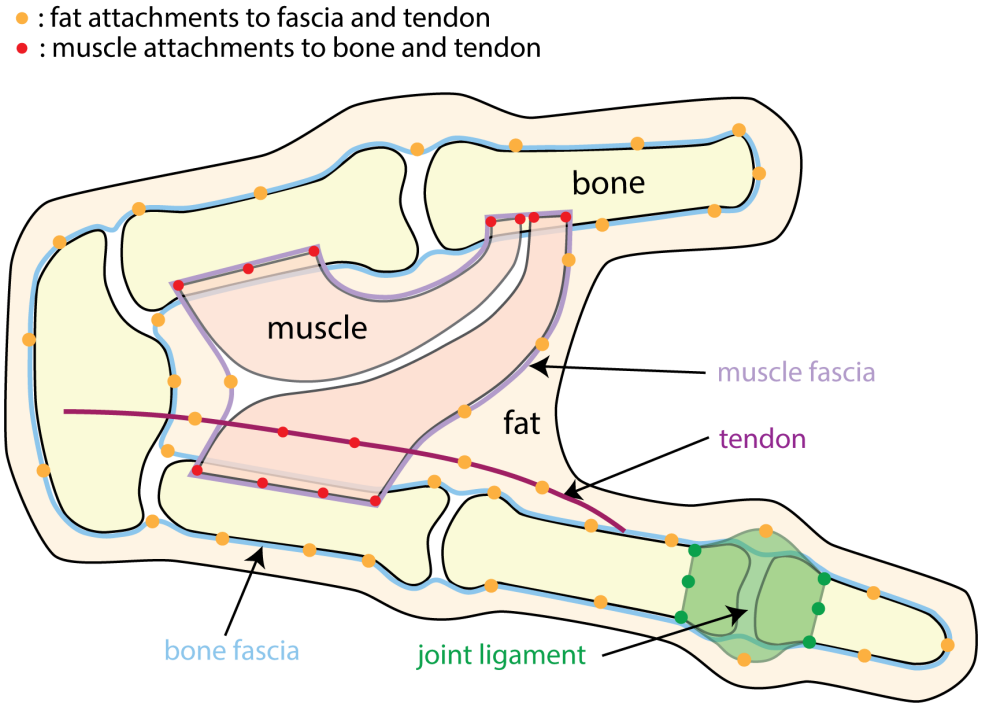


Figure 6.2: **The overview of our human hand rig.** We model tendons (dark purple), fat (flesh color), muscles (light pink), bones (light yellow), and ligaments (green) around joint regions. To improve the simulation quality, we also model two additional fascia layers: bone fascia (light blue) and muscle fascia (light pruple). The fat tissue is directly attached to fascia layers and ligaments.

kinematic. The fat layer is directly constrained to tendons, the muscle fascia, the bone fascia, and

joint ligaments. The fat layer simulation will be explained in Section 6.4. The remainder of the pipeline from simulation results to the rendering results has been discussed in Chapter 4.

6.1 Hand muscle

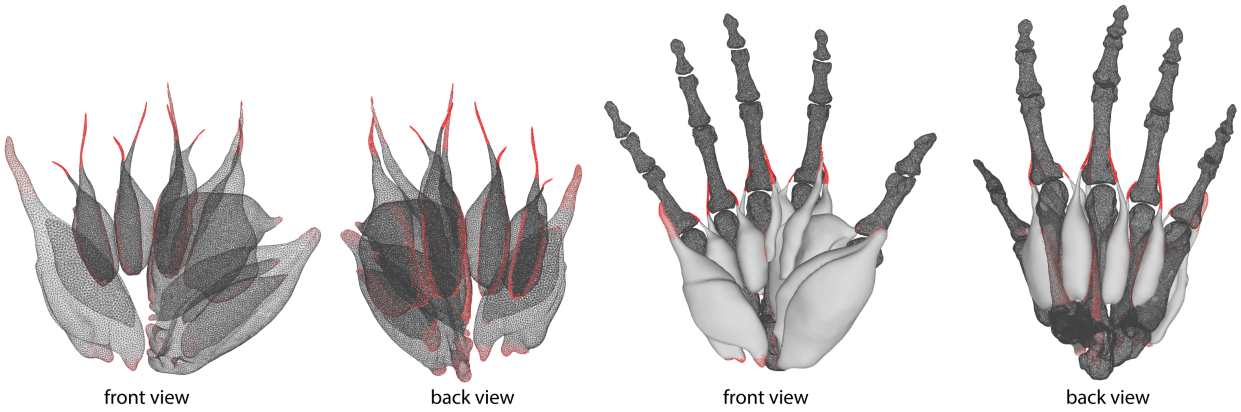


Figure 6.3: **Muscle attachments.** The muscle attachment vertices are depicted as red dots. To show the attachments clearly, the muscles are visualized using dark wireframes in the left two figures. Further, to present the relative locations to the bones, we show the bones in dark wireframes in the right two figures.

We now describe our muscle preprocessing pipeline and simulation model. We first extract the muscles at the neutral shape using the approach described in Chapter 5. To do so, we specify landmarks, attachments and ICP markers manually based on the MRI images and guided by medical literature and a medical doctor. For landmarks, we try to identify the anatomical correspondences between the MRI and the input template mesh. Further, we place ICP markers at the locations of the muscle boundary in the MRI with high confidence. For the attachments, we first specify the attachment vertices at the template mesh. Then, we extrapolate the vertex positions based on the deformation from the template bone meshes to our bone meshes. Specifically, for each attachment vertex, we find the closest patch on the template bone meshes. The patch is defined as a set of connected triangles. Then, we extract the transformation based on the deformation of the same patch from the template bone meshes to our bone meshes. Finally, the attachment vertex is transformed based on the extracted transformation. After that, we further check whether the attachment

vertex positions correctly correspond to the MRI. If not, we manually adjust them using modeling tools. Applying above steps yields the final results, as depicted in Figure 6.3. After we obtained the markers and attachment locations, we successfully extracted all hand muscles in the neutral pose, as shown in Figure 5.5. As is well known, human muscles are not passive tissues and muscle deformation is governed by Hill's model [144]. Hill's model takes muscle activation into account and combines it with passive elastic deformation. Given the muscle elastic material properties, fiber directions, and activation parameters, the model is able to deform the muscle accordingly. However, in practice, these parameters are difficult to obtain. For example, extracting the fiber directions of a muscle from MR images is usually not feasible or reliable. It requires imaging techniques like diffuse tensor imaging, which is prohibitively expensive and rarely available. It is only possible to approximate them based on the shape of the muscle [102]. These facts make Hill's model difficult to apply. Instead, we use 6-DOF plastic model from [55] as our muscle activation model.

Unlike [55], we do not embed all tissue into a single tetrahedral mesh. Instead, we model each muscle into a separate tetrahedral mesh. The first reason for doing this is that tissues are very often sliding and contacting against each other in addition to attaching to certain locations. If a single tetrahedral mesh were used, such effects are neglected. Moreover, it is much more reliable to define the pose space for each muscle individually than to define it for the entire hand. Unlike human faces, a human hand has a highly concave shape and has a large amount of flexible joints. If a single tetrahedral mesh were used, each joint would contribute a few DOFs into the pose space, thereby making the final dimension of the pose space large. Admittedly, we could define a spatially-varying pose space, meaning that different parts of the hand have different pose spaces, but an accurate and smooth separation of the human hand is not easy. Moreover, given the fact that our example poses are sparsely distributed in the pose space, the large dimension of the pose space implies lower accuracy of the interpolation.

As shown in Figure 6.4, a muscle is attached to several rigid bones. As described earlier in Chapter 4, bones undergo rigid motions around their parent bones. The attachment vertices (in

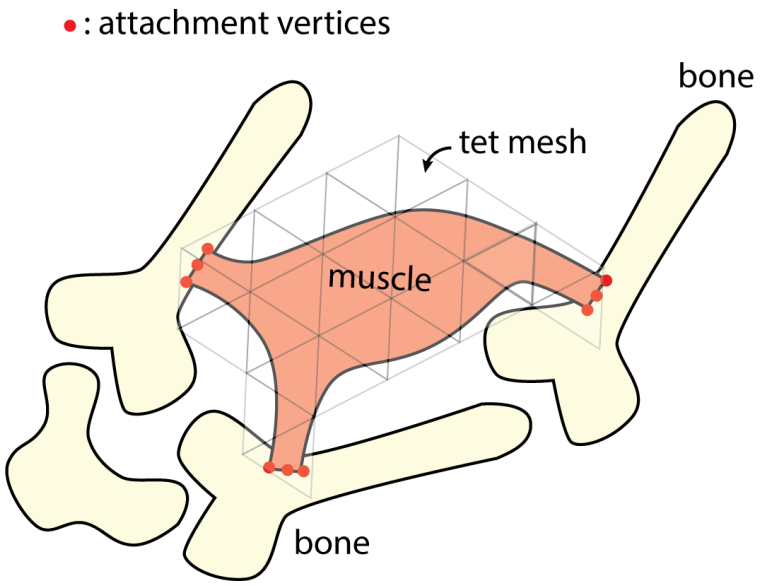


Figure 6.4: **Overview of muscle simulation.** A muscle is attached to more than one bone (or tendons). The muscle surface is embedded into a tetrahedral mesh. The muscle contraction is controlled by the plastic strain of each tetrahedron.

red) move rigidly with the attached bones. The muscle tetrahedral mesh contains plastic strains that are also controlled by the joint transformations.

6.1.1 Muscle pose space

Without loss of granularity, consider a single hand muscle. We assume we have N_p example poses. For example pose i, denote the surface mesh by S_i, and the plastic strain by P_i. Then, the next question we solve is to define the pose space for the muscle. A single muscle usually attaches to several bones and tendons at the ends of the muscle. The tendons, as described subsequently, are also controlled by the motion of the bones. Thus, we can consider that the muscle is solely controlled by the motion of the bones. The dominant motion of the bones are rotations around the parent bones and, thus, we define the pose space of the muscle by the rotations of the related bones. Take a muscle with two attached ends as an example in which each attached bone contributes one rotation. Since we do not care the global rigid transformation of the muscles, we can eliminate one of the two rotations. Consequently, the pose space of a muscle with two insertions is only controlled by a single rotation.

To represent a rotation, there are several choices: 1) a quaternion; 2) a 3D rotation matrix; 3) a 3D axis-angle vector. 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 difficult to measure the distance to the rotation matrix/quaternion of the example pose and, thus, it is difficult to generate interpolation weights for example poses. In contrast, a 3D axis-angle vector is defined in Euclidean space, so it is simple to compute the distance to each example pose. Moreover, compared to other options, a 3D axis-angle vector has the least number of dimensions. We still need to address the problem that multiple 3D vectors correspond to the same rotation. To solve this problem, we define two additional conditions when computing the corresponding 3D vector to a rotation. The first condition is that the rotation axis must form a angle of less than 90 degree with the primary rotation axis of the joint. The primary rotation axis has been computed when we generated the joint hierarchy of the skeleton. We define the rotation axis with the largest angle ranges as the primary rotation axis. By doing so, all rotation axes of the rotations in all example poses will be similar to each other. This avoids the flip of the rotation axes. The second condition is that the rotation angle must be between -2π and 2π, because any angle +/- 2π is equivalent to itself.

With these definitions, we are able to define the pose space of a muscle. Assume a muscle has N_e insertions attaching to the bone. Then, the dimension of the pose space is $3(N_e - 1)$. If a muscle is attached to the tendon, then we treat the bones that control the tendon as the bones that also control the muscle in addition to the bones that the muscle is directly attached to. Once the bones are determined, we first extract the current rotation of the joint that each bone is rotating around. Then, we remove one joint that is the common ancestor of all joints to eliminate the global rotation. Finally, we convert all rotations into 3D vectors using the method mentioned in the previous paragraph, and combine them into a pose-space vector a_i for example pose i. We repeat this process for all example poses for all muscles.

During the simulation, we encounter an arbitrary pose-space vector a. Then, the interpolation weights w_i for example pose i are defined as

$$w_i = \frac{\phi(a, a_i)}{\sum_j^{N_p} \phi(a, a_j)} \tag{6.1}$$

$$\phi(a, b) = \frac{1}{\|a - b\|}. \tag{6.2}$$

Thus, the final plastic strain P of the muscle is computed as

$$P = \sum_i^{N_p} w_i P_i. \tag{6.3}$$

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.

6.1.2 Muscle plastic strain extraction

As we use plastic strains to control the muscle activation, we need to have plastic strains for all example poses defined on a single tetrahedral mesh for each muscle. To obtain the plastic strain P_i at example pose i, we start from the volumetric mesh at the neutral pose. We could simply apply the method proposed in Chapter 5 for each example pose separately, as it can directly output the plastic strain. However, such an approach has several issues. First, the method does not provide the smoothness of the plastic strain between example poses. Our objective is to achieve the property that if the pose-space vectors of two example poses are close to each other, the corresponding plastic strains must be similar to each other. As the muscle boundaries are sparse markers, the surface areas where there are no markers are only deformed in accordance with the spatial smoothness but not the pose-space smoothness. We observed that if we did not apply the pose-space smoothness, the muscle simulation produced non-smooth motions when transitioning from one example to another one. Second, when optimizing the shape to match the sparse markers, the optimization parameters—like attachment strengths—may not always be the same for a single muscle across

all example poses, for modeling purposes. Therefore, applying the plastic strains obtained separately using the method described in Chapter 5 to each example pose, the runtime simulation may not reproduce the same shape. In general, such coupling between simulation methods and modeling/segmentation methods is not desirable. Ideally, we should be able to obtain the muscle shape using any applicable method, and then generate smooth pose-varying plastic strains across the entire hand range of motion using such data.

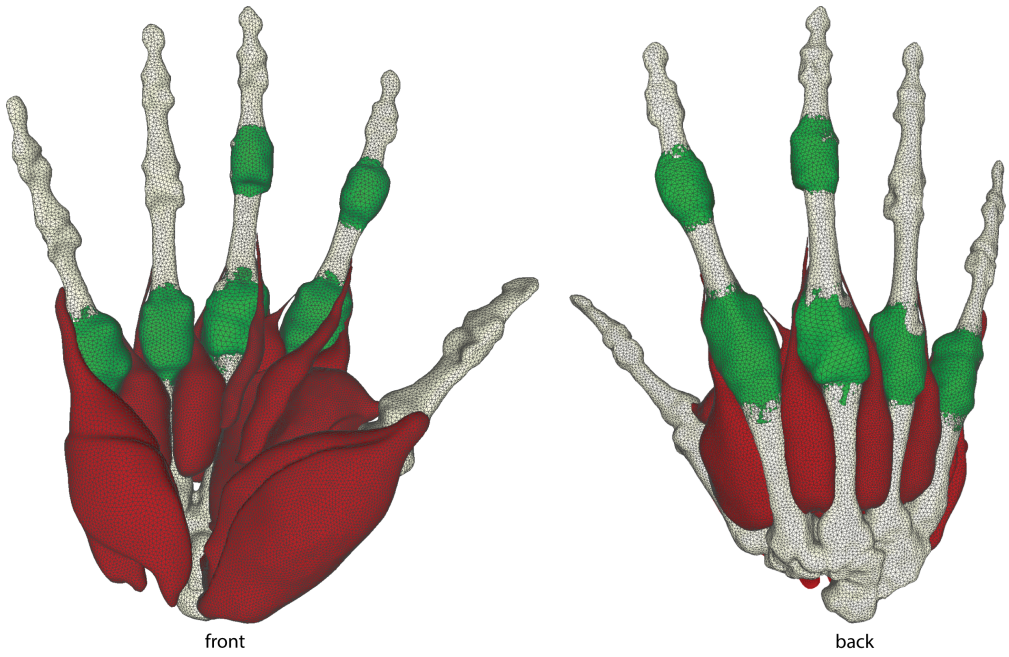


Figure 6.5: **The muscle and joint ligament surface meshes.** The muscles are presented in red and the joint ligaments are presented in green.

To obtain smooth muscle plastic strains across the range of the motion of the hand, we proposed the following procedure. Assume that the plastic strains for the neutral pose are identity matrices. At the beginning of the procedure, only the neutral pose is the processed example pose.

1: **Select a non-processed example pose.** Select the non-processed example pose that is the most different from all already processed example poses.

2: **Prepare the initial guess for the method of Chapter 5.** If there are two or fewer processed example poses, use the neutral shape as the initial guess for the selected example pose (explained later in Section 6.1.3). Otherwise, use the processed example poses to initialize the

muscle simulation model (explained later in Section 6.1.4). Then, simulate the muscle to the selected example pose (explained later in Section 6.1.5), which yields the initial tet mesh of the muscle.

3: **With the initial shape, apply the method of Chapter 5 to generate the tet mesh of the muscle.** During this process, we manually place markers on the boundary of the muscle in MRI based on the medical knowledge, as explained in Chapter 5. These markers will be reused in further iterations of the algorithm.

4: **Add the selected pose to the processed example poses and go back to step 1.**

It can be seen that each iteration (four steps) involves simulating the muscle to the example pose, and then optimizing the muscle shape, which is time consuming. To extract the muscle shapes in all N_p example poses, at least N_p iterations are required, because each pose must at least be processed once. After N_p iterations, we can continue repeating it by flagging all processed poses as unprocessed. This time, we start from the first example pose to the last example pose. Each time we select a pose, we treat the other ones as processed poses when generating the initial guess. We can continue repeating these iterations as much as we want. In our demos, we use six example poses. We apply 1×6 iteration for nine muscles and six ligaments and 2×6 iterations for the remaining six muscles. At the end of this procedure, we obtain the muscle simulation rig for the hand, which includes the plastic strains for each muscle in each example pose and the corresponding pose-space vector. By applying our procedure, we successfully processed 15 muscles (red) and 6 joint ligaments (green), as illustrated in Figure 6.5.

6.1.3 Initial guess of muscle tet mesh in non-neutral example poses

When the number of processed example poses is two or fewer, we create the initial guess directly from the shape in the neutral pose. To do so, we first create a muscle fiber field by solving a Laplace equation [102]. We assume that a muscle only contracts along the fiber field when it switches the pose from the neutral pose to any other pose. Then, we define $R\ \mathrm{diag}(a,1,1)\ R^T$ as the plastic

strain of each tetrahedron for the muscle, where R is the local frame defined by the muscle fiber direction and a is the muscle contraction parameter [102]. By applying the same idea as described in Chapter 5, we first solve a simple optimization problem

$$\arg\min_{\mathbf{a},\mathbf{x}} \|\mathbf{L}\mathbf{a}\|^2 + c_1 E_{\text{elastic}}(\mathbf{a},\mathbf{x}) + c_2 E_{\text{att}}(\mathbf{x}), \tag{6.4}$$

where $\mathbf{a}$ is the muscle contraction parameters for all tetrahedra, $\mathbf{x}$ is the shape of the muscle, E_{elastic} is the elastic energy, and E_{att} is the attachment energy. This problem is a simplified version of the optimization problem defined in Chapter 5, because (1) it uses a much smaller space for the plastic strains, (2) it does not attempt to match the sparse markers, and (3) we treat the static equilibrium constraint as a penalty term in the objective function. The reason we can do (3) is that there are no sparse markers. Therefore, the unconstrained optimization problem can be solved simply by using Newton's method. Intuitively, this objective function attempts to ascertain the muscle contraction along the transversal direction, so that, in the static equilibrium under the contraction, the attachment energy to the bones is small. The DOFs provided by the contraction $\mathbf{a}$ are necessary, because the shape of a straight muscle would bend, as opposed to contracting, when two insertions come closer to each other. This is due to volume preservation terms in the elastic energy.

To make the problem even simpler, we perform model reduction over vector $\mathbf{a}$. We manually select a few tetrahedron (usually 3-4) on the tetrahedral mesh and treat them as "handles," meaning that the contractions of these tetrahedra will control the entire $\mathbf{a}$. Then, we generate the subspace basis $\mathbf{U}$ by calculating bounded bi-harmonic weights [59]. Then, the contraction $\mathbf{a} = \mathbf{U}\hat{\mathbf{a}}$, where $\hat{\mathbf{a}}$ is the subspace quantity, and our optimization problem becomes

$$\arg\min_{\hat{\mathbf{a}},\mathbf{x}} \|\mathbf{L}\mathbf{U}\hat{\mathbf{a}}\|^2 + c_1 E_{\text{elastic}}(\mathbf{U}\hat{\mathbf{a}},\mathbf{x}) + c_2 E_{\text{att}}(\mathbf{x}). \tag{6.5}$$

After we solve it, the initial guess of the muscle shape is given by $\mathbf{x}$.

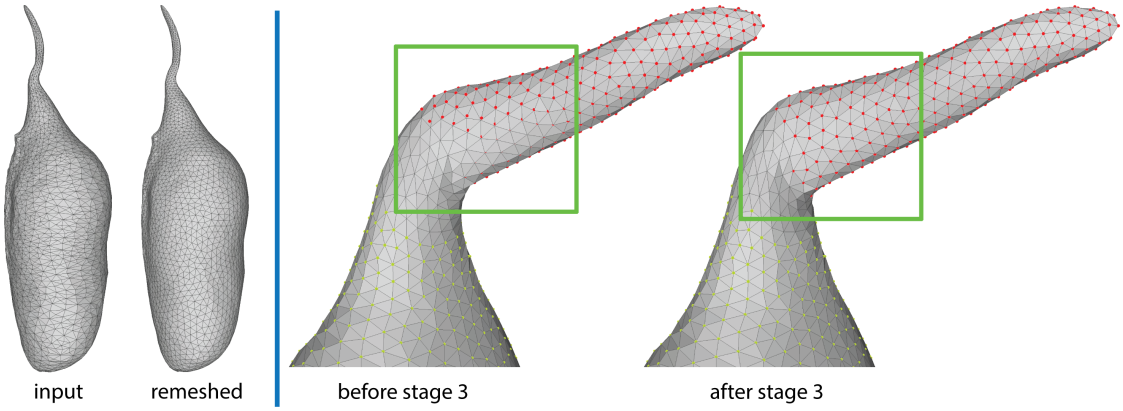


Figure 6.6: **Muscle preprocessing results.** In step 2, we remesh the muscle surface mesh. As demonstrated on the left, remeshing improves the triangle quality. On the right of the figure, the target attachment locations are depicted as red dots, and the green dots are constraints to preserve the shape of the muscle. The target attachment locations are obtained by rigidly transforming the positions of the attachment vertices at the neutral pose to the target example pose. By applying step 3, we make the surface vertices better match the attachment locations.

6.1.4 Generating muscle volumetric plastic strains in example poses

Before we can run muscle simulation with pose-varying plastic strains, we need to generate volumetric plastic strains from the example muscle surface meshes that have been registered to each other across example poses. The following are the steps to achieve it.

Step 1: Compute pose-space vectors for all existing example poses. This step has been described in Section 6.1.1.

Step 2: Remesh example surface meshes (if needed) for improving the mesh quality. During the muscle modeling, the mesh quality may decrease on account of matching the sparse markers. This step improves the mesh quality, which leads to a better-quality tetrahedral mesh and simulation results (Figure 6.6 left). To keep the surface meshes registered across all example poses, we only remesh the surface mesh at the neutral pose. Then, we embed the remeshed surface to the input surface mesh and deform it to all other non-neutral example poses. This creates registered remeshed surfaces in all example poses.

Step 3: Eliminate muscle-bone attachment mismatches incurred during the muscle modeling process. During the muscle modeling, the attachment vertices may deviate from the target attached locations in order for the surface to match the markers better (Figure 6.6 right). Even though these errors are very small (usually about 1mm), they must be eliminated to the furthest extent possible. Note that if the error is zero, there is no attachment force in the example pose. Under static equilibrium, the elastic forces will also be equal to zero because there are no other forces. Therefore, the resulting shape under static equilibrium is only deformed by the example plastic strain.

To achieve this zero error, we keep all muscle surface vertices that are further than d edges from the attachment vertices fixed (green dots in Figure 6.6). We use $d = 4$ for our muscle simulation. In addition, we constrain all attachment vertices to the exact target locations (red dots in Figure 6.6) obtained by rigidly transforming the neutral pose positions of the attachment vertices. Then, we define a smoothness energy and solve for positions of the other vertices,

$$\arg\min_{x} \quad cE_{\text{fixed}}(x) + cE_{\text{att}}(x) + E_{\text{smooth}}(x), \tag{6.6}$$

where c is a large constant, $E_{\text{fixed}}(x)$ and $E_{\text{att}}(x)$ penalize the mismatch of fixed vertices and attachment vertices, and $E_{\text{smooth}}(x)$ defines the smoothness of the shape. We typically use bi-Laplacian smoothness of the surface positions or bi-Laplacian smoothness of the deformation gradients.

Step 4: Resolve the interpenetrations between muscles. As we extract the muscles separately, it is inevitable that muscles may be in contact with each other. Section 5.9.1.4 gives a method to procedurally resolve the contacts muscle by muscle. However, we found that it is too slow to process all muscles in all example poses. It takes hours to resolve the contact in a single example pose. This is because the volumetric meshes of muscles 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. In the end, we found that using ShapeOp as a

replacement of the elastic energy dramatically improves performance [23]. Specifically, we use ShapeOp bending energy as our smoothness energy [24], and then we add a sliding constraint for each vertex-triangle colliding pair. In our experiments, it takes less than 10 minutes to resolve the inter-contacts between muscles for each example pose.

Step 5: Given the cleaned-up and registered surface meshes for all example poses, find volumetric plastic strains to satisfy both spatial smoothness and pose-space smoothness. At first glance, it appears that we can modify our method from Chapter 5 in the following manner

$$\underset{\mathbf{s_i},\mathbf{x_i}}{\arg\min} \quad \sum_{i}^{N_p} \left(||\mathbf{L}\,\mathbf{s_i}||^2 + \alpha\mathscr{E}_{\mathrm{MI}}(\mathbf{x_i}) + \beta\mathscr{E}_{\mathrm{a}}(\mathbf{x_i})\right) + \gamma||\mathbf{L_{ps}}\,\mathbf{s}||^2, \tag{6.7}$$

$$\text{st.} \;\; \mathbf{f}_{\mathrm{e}}\left(\mathbf{F_p}(\mathbf{s_i}),\mathbf{x_i}\right) + \mathbf{f}_{\mathrm{a}}(\mathbf{x_i}) = 0, \text{for each} \;\; i = 1,2,\ldots,N_p \tag{6.8}$$

where $\mathbf{L_{ps}}$ is a pose-space Laplacian matrix. We can build a graph in the pose space. For each example pose-space vector, we find m closest example pose-space vectors as the neighbors. Parameter m is set to be equal to the number of example poses - 1. Then, we can build a graph and define $\mathbf{L_{ps}}$ as the graph Laplacian matrix. Therefore, $||\mathbf{L_{ps}}\,\mathbf{s}||^2$ can be rewritten as $\sum_i^{N_p}||\mathbf{s_i} - \sum_{j\in N(i)} w_{ij}\mathbf{s_j}||^2$. To solve this, we adopt the idea of the block-gradient descent method. In each iteration, we randomly select an integer $k \in [1,N_p]$ and fix all $\mathbf{s_j},\mathbf{x_j}$ for $j \neq k$. Then, the problem in each iteration becomes

$$\underset{\mathbf{s_k},\mathbf{x_k}}{\arg\min} \quad ||\mathbf{L}\,\mathbf{s_k}||^2 + \alpha\mathscr{E}_{\mathrm{MI}}(\mathbf{x_k}) + \beta\mathscr{E}_{\mathrm{a}}(\mathbf{x_k}) + \gamma||\mathbf{s_k} - \bar{\mathbf{s}}_\mathbf{k}||^2, \tag{6.9}$$

$$\text{subject to:} \quad \mathbf{f}_{\mathrm{e}}\left(\mathbf{F_p}(\mathbf{s_k}),\mathbf{x_k}\right) + \mathbf{f}_{\mathrm{a}}(\mathbf{x_k}) = 0, \tag{6.10}$$

where $\bar{\mathbf{s}}_\mathbf{k} = \sum_{j\in N(k)} w_{kj}\mathbf{s_j}$. Since $\mathbf{s_j}$ are fixed, $\bar{\mathbf{s}}_\mathbf{k}$ is fixed. Closely examining the objective function, we can see that this formulation produces artifacts. Remember that $\mathscr{E}_{\mathrm{MI}}(\mathbf{x_k})$ and $\mathscr{E}_{\mathrm{a}}(\mathbf{x_k})$ are sparse observations. They sparsely specify that certain points on the surface of the muscle must reach the target positions. On the other hand, $||\mathbf{s_k} - \bar{\mathbf{s}}_\mathbf{k}||^2$ implies that the objective function attempts

to match the plastic strains of the entire mesh to given target plastic strains. This implies that it seeks a concrete target shape, which is a dense observation. Therefore, the dense term $||\mathbf{s_k} - \bar{\mathbf{s}}_\mathbf{k}||^2$ conflicts with the sparse terms $\mathscr{E}_{\mathrm{MI}}(\mathbf{x_k})$ and $\mathscr{E}_{\mathrm{a}}(\mathbf{x_k})$. We have implemented this approach and verified experimentally that the method does not work. In order for the pose-space smoothness to be effective, the weight γ has to be sufficiently large (¿= 1.0). The weight α must also be sufficiently large compared with γ because our top priority is to match the ICP markers. We observed that there are local tiny bumps around the sparse ICP markers. This happens because the ICP marker positions conflict with the target shape represented by the dense plastic strain $\bar{\mathbf{s}}_\mathbf{k}$. The conflict is unavoidable. To avoid this problem, we have to use a dense observation of the ICP markers. However, our method only supports sparse ICP markers due to the time complexity. Therefore, this approach is not feasible.

The next idea (which we also abandoned) is to use deformation transfer [118]. This is also commonly used for anatomy transfer [102]. In this method, we use the resulting muscle surface in each example pose generated by previouse steps. The formulation is essentially the same as that in Equation 6.7 except that there is no elastic deformation.

$$\underset{\mathbf{x_i}}{\arg\min} \quad \sum_{i}^{N_p} \left(||\mathbf{L}\,\mathbf{F}(\mathbf{x_i})||^2 + \alpha \mathscr{E}_{\mathrm{MI}}(\mathbf{x_i}) \right) + \gamma ||\mathbf{L_{ps}}\,\mathbf{F}(\mathbf{x}))||^2, \tag{6.11}$$

where $\mathbf{F}(\mathbf{x})$ is the deformation gradients of $\mathbf{x}$, and $\mathscr{E}_{\mathrm{MI}}(\mathbf{x_i})$ are now dense correspondences. Here, $\mathbf{F}(\mathbf{x}) = \mathbf{G}(\mathbf{x} - \bar{\mathbf{x}})$, where $\mathbf{G}$ is the gradient operator matrix and $\bar{\mathbf{x}}$ is the rest position of the volumetric mesh. As we can see, the deformation gradient $\mathbf{F}$ is a linear function of the position $\mathbf{x}$. The resulting $\mathbf{x}$ can then be converted to plastic strains. Because we are using isotropic hyper-elastic materials, the plastic strain is essentially the symmetric matrix of the polar decomposition of the deformation gradient F for each tetrahedron. However, this method cannot handle rotations well, leading to a large amount of inverted deformation gradients. To solve this problem, we could add a non-linear inequality constraint that guarantees that the determinant F of each tetrahedron is positive, that is, $\det(F) > \varepsilon$, where ε is a very small positive number such as 0.02. Then, we can solve the

optimization using the interior-point method. Nonetheless, this method creates extreme plastic strains whose determinants constantly hit the boundary ε and whose eigenvalues span widely from ε to 15.2. This causes problems when interpolating the plastic strains during simulation. Moreover, the interior-point method cannot always find a solution for all the muscles. To address the problems due to the rotations, we replaced the Laplacian smoothing energy from deformation transfer with a standard elastic energy [110]. By tuning the resolution of the tetrahedral mesh, the interior-point method could find a solution for every muscle. Nonetheless, this method still created extreme plastic strains. This is because an elastic energy penalizes the growth of the object, while the muscles are growing/shrinking in different example poses.

To address all mentioned issues, we define the following energy:

$$\underset{\mathbf{x_i}}{\arg\min} \quad \sum_{i}^{N_p} \left(||\mathbf{L}\mathbf{S}(\mathbf{x_i})||^2 + \alpha \mathscr{E}_{\mathrm{MI}}(\mathbf{x_i}) \right) + \gamma ||\mathbf{L_{ps}}\mathbf{S}(\mathbf{x}))||^2, \tag{6.12}$$

where $\mathbf{S}(\mathbf{x})$ is a function that extracts a symmetric matrix by performing polar decomposition on each deformation gradient. Under this definition, the objective function does not penalize the growth of the object and is not affected by the rotation. Note that $\mathscr{E}_{\mathrm{MI}}(\mathbf{x_k})$ uses a dense correspondence to the input shape, which is the result of Step 4. We use the same idea to compute $\mathbf{L_{ps}}$ as the first method, that is, a graph Laplacian matrix. To solve the optimization problem, we use block-gradient descent. In each iteration, we randomly select an integer $k \in [1, N_p]$ and fix all $\mathbf{s_j}, \mathbf{x_j}$ for $j \neq k$. Then, the problem becomes

$$\underset{\mathbf{x_k}}{\arg\min} \quad ||\mathbf{L}\mathbf{S}(\mathbf{x_k})||^2 + \alpha \mathscr{E}_{\mathrm{MI}}(\mathbf{x_k}) + \gamma ||\mathbf{s_k} - \bar{\mathbf{s}}_\mathbf{k}||^2 \tag{6.13}$$

at each iteration. The resulting $\mathbf{x}$ can then be converted to plastic strains, as explained earlier. To perform optimization efficiently, we first solve N_p separate optimization problems without the pose-space smoothness term. The solution for each example pose is used as the initial $\mathbf{x}$ to our optimization problem. Then, we begin our block-gradient descent. We do 15 iterations for all our muscles. Energy 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.

6.1.5 Muscle simulation

We perform dynamic simulation for our muscles. We do not need the dynamic motion of the muscle, so we use a simple and stable integrator, namely implicit backward Euler. During each timestep, we first calculate the bone rotations and convert them to the pose-space vector for each muscle. Based on the pose-space vector, we compute the plastic strains for each muscle using Equation 6.3. In addition to the plastic strains, we apply four different types of constraints during the simulation to mimic the muscle biomechanical behaviors. They are (1) muscle attachments to tendons and bones; (2) muscle contacts to bones; (3) muscle inter-contacts and self-contacts; and (4) muscle inter-slidings. The first three are easily understandable. Muscles are attached to the bones and tendons. Muscles also cannot penetrate the bones, other muscles, and themselves. In addition to these facts, we found that muscles are also sliding against neighboring muscles. To model this effect, we add the sliding constraints between muscles. We also tested what happens if there are no sliding constraints: large empty space appears between neighboring muscles, which is unrealistic. Due to these constraints, we treat all muscles as a single simulation system and simulate them together.

For joint ligaments, we process them using the same pipeline as muscles. For simulation, they are also treated the same as muscles, except that we do not model contacts and sliding between them. Therefore, we simulate each joint ligament separately.

6.2 Hand tendons

Tendons are important not only because they control the motion of the hand, but also because they affect the appearance of the hand. In this thesis, we only model tendons that are attached

to the muscle, because they drive the muscle deformations. Tendons are often beam-shaped and are always black in the MRI image, thereby giving them good contrast to the neighboring tissues. However, tendon extracts is difficult in practice because of the limited resolution of the MRI. For such reason, thin tendons are often impossible to extract. We only model four tendons from the end of the flexor digitorum superficialis muscles for index finger, middle finger, ring finger, and pinky finger. because some muscles are attached to these tendons. Because they are thick and the contrast is clear, they are easily extracted from MRI.

6.2.1 Tendon simulation

Sachdeva et al. modeled hand tendons biomechanically [100], but the complex model requires many biomechanical constraints, such as attachment to the bones and sliding through the pulleys. In their work, these structures are obtained manually by referencing the medical literature, but they are difficult to discern from MRI; hense we did not adopt their method.

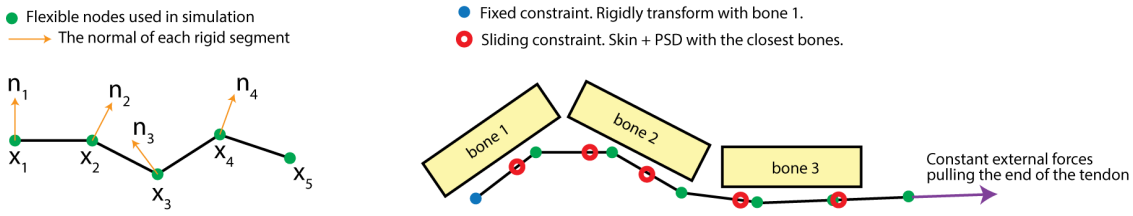


Figure 6.7: **Tendon simulation model.** The left of the figure presents our rod simulation model. We break the rod into linked rigid segments. Each segment is represented by the positions of two end vertices and a normal showing the orientation. The normal (orange) is perpendicular to the segment. The tendon in our system is simulated using such a rod. The rod goes through a few locations (red circles) that are skinned to the closest bones. One end of the tendon is attached to the bone with a fixed constraint. In addition, we apply a constant force (purple) at the end of the tendon to stretch it as much as possible without invalidating the constraints.

Similarly to [100], we model tendons as rods. We could use a complex Lagrangian coordinate-based rod simulation model used in [100], but we found that it is much simpler and sufficient to use a Euclidean coordinate-based model. Our tendon model captures the following effects: (1) the length of the tendon is constant and (2) the tendon does not twist. As illustrated in Figure 6.7 (left), we first discretize the tendon into N small rigid segments. Each rigid segment i is controlled by the

vertex positions x_i, x_{i+1} of two ends and the normal of the segment n_i. Normal n_i is perpendicular to the segment i. The simulation flexible DOFs are x_i and n_i.

As presented in Figure 6.7 (right), one end of the tendon (blue) is attached to the closest bone and is rigidly transformed with the attached bone (bone 1). The other end of the tendon is pulled by a constant external force (purple) in the transverse direction to mimic the fact that the muscle of the forearm is pulling the tendon. The tendon is constrained by a series of points (shown in red circles) near the bones. The tendon passes through these points, that is, the tendon is sliding against them. Given the k sliding locations, constant external forces, and the fixed attachments, we define our tendon static simulation as the following optimization problem:

$$\underset{x,n,t}{\arg\min} \overbrace{\sum_{i\in S}\left((1-t_i)\,x_i + t_i x_{i+1} - \bar{x}_i\right)^2}^{\text{sliding constraints}} - \overbrace{c_1 x_{N+1}^T f}^{\text{pulling force}} - \overbrace{c_2 \sum_{i=1}^{N-1} n_i^T n_{i+1} - c_3 n_1^T \bar{n}_1}^{\text{twisting}} \tag{6.14}$$

$$\text{subject to } (x_{i+1} - x_i)^2 - \ell^2 = 0, \quad \forall i \in [1, N+1], \tag{6.15}$$

$$n_i^T\,(x_{i+1} - x_i) = 0, \quad \forall i \in [1, N], \tag{6.16}$$

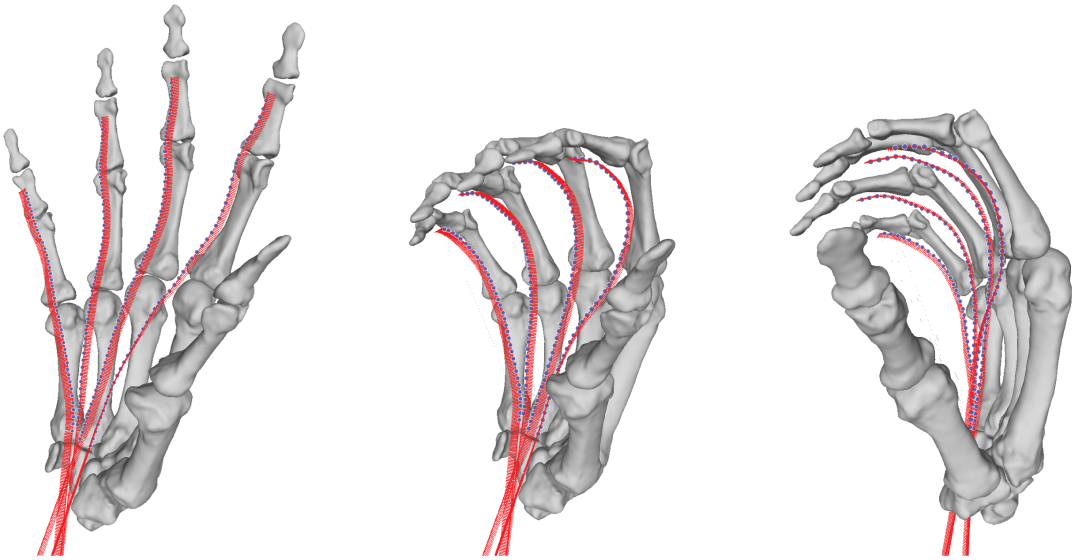
$$n_i^T n_i - 1 = 0, \quad \forall i \in [1, N], \tag{6.17}$$

$$0 \le t_i \le 1, \quad \forall i \in S, \tag{6.18}$$

$$x_1 = \hat{x}_1, \tag{6.19}$$

where S is the set of indices of segments with sliding constraints, $t_i \in [0,1]$ is the line segment barycentric coordinates giving the closest position to the given sliding constrained location $\bar{x}_i$, c_1, c_2 are parameters, ℓ is the length of the rigid segments, $\hat{x}_1$ is the fixed attachment location, and f is the constant pulling force. We define the sliding constraint as the sum of the squared distance from each sliding location to the closest point on the closest segment. For the twisting, we define it as the negative dot product between two neighboring normals. The bigger the dot product, the less twisting. Nevertheless, it is not sufficient to determine all n_i because the tendon could be globally rotated by the same rotation. Therefore, we use an input parameter $\bar{n}_1$ to regularize the normals. In the constraints, we guarantee that n_i is perpendicular to the segment i and is a unit vector. The

optimization problem is solved using the interior-point method. The method can easily find the optimal solution.



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

Figure 6.8: **Tendon simulation.** Here, we show the rods representing tendons during simulation. The red lines are the rigid segments and their normals. The blue dots are the sliding locations.

Given the simulation formulation, the only thing left is how we obtain the sliding locations, fixed attachment location, and the constant force. For the fixed attachment location $\hat{x}_1$, it is computed by rigidly transforming its initial position using the transformation of the closest bone. For the pulling force, we define its direction as the opposite of the primary direction of the last few segments in the rest configuration, because they are usually at the wrist region of the hand and move along their primary direction in the real world. For each sliding location, we compute it using skinning + pose-space correction using bone transformations. This gives the location based on the rotations of bones. After skinning and pose-space correction, we obtain new sliding locations in each frame. These sliding locations, however, may not be evenly distributed. We observed that some points may be very close to each other. To solve the problem, we filtered out points that are very close to each other, because these sliding constraints would introduce noises. Figure 6.8 presents the tendon sliding locations (blue) and the tendon simulation results. The process to obtain skinning weights and the pose-space corrections are explained in the next section (Section 6.2.2).

6.2.2 Tendon preprocessing

To perform the simulation, we need to know the tendon sliding locations. They are determined from MRI in multiple example poses. Therefore, we first extract the tendons from MRI.

6.2.2.1 Tendon extraction

Although we model tendons as rods, real tendons are volumetric objects. To extract tendons, we first treate 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 image using classic computer vision techniques. Next, we non-rigidly deform the template cylinder tube to match the MRI mesh. The center rod is deformed according to the surface mesh as we can consider it to be embedded into the tube volume. In this manner, we create our neutral tendon simulation rod and the surface mesh. For non-neutral poses, we only extract the surface of the tendon from MRI using classic computer vision techniques.

6.2.2.2 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 the 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 a high-quality result. Therefore, the registration of example poses is done in three stages. The first two stages create a good initial guess and the last stage performs the actual non-rigid registration.

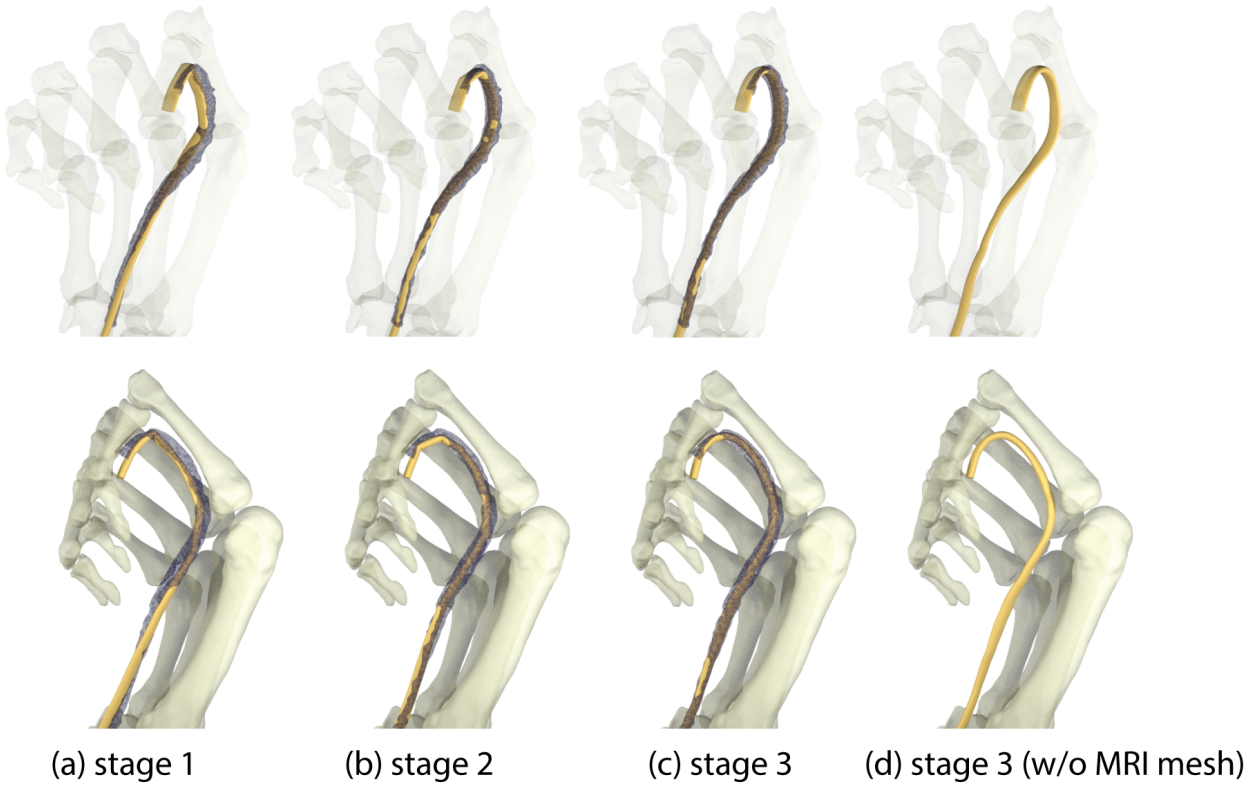


Figure 6.9: **Tendon stages.** In stage 1, we simulate the tendons based on the sliding constraints that are only rigidly transforming with the closest bones. This gives a relatively correct position of the tendon. However, it does not match the MRI shape (blue wireframe). To make it match as close as possible, we first create an initial guess for our non-rigid registration (stage 2). As shown in (2), the resulting mesh matches the MRI shape better. In stage 3, we do non-rigid registration to match the MRI shape.

Stage 1: Deform tendon to 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 considered as sliding locations. 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 Equation 6.19. The result is depicted in Figure 6.9(a).

Stage 2: Deform tendon to match the centerline of the extracted MRI mesh. After the first stage is completed, the position of the resulting tendon is close to the MRI mesh. 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 extract the centerline of the MRI mesh and sparsely sample a few points (10 points). Then, we use them as our sliding locations and run our simulation, beginning from the result of stage 1. The result of stage 2 is depicted in Figure 6.9(b).

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

$$\underset{x,n}{\arg\min} \quad \overbrace{\sum_i \left(\hat{n}_i^T \left(\left(\sum_{j \in M_i} w_{ij} T_j(x_j, x_{j+1}, n_j) \bar{p}_{ij} \right) - \hat{p}_i \right) \right)^2}^{\text{ICP constraints}} + E_{\text{pulling}}(x) + E_{\text{twisting}}(n) \tag{6.20}$$

$$\text{subject to} \quad (x_{i+1} - x_i)^2 - \ell^2 = 0, \quad \forall i \in [1, N+1], \tag{6.21}$$

$$n_i^T (x_{i+1} - x_i) = 0, \quad \forall i \in [1, N], \tag{6.22}$$

$$n_i^T n_i - 1 = 0, \quad \forall i \in [1, N], \tag{6.23}$$

$$x_1 = \hat{x}_1, \tag{6.24}$$

where $\bar{p}_{ij}$ is the unskinned rest position of surface vertex i to tendon segment j; $T_j(x_j, x_{j+1}, n_j)$ is the skinning transformation for segment j obtained using x_j, x_{j+1}, n_j; w_{ij} is the skinning weight of surface vertex i to the tendon segment j; M_i is the set of skinning tendon segments for surface vertex i; $\hat{p}_i$ is the ICP target position on the MRI mesh; and $\hat{n}_i$ is the ICP target normal. Again, we solve it using the interior-point method. The result of stage 3 is shown in Figure 6.9(c). As can be seen, it is smooth and closely matches the MRI mesh compared to previous stages.

6.2.2.3 Tendon sliding locations

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 sliding locations. For each sliding location i, 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 position to the example tendon rod in each example pose.

$$\arg\min_{w} \quad \sum_{j=1}^{N_p} \left(\sum_{k=1}^{N_s} w_k T_{jk} \bar{p}_{ik} - \hat{p}_j \right)^2, \tag{6.25}$$

$$\text{st.} \quad \sum_{k=1}^{N_s} w_k = 1, \tag{6.26}$$

$$0 \leq w_k \leq 1, \forall k \in [1, N_s], \tag{6.27}$$

where T_{jk} is the example transformation for bone k in pose j, $\bar{p}_{ik}$ is the unskinned rest position of sliding location i to bone k, and $\hat{p}_j$ is the closest position on the rod to the skinned point in example pose j. The example rod meshes are obtained from our non-rigid registration. To solve this optimization problem, we first initialize w to $(1,0)$, where the weight of the closest bone is 1 and the weight of the other bone is 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 the previous steps until the change of w between two consecutive iterations is close to zero. After the optimization problem is solved, we consider the vector between the skinning position and the closest point on the tendon rod as our pose-space correction in each example pose. Similarly to muscles, the pose-space vector consists of rotations of bones that drive the tendon. The representation of a rotation is the same as the muscle (Section 6.1.1).

6.3 Hand fascia

The gaps and valleys around bones and muscles cause problems when we simulate the fat layer directly on top of the bones and muscles. Therefore, we create fascia layers wrapping muscles and bones, respectively. Each single fascia is a single surface mesh. For bone fascia, we merge our bone meshes with the convex hull meshes generated in Section 4.5. This gives us the surface mesh of the bone fascia. Then, we shrink the rest material coordinates of the bone fascia mesh

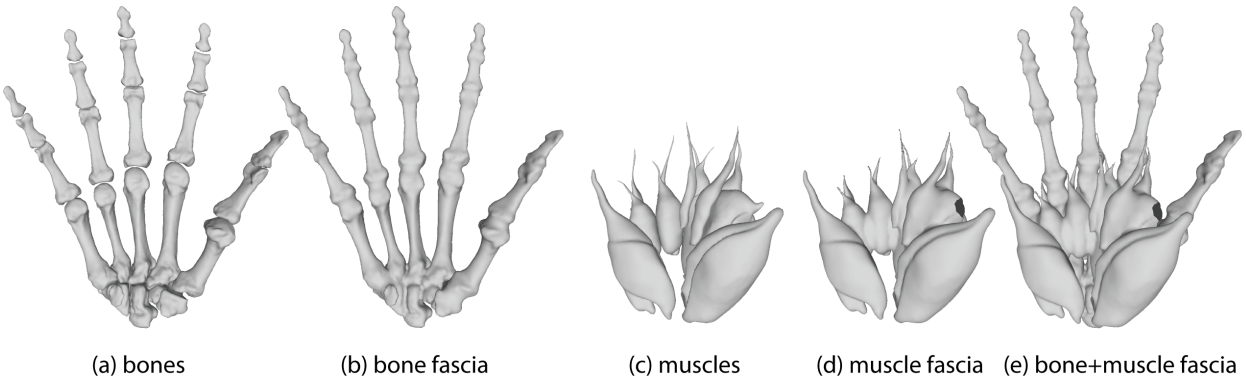


Figure 6.10: **Fascia meshes.** (a) Bone meshes, (b) corresponding bone fascia mesh, (c) muscle meshes, and (d) muscle fascia mesh.

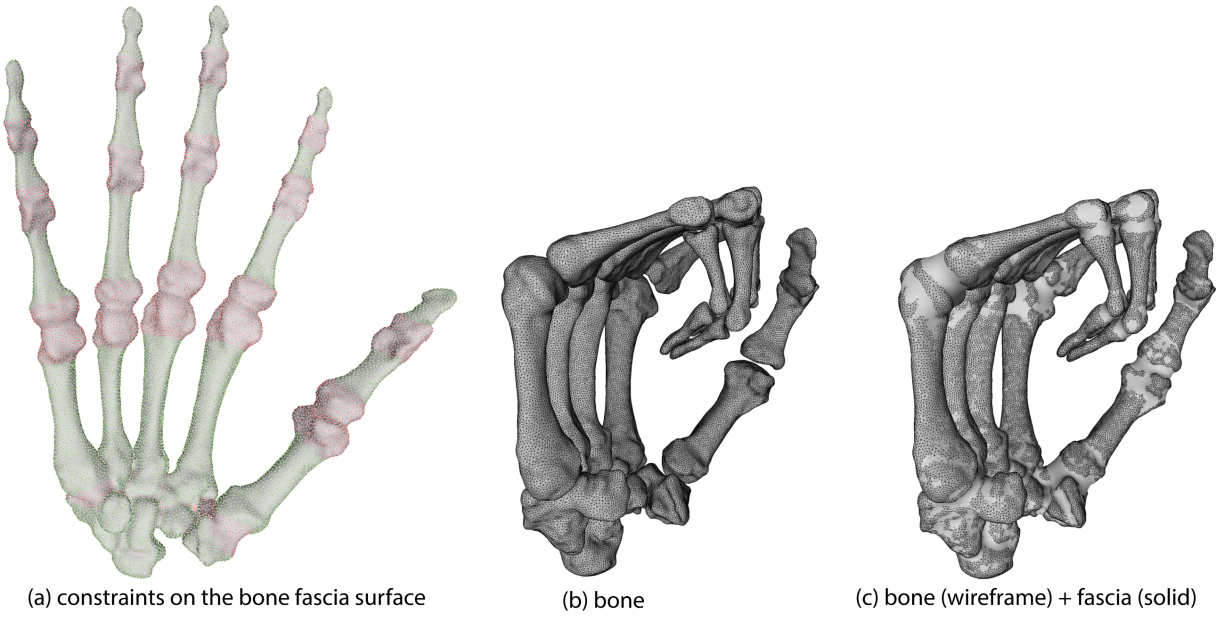


Figure 6.11: **Bone fascia.** (a) The constraints applied to bone fascia simulation. Vertices in green are attached to the closest bones using fixed constraints. Vertices in red are in contact with the bone meshes. (b, c) Bone and bone fascia meshes in the fist pose.

so it tightly wraps the bone geometry, which is already in the rest state. In our experiment, we shrink them by 0.35x. Similar to bones, we merge all muscle surface meshes together and generate the muscle fascia mesh. For muscles, we shrink the rest material coordinates by 0.6x. In each simulation timestep, we first rigidly transform the bones and then simulate the bone fascia using a

cloth solver [129]. After that, we perform muscle simulation and then simulate the muscle fascia using the

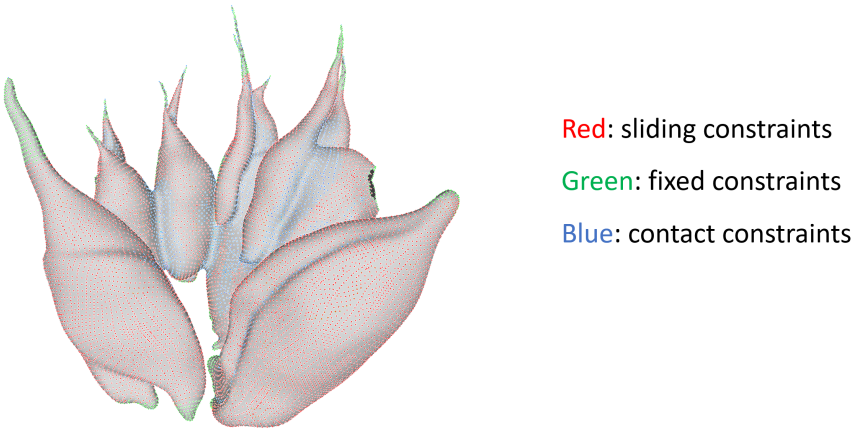


Figure 6.12: **Muscle fascia constraints.** Here, we show the constraints applied during muscle fascia simulation. Vertices in green are attached to the muscles using fixed constraints. Vertices in red are sliding against the muscle surface meshes.

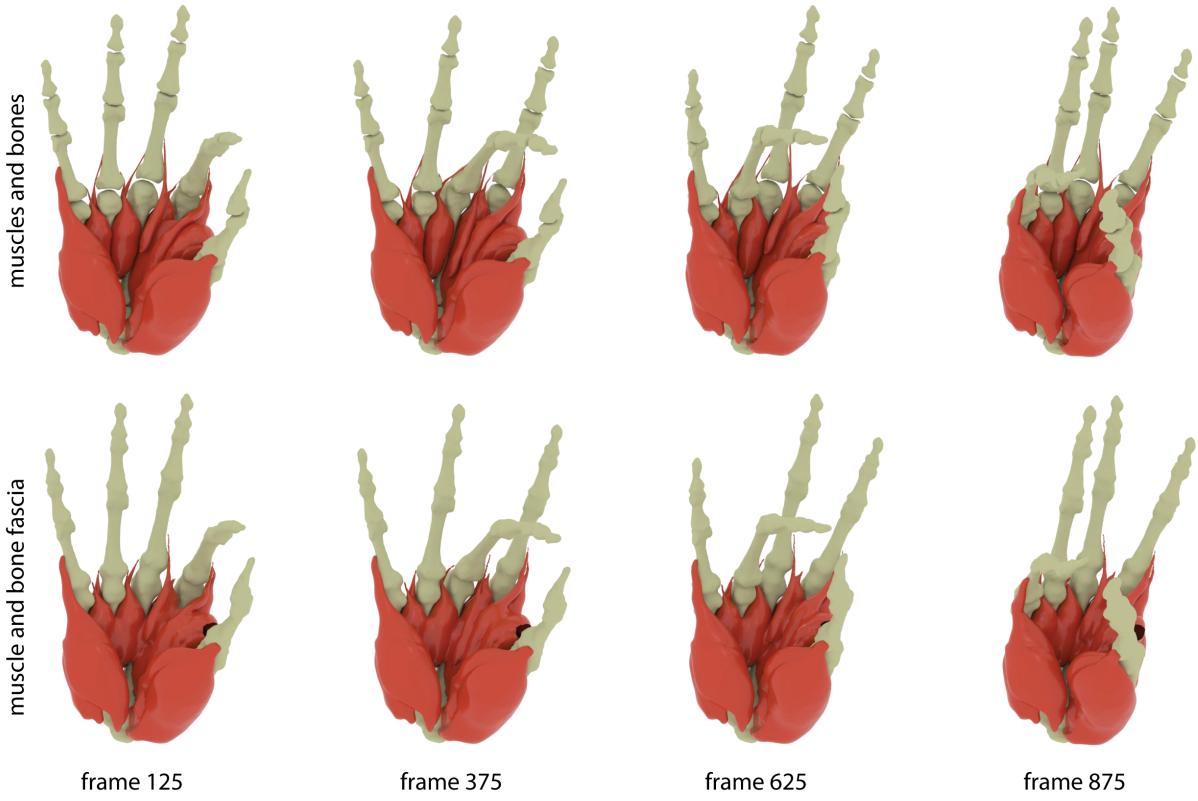


Figure 6.13: **Fascia animation.** Top: a few frames of an animation sequence of bones and muscles. Bottom: the fascia simulation results.

For bone fascia, the vertices in the area around bone bodies of the bones are attached to the bone surfaces and move with the bone rigidly. On the other hand, the fascia vertices around joints are in contact with the bone surfaces. They are selected manually, the same way as for creating the convex hull in Section 4.5. We select the heads of the bones, as shown in Figure 6.11. The bone fascia surface nicely shrinks/expands as the bones move (Figure 6.11), partially due to the shrinking of the rest material coordinates. For example, when the fascia mesh around the joints shrinks because of the bone movement, the fascia mesh does not fold. Instead, it shrinks nicely and keeps the tension of the surface. For muscle fascia, the vertices are sliding and contacting against the muscle surface except in the areas of attachment vertices (Figure 6.12). The animation of the bone and muscle fascia corresponding to the deformations of the bones and muscles is shown in Figure 6.13.

6.4 Hand fat

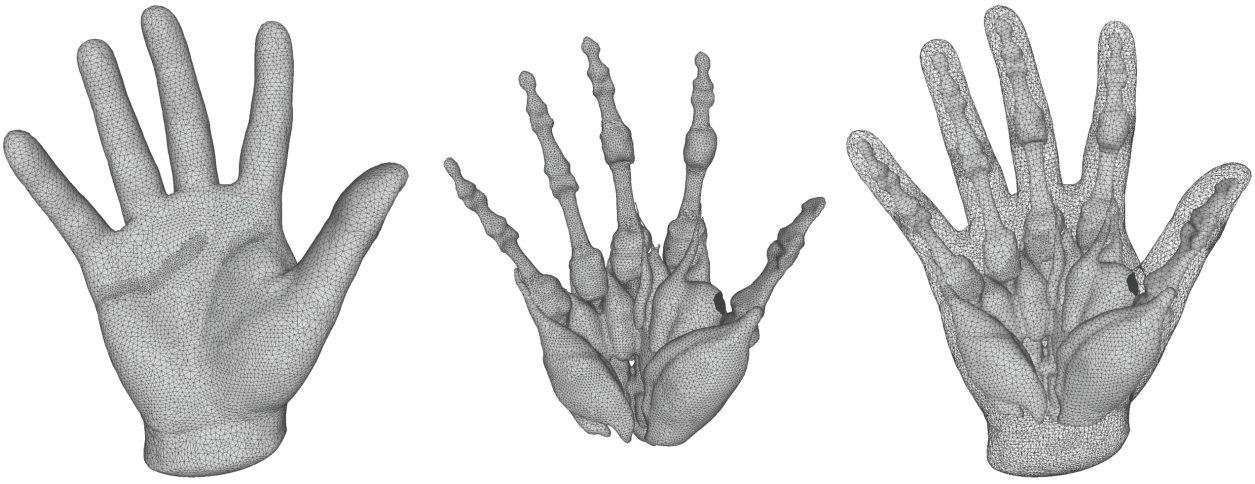


Figure 6.14: **The exterior and interior surface of the fat tissue.** (a) The exterior surface of the fat tissue is the skin. (b) The interior surface of the fat tissue is composed of the bone fascia, muscle fascia, and joint ligaments.

Because fat is a passive tissue, our fat simulation is similar to the method proposed in Chapter 4 but with a few changes. Our fat tissue has two surfaces, the skin (Figure 6.14(a)), and the interior surface (Figure 6.14(b)) that contacts with the bones, muscles, and joint ligaments. The skin is obtained using the same method as that mentioned in Chapter 4. The skin and the interior surface of the fat are embedded into a tetrahedral mesh. We can significantly reduce the number of tetrahedra because the tetrahedral mesh does not have to conform the surface. We first create a single tetrahedral mesh that represents the skin of the hand. Then, we perform inside/outside test for all tetrahedra against interior surfaces. We keep tetrahedra that are intersected with muscle fascia, bone fascia, or joint ligaments. We assign a single material property to the fat. 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 weight the Young's modulus and mass density by the volume occupation.

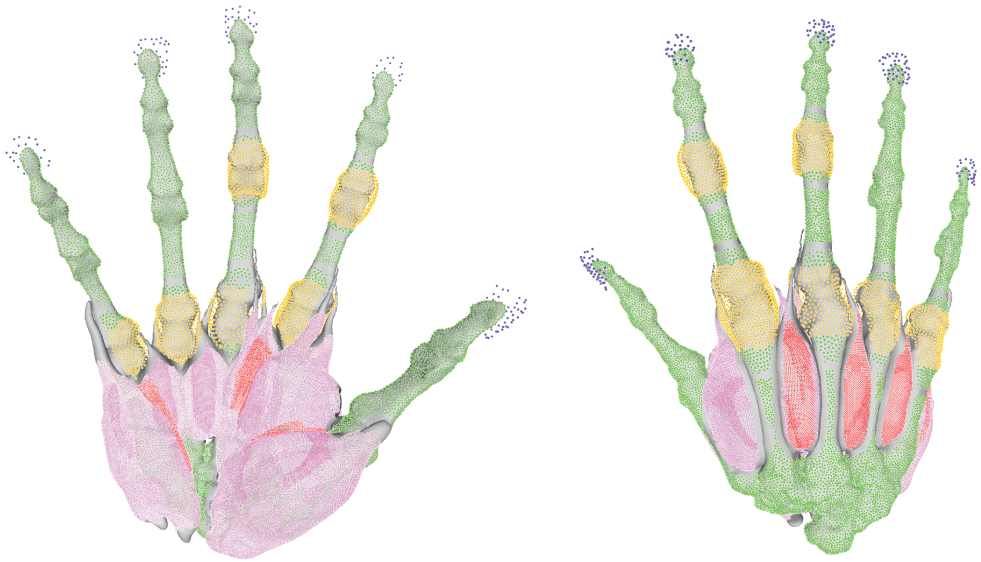


Figure 6.15: **The constraints for simulating the fat.** The fat is attached to bone fascia, muscle fascia, ligaments, and tendons. Green dots are the fixed constraints to the bone fascia. Yellow dots are the contact constraints to the ligaments. Pink dots are the sliding constraints against the muscle fascia, while the red dots are contact constraints against the muscle fascia. In addition, the purple dots are the constraints to mimic the rigidity of the nails.

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. There are three types of constraints in our model: fixed constraints, sliding constraints, and contact constraints, as shown in Figure 6.15. A fixed constraint anchors a point to a target location. A sliding constraint anchors a point to a target plane. A contact constraint penalizes penetration of a point beneath a target plane. Inspired by biomechanics, for bone fascia, we create fixed constraints. As the bone fascia already slides against the bones, we do not need to apply another layer of sliding constraints. For muscles, we found that the sliding constraints + contact constraints yield the best results for all vertices, except the attachment vertices to the bones. If fixed constraints were used, it would result in a bumpy skin shape. We manually determined the weights for sliding constraints and contact constraints. Even though such a process is manual, it is simple and usually takes less than 10 minutes to do it once. Moreover, the weights are the same for all example poses, so we only need to do it once. The procedure for the joint ligaments is similar to the muscles. We apply sliding constraints to all vertices except the attachment vertices to the bones. Finally, as mentioned in Chapter 4, we also apply fixed constraints to the nails of the fingers, to keep them rigid. In addition to the constraints on the fat surface, we also apply self-collision handling on the skin, as described in Chapter 4.

6.5 Results

The MRI data is obtained using the approach described in Chapter 3. As shown in Chapter 4, we acquired 12 example poses. Among them, we selected six example poses for our muscles, tendons, and joint ligaments, as shown in Figure 6.16. We selected them to maximize the exertion of these tissues as much as possible. We first compared our simulation results with [134] quantitatively in all example poses. Then, we compared our simulation results visually with [134]. Both comparisons demonstrated that our hand model is superior to the previous work and matches the ground truth better.

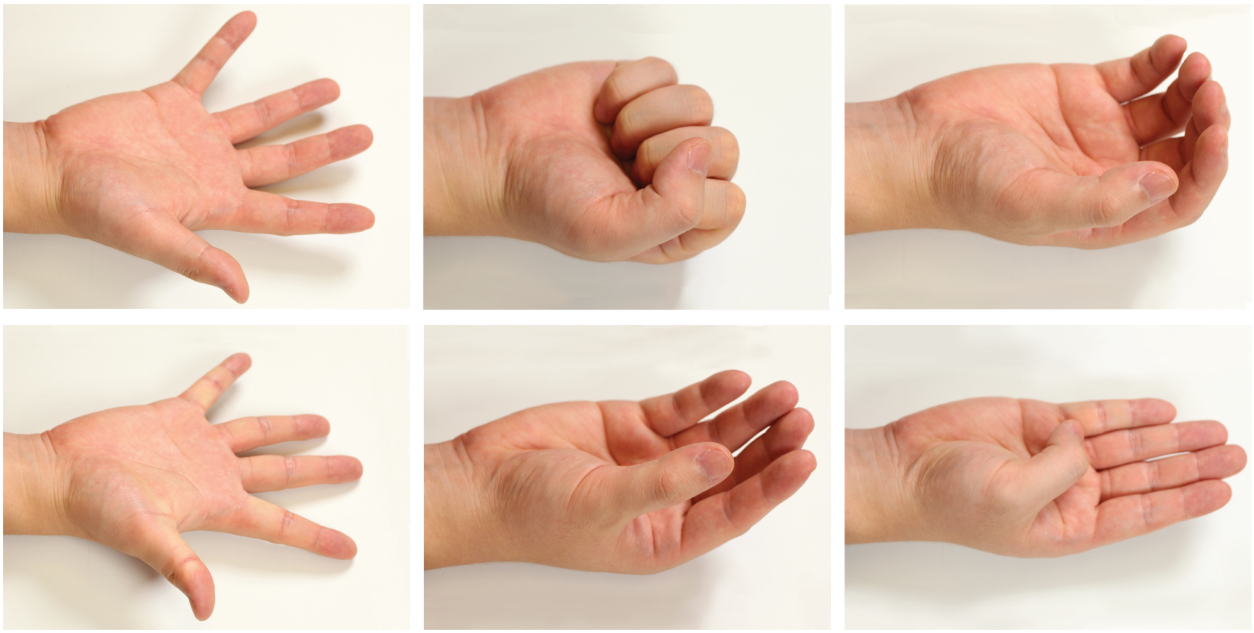


Figure 6.16: **Hand poses used for joint ligaments, muscles, and tendons.**

We deformed our entire simulation model to each target example pose. For each example pose, we used the ground truth transformation for each bone. This guarantees that the bones are at the correct locations. We first began with the muscles. The extracted meshes of the muscles are presented in Figure 6.17. We then pre-processed these muscles using our pipeline and build our muscle simulation model. Therefore, we simulated muscles to reach the example poses and compare the results with the ground truth meshes extracted from MRI. As shown in Figure 6.18, we selected a few example poses that we consider are the most extreme poses among all six example poses. It is evident that the simulation results closely overlap with the ground truth meshes. Still, tiny errors remain. These errors are expected as our model contains inter-muscle sliding forces and contact forces. We did not consider these forces when creating the example muscle surfaces and corresponding example plastic strains. Nonetheless, our pre-processing pipeline guarantees that these forces are minimized at example poses, because contact resolving cleans up the example surface meshes. In addition, we performed tendon comparisons at the same example poses as those with the muscles. As depicted in Figure 6.19, the simulation results closely match with the ground

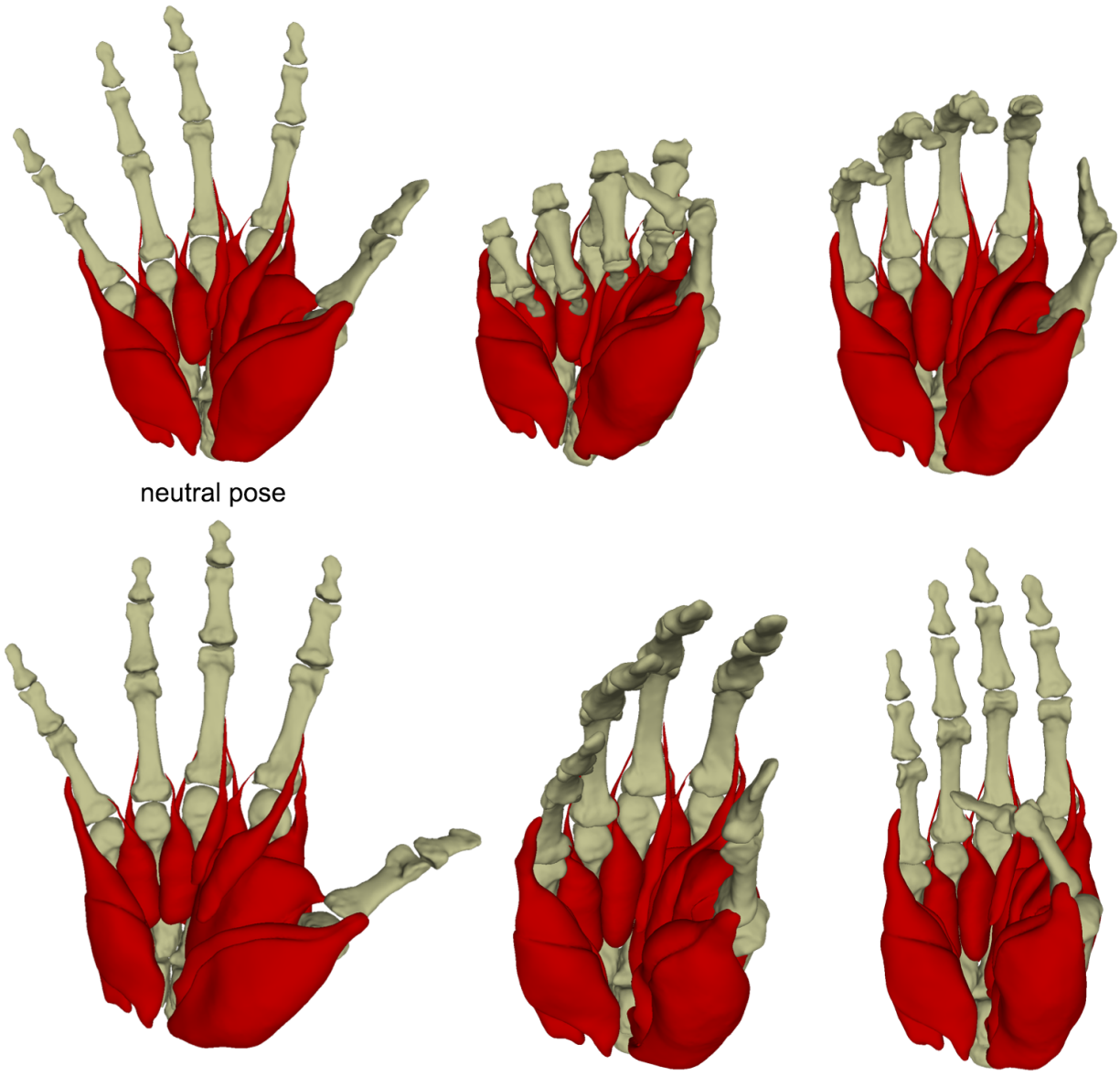


Figure 6.17: **Extracted muscle meshes in example poses.**

truth meshes. These errors are expected as we only control the sliding locations. We did not attempt to match the orientations of the tendon rod vertices (i.e., normals) in the example poses.

After the comparing the interior structure, we would like to measure the accuracy improvement between modeling each organ as a separate object and simulating all soft tissues using a single tet mesh. For each vertex on the ground truth surface mesh matching matches the MRI in each example pose, we compute the closest distance to the simulation mesh. Table 6.5 presents

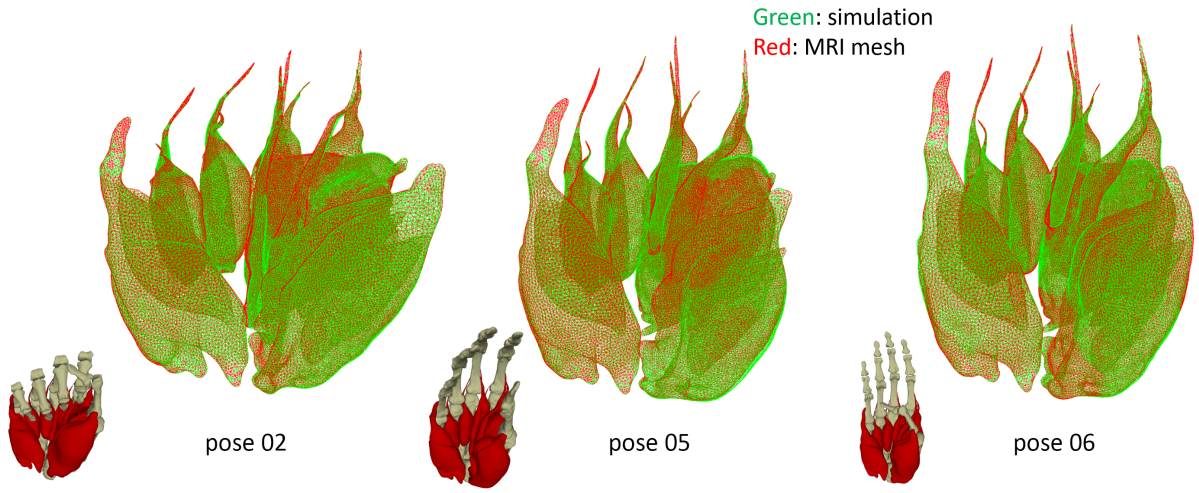


Figure 6.18: **Comparisons between muscle simulation results and the ground truth meshes in example poses.** We simulated hand muscles using our model to reach a few example poses that we believe are the most extreme poses among all six example poses. Simulation results (green wireframe) closely overlap with the ground truth meshes (red wireframe).

Figure 6.19: **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).

average distance, median distance, and max distance for each example pose. We can see that the separate-mesh reduces the error by 24% for average distance, 21% for median distance, and 24% for maximal distance. For all 12 poses, this is better in average, median and max distance in majority of the cases. There are only 4 cases (marked in bold) out of 36 where our method produces slightly inferior numbers.

Table 6.1: **Comparisons between simulated and ground truth skins.** Model m_1 simulates each organ separately (as proposed in this Chapter), whereas model m_2 simulates all soft tissues using a single mesh (as proposed in this Chapter 4). Column "%" denotes the ratio between the value in m_1 and the value in m_2. The values in bold face are cases where our model is slightly worse than m_2. The first five rows are the example poses used for our simulation, whereas the last six rows are the unseen poses. All distances are represented in millimeter units.

pose	average			median			max		
	m_1	m_2	%	m_1	m_2	%	m_1	m_2	%
2	0.94	1.35	69.63%	0.76	1.22	62.30%	4.81	4.70	**102.34%**
3	0.72	1.19	60.50%	0.58	0.88	65.91%	3.89	6.34	61.36%
4	0.93	1.31	70.99%	0.77	0.86	89.53%	3.34	7.09	47.11%
5	1.02	1.21	84.30%	0.83	1.00	83.00%	4.60	5.61	82.00%
12	0.72	1.15	62.61%	0.59	0.90	65.56%	4.89	7.86	62.21%
6	1.01	1.05	96.19%	0.86	0.78	**110.26%**	4.50	5.46	82.42
7	0.63	0.84	75.00%	0.50	0.66	75.76%	2.85	2.70	**105.56%**
8	0.89	1.02	87.25%	0.69	0.8	86.25%	5.20	4.67	**111.35%**
9	0.97	1.08	89.81%	0.85	0.92	92.39%	4.15	4.80	86.46%
10	1.09	1.43	75.22%	0.80	1.08	74.07%	4.89	6.30	77.62%
11	0.88	1.24	70.97%	0.62	0.87	71.26%	5.63	7.69	73.21%
Average	0.89	1.17	76.22%	0.71	0.91	78.74%	4.38	5.75	76.23%

We employed the same sequence as used in [134] and compared the final rendering results with it. We observed a clear improvement over the palm area, as shown in Figures 6.20, 6.21. Compared to [134], the muscles are modeled using plastic strains in our model, and are controlled by the bone transformations. In other words, we "activate" the muscles based on the hand pose, which improves the simulation results.

Using our model, the joint regions are also improved, as shown in Figure 6.21. This is because we model the joint ligaments using plastic strains controlled by the bone transformations, where previously, these tissues were not modeled.

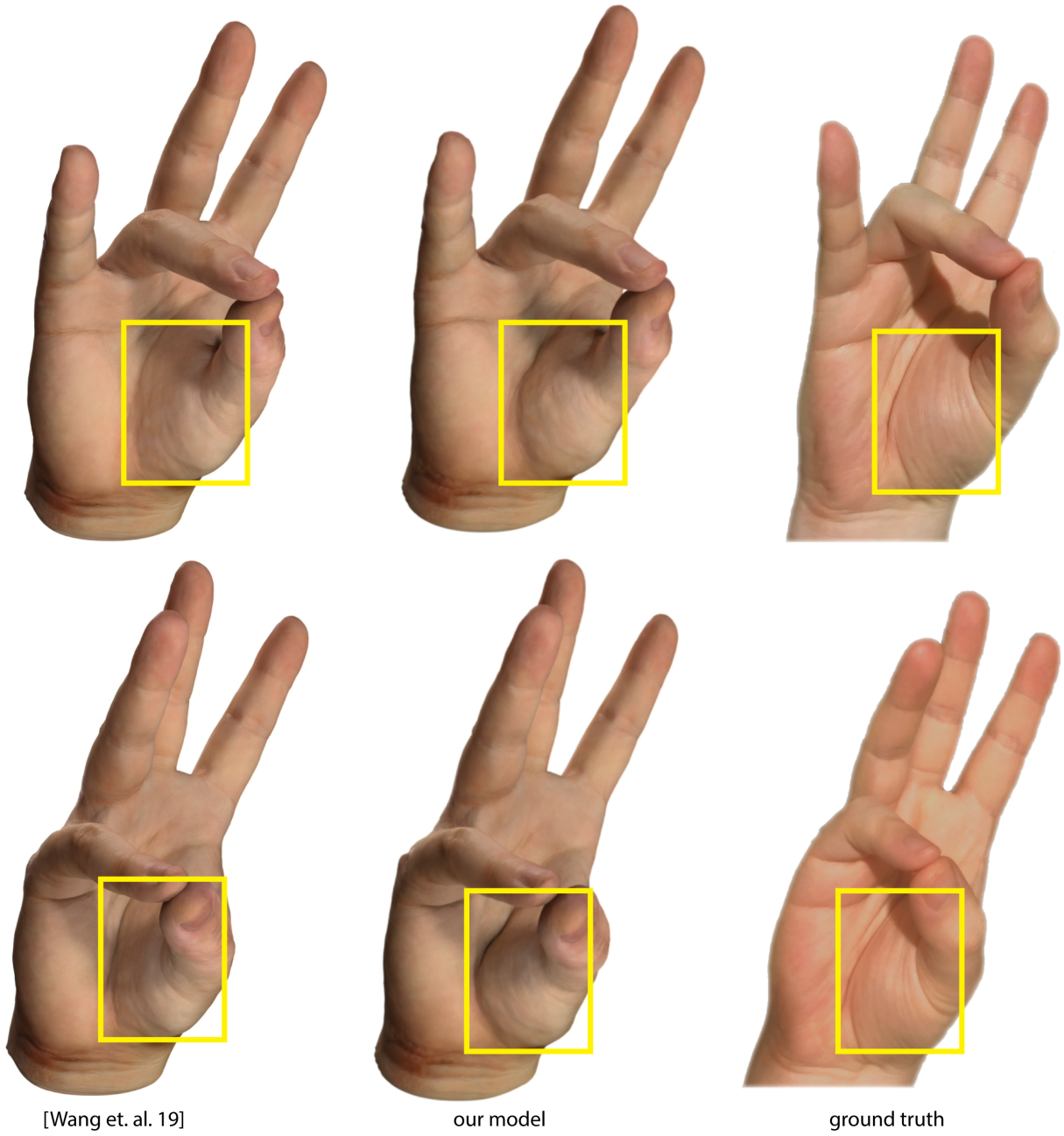


Figure 6.20: **Visual comparisons to Wang et al. [134].** The palm area bulges more under our simulation model, which is closer to the behavior in the real world.

We also performed a few experiments to see how well the muscle shapes are reproduced in both example poses and non-example poses. Figure 6.22 indicates that our simulation result closely matches the MRI data in example poses. For the non-example poses, we can see that our model produces reasonable results (Figure 6.23).

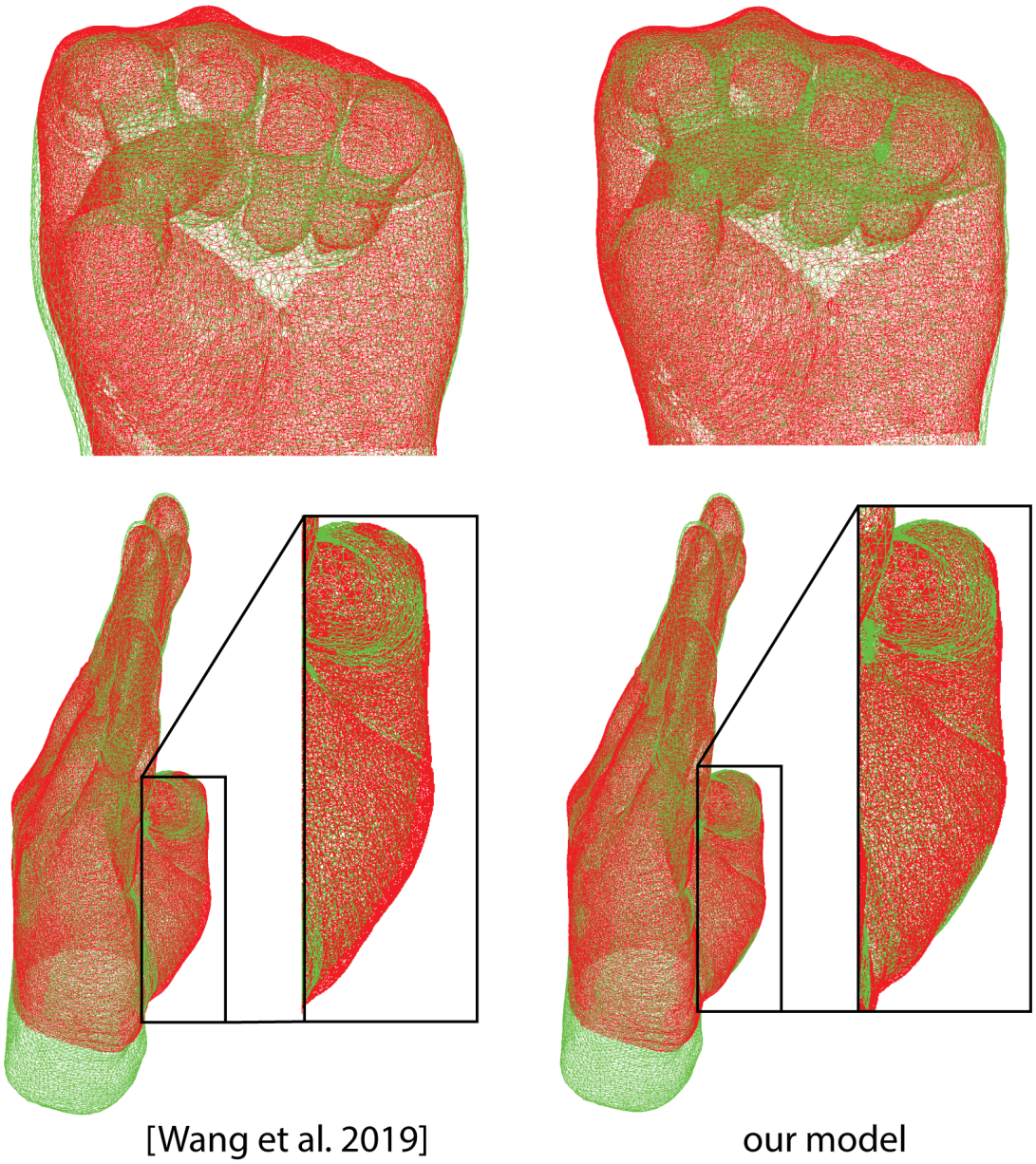


Figure 6.21: **Visual comparisons of joint regions to Wang et al. [134] in example poses.** The ground truth mesh is depicted in red wireframe, whereas the simulation results are shown in green wireframe. The palm and joint areas bulge more correctly in our simulation model compared to the single-layer soft tissue model.

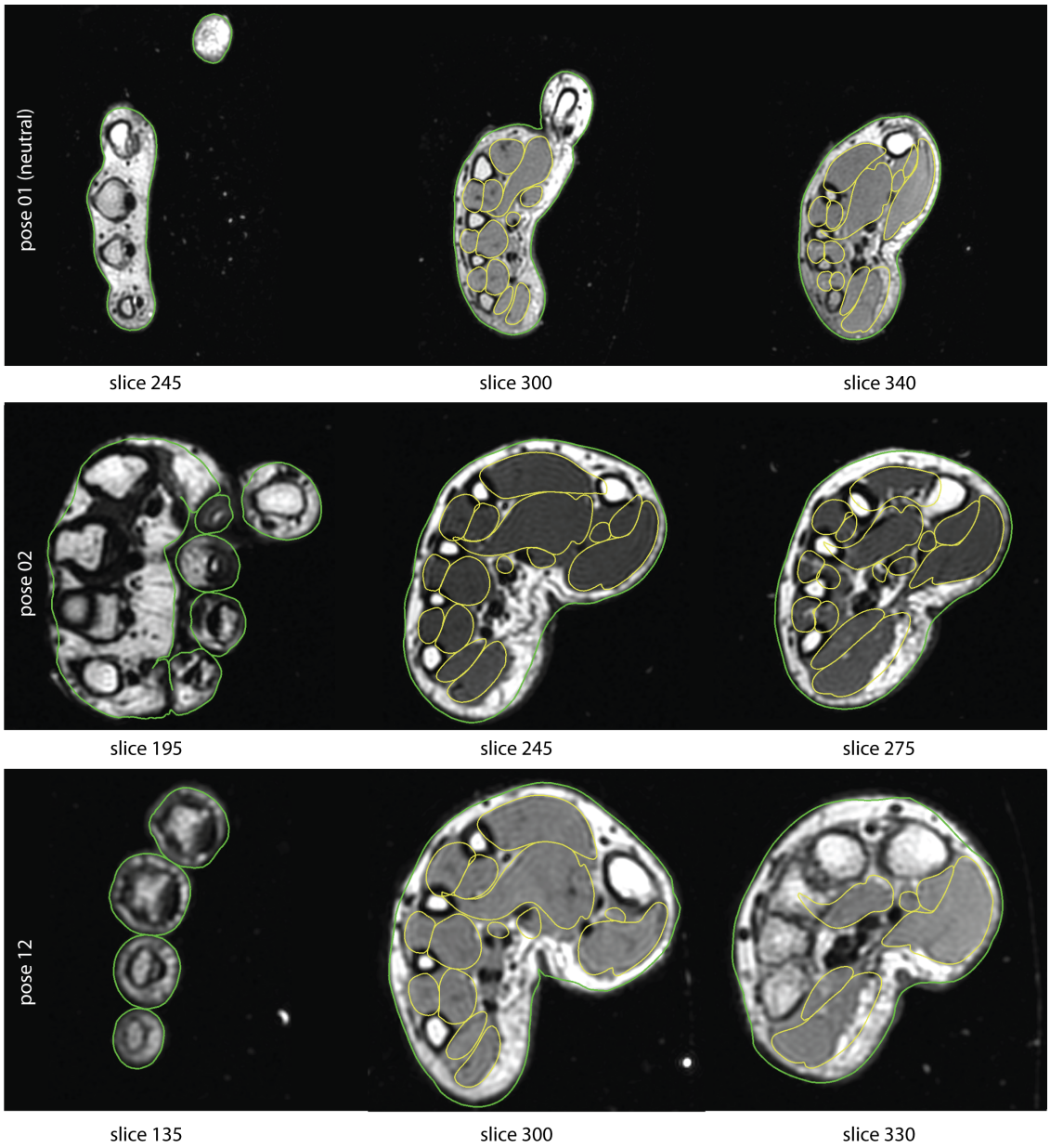


Figure 6.22: **Visual comparisons to MRI slices in example poses**. We compare our simulated skin and muscle surfaces with the MRI images in example poses 1, 2 and 12. Pose 1 is the neutral pose. Pose 2 (fist) and 12 (thumb moving to the most opposite location) are extreme example poses. We intersect our surface meshes with the MRI slices. The intersections between skin and the slices are presented in green. Further, the intersections between muscles and the slices are depicted in yellow. We observe a very close match to the MRI data for both skin and muscles.

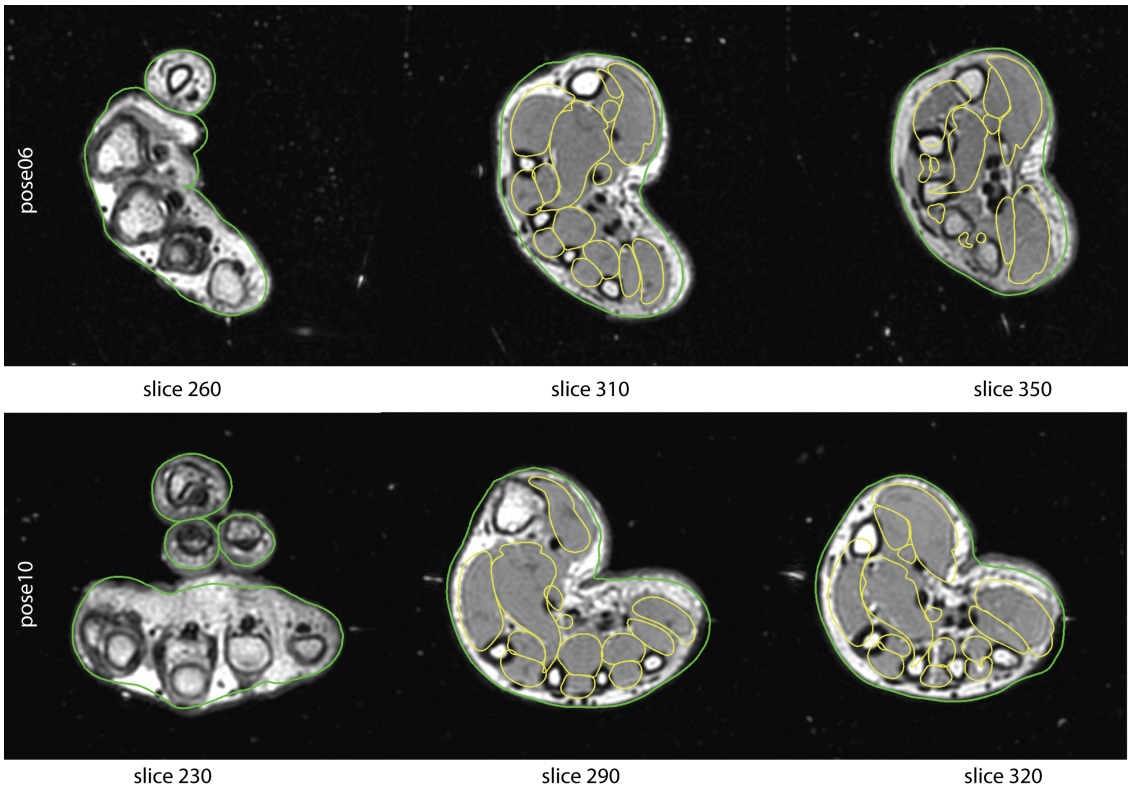


Figure 6.23: **Visual comparisons to MRI slices in non-example poses**. We compare our simulated skin and muscle surfaces with the MRI images in non-example poses (poses 6 and 10). The intersections between skin and the slices are shown in green, and the intersections between muscles and the slices are depicted in yellow. We can observe a reasonably close match to the MRI data for both skin and the muscles.

Chapter 7

Conclusion and Future Work

This thesis proposed a pipeline for personalized human hand modeling and simulation. The pipeline begins from the acquisition of personalized MRIs and surface scans in multiple example poses, and then a realistic and animation-ready personalized simulation rig is built. We used a novel molding procedure to stabilize hands during MRI scanning. We not only captured MRI in multiple poses but also acquired the high-resolution skin geometry that matches the MRI simultaneously. Then, we demonstrated how to acquire highly accurate human skeleton geometry in multiple poses using MRI. We registered all poses to the same bones mesh connectivity, and built a skeleton mesh kinematic model ("skeleton rig") to interpolate or extrapolate the acquired pose to the entire range of motion of the hand. We demonstrated that our skeleton rig can be used to drive soft-tissue FEM simulation to produce anatomically plausible organic hand shapes that qualitatively match photographs.

In addition, we improved our model by capturing the deformation of interior structures including muscles, tendons, joint ligaments, and fat independently. To extract these tissues from the MRI, we proposed a novel shape deformation method that can model objects undergoing large spatially varying strains, such as muscles. Our method works by computing a plastic deformation gradient at each tet, such that the mesh deformed by these plastic deformation gradients matches the provided sparse landmarks and closest-point constraints. We applied our method to the extraction of shapes of organs from medical images. Further, our method has been designed to extract as much information as possible from MRI images, despite the soft boundaries among the different organs.

Our method not only applies to hand muscles, but also to other organs such as liver, hip bone and hip muscle. Finally, we gave a complete system for hand animation by modeling bones, muscles, tendons, joint ligaments, and fat separately. We designed a pipeline to pre-process MRI and optical scans into a simulation-ready hand rig. We demonstrated that our model is accurate. Our model not only matches the ground truth skin, with average errors less than 1mm in all example poses, but it also produces matching interior structures in all example poses. Furthermore, our model produces realistic hand motions. The simulation results qualitatively match the hand in the real world.

We performed 12 scans for each subject, which is sufficient to demonstrate stable and precise common hand motions, including the opposition of the thumb to all the other four fingers. Accuracy would be improved with more scans. An interesting research question is how to automatically select the next best pose to scan, in order to maximize coverage of the hand's range of motion. Some complex hand poses are challenging for our technique, for example, all fingertips touching at a single point. This is because we cut our molds into two pieces manually using a knife. It would be interesting to explore how to cut the molds into three or more pieces, to improve the ergonomics of the insertion of the hand into the mold. Our accurate hand model can potentially benefit virtual hands in games / film / VR, robotic hands, grasping, and medical education, such as visualizations of internal hand anatomy motion. The understanding of the motion of internal hand anatomy as proposed here takes us one step closer to life-like robots with hands that mimic biological hands.

Modeling the human hand accurately is a very broad area and there are still many aspects in our model that can be improved and explored in the future. First of all, we only capture 12 poses due to our limited resources. As several components of our model depend on captured data, the more data is captured, the more accurate the results would be. Moreover, capturing more hand poses implies that the ranges of motions of the hand are explored better. For each example pose, our capturing procedure is more complex compared to optically-based scanning methods, such as motion capture and 3D reconstruction, but our model requires much fewer number of example poses compared to them. Still, automating our pipeline would improve the efficiency of capturing data. In addition, improving the MRI resolution would also help the fitting procedure, as there

are many small structures in the hand. For example, we observed that the thickness of one of the extensor tendons is occasionally represented by only 1-2 voxels. The distal phalanges of our female subject are also represented by only a few voxels.

Further, we could also explore more types of joints. In our model, we only consider each hand joint as either a 1D joint or a 2D joint: either a hinge joint or a condyloid joint. Other types of joints such as saddle joints or ball joints could also be considered and these may lower the fitting errors in our skeleton model. Our bone rig is a kinematic model: the transformations are controlled by joint angles. Therefore, our model is artist-friendly and easy-to-use. Nonetheless, it is difficult to validate the input joint angles in our model. When we captured hand motions from LeapMotion or other motion capture systems like Google Mediapipe, it was difficult to know if the captured motion is valid. An invalid input sequence of joint angles can cause contact of the skin, or unrealistic deformations of the hand. Unless we perform our full-hand simulation, it is not easy to validate the input motion. Moreover, if repair is required, it needs to be done manually. This problem could be addressed by capturing many hand poses to build a comprehensive and valid pose space.

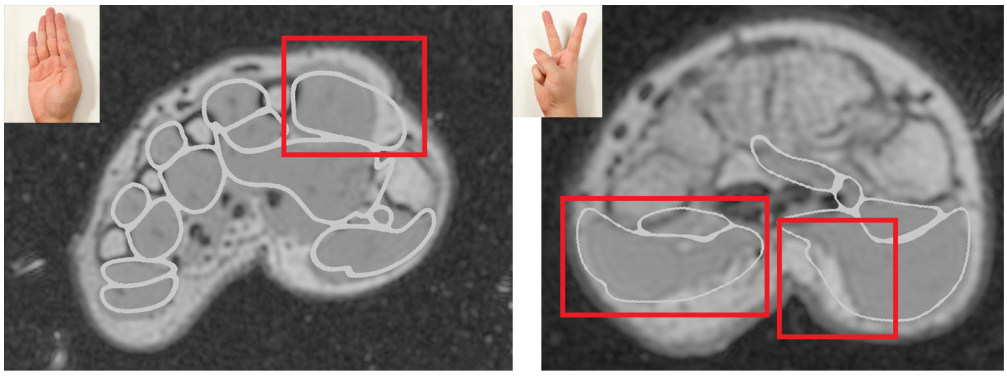


Figure 7.1: **The mismatches between simulation results and the MRI.** The silhouettes of muscles are marked in white. Mismatches are highlighted by red rectangles. We show two typical types of mismatches that are not handled by our model.

To use our muscle, ligament, and tendon simulation models, we must preprocess the MRI and optical scans at multiple example poses. We need to separately extract every tissue in the scans of multiple example poses. Even with our proposed shape modeling method, this is still a

considerable amount of manual work. Reducing this manual work would be beneficial and could be left for future work. Our model is based on the example pose data. Accordingly, if there is an unseen pose that is close to the example poses, the prediction of the muscle shape is reasonable, as demonstrated by experiments depicted in Figure 6.23. On the other hand, if the unseen pose is far from the examples, it may not be predicted accurately. For example, we found that the shape of first dorsal interosseus muscle is not well matched with the MRI in our example pose 6, as shown in Figure 7.1(left). Using more example poses would alleviate this issue.

Another cause of this mismatch is that we did not model the interactions between fat tissue, fascia, and the muscles. In the real world, fat tissue and fascia are pushing muscles inwards so that muscles are aggregated together. Without modeling these forces, the muscles may not be positioned correctly, as depicted in Figure 7.1(right). The fat layer is also affected by the same problem, although we rarely find such artefacts. It is difficult to introduce this coupling into our system due to the large amount of computation. We would have to perform our optimization and simulation with all tissues—including all muscles, fat, and fascia—coupled together. This does not only increase the time cost tremendously, but it also brings a lot of difficulties in identifying the optimal solution during the optimization, such as the optimization that extracts the muscle shapes.

We did not model the muscle activation forces. Although our muscles match the ground truth shape at example poses, they do not produce correct forces. To address this, we could fit the scans to a muscle model that models muscle activation, such as Hill's model [144].

Finally, although we modeled key structures underneath the hand skin, there remain additional tissues that could be modeled in the future. It can be seen in Table 6.5 that, while the average and median errors are small, the maximal errors are not negligible. One of the possible reasons for this is that we did not model the extensor tendons on the back side of the hand. This could cause a mismatch on the back side of the hand, as shown in Figure 6.21. Thus, modeling all important tissues of the hand is an important avenue for future work.

Acknowledgments

This research was sponsored in part by NSF (IIS-1911224), USC Annenberg Fellowship to Bohan Wang, Bosch Research and Adobe Research.

Conflict of interest statement

The hip bone and hip muscle template meshes were obtained at Ziva Dynamics. Jernej Barbič is a shareholder, CTO and board member of Ziva Dynamics. Ziva Dynamics was not involved in this research. Nothing in this thesis is to be understood as endorsement of Ziva Dynamics or its products.

Bibliography

[1] Jascha Achenbach, Eduard Zell, and Mario Botsch. Accurate Face Reconstruction through Anisotropic Fitting and Eye Correction. In David Bommes, Tobias Ritschel, and Thomas Schultz, editors, *Vision, Modeling & Visualization*, 2015.

[2] Agisoft. Photoscan, http://www.agisoft.com, 2018.

[3] M. Alexa, A. Angelidis, M.-P. Cani, S. Frisken, K. Singh, S. Schkolne, and D. Zorin. Interactive shape modeling. In *ACM SIGGRAPH 2006 Courses*, page 93, 2006.

[4] AljaSafe. SmoothOn Inc., 2018. www.smooth-on.com.

[5] Brett Allen, Brian Curless, and Zoran Popović. The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Trans. Graph.*, 22(3):587–594, 2003.

[6] Brian Amberg, Sami Romdhani, and Thomas Vetter. Optimal Step Nonrigid ICP Algorithms for Surface Registration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[7] Amira. Amira for Life Sciences, 2018. https://www.fei.com/software/amira-3d-for-life-sciences/.

[8] Artec3D. Spider Scanner, http://www.artec3d.com, 2018.

[9] Artelys. Knitro, 2019. https://www.artelys.com/solvers/knitro/.

[10] Noelle M. Austin. *Chapter 9: The Wrist and Hand Complex". In Levangie, Pamela K.; Norkin, Cynthia C. Joint Structure and Function: A Comprehensive Analysis*. F. A. Davis Company, 4th edition, 2005.

[11] J. Barbič, M. da Silva, and J. Popović. Deformable object animation using reduced optimal control. *ACM Trans. on Graphics*, 28(3), 2009.

[12] J. Barbič and D. L. James. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. on Graphics*, 24(3):982–990, 2005.

[13] J. Barbič and D. L. James. Six-dof haptic rendering of contact between geometrically complex reduced deformable models. *IEEE Trans. on Haptics*, 1(1):39–52, 2008.

[14] J. Barbič and Y. Zhao. Real-time large-deformation substructuring. *ACM Trans. on Graphics (SIGGRAPH 2011)*, 30(4):91:1–91:7, 2011.

[15] A. W. Bargteil, C. Wojtan, J. K. Hodgins, and G. Turk. A finite element method for animating large viscoplastic flow. In *ACM Transactions on Graphics (SIGGRAPH 2007)*, volume 26, page 16, 2007.

[16] Pierre-Yves Baudin, Noura Azzabou, Pierre G Carlier, and Nikos Paragios. Automatic skeletal muscle segmentation through random walks and graph-based seed placement. In *IEEE Int. Symp. on Biomedical Imaging (ISBI)*, pages 1036–1039, 2012.

[17] Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. TRACKS: Toward directable thin shells. *ACM Trans. on Graphics (SIGGRAPH 2007)*, 26(3):50:1–50:10, 2007.

[18] Paul J. Besl and N.D. McKay. A method for registration of 3-d shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[19] Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67(5):405 – 451, 2005.

[20] M. Botsch and O. Sorkine. On linear variational surface deformation methods. *IEEE Trans. on Vis. and Computer Graphics*, 14(1):213–230, 2008.

[21] Mario Botsch and Leif Kobbelt. An intuitive framework for real-time freeform modeling. volume 23, pages 630–634, 2004.

[22] Mario Botsch, Mark Pauly, Markus Gross, and Leif Kobbelt. PriMo: Coupled Prisms for Intuitive Surface Modeling. In *Eurographics Symp. on Geometry Processing*, pages 11–20, 2006.

[23] Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. Shape-up: Shaping discrete geometry with projections. *Comput. Graph. Forum*, 31(5):1657–1667, 2012.

[24] Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph.*, 33(4):154:1–154:11, July 2014.

[25] Eddy Boxerman. *Speeding up cloth simulation*. PhD thesis, University of British Columbia, 2003.

[26] C. Erolin. Hand Anatomy. University of Dundee, Centre for Anatomy and Human Identification, 2019. https://sketchfab.com/anatomy_dundee/collections/hand-anatomy.

[27] S. Capell, M. Burkhart, B. Curless, T. Duchamp, and Z. Popović. Physically based rigging for deformable characters. In *Symp. on Computer Animation (SCA)*, pages 301–310, 2005.

[28] I. Chao, U. Pinkall, P. Sanan, and P. Schröder. A Simple Geometric Model for Elastic Deformations. *ACM Transactions on Graphics*, 29(3):38:1–38:6, 2010.

[29] W. Chen, F. Zhu, J. Zhao, S. Li, and G. Wang. Peridynamics-based fracture animation for elastoplastic solids. In *Computer Graphics Forum*, volume 37, pages 112–124, 2018.

[30] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In *Eurographics Italian Chapter Conference*, 2008.

[31] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle, L. Tarbox, and F. Prior. The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository. *J Digit Imaging*, 26(6):1045–1057, Dec 2013.

[32] CyberGlove Systems. CyberGrasp, 2017. http://www.cyberglovesystems.com/cybergrasp.

[33] Mary F Dempsey, Barrie Condon, and Donald M Hadley. Mri safety review. In *Seminars in Ultrasound, CT and MRI*, volume 23, pages 392–401. Elsevier, 2002.

[34] C. M. Deniz, S. Xiang, S. Hallyburton, A. Welbeck, S. Honig, K. Cho, and G. Chang. Segmentation of the proximal femur from mr images using deep convolutional neural networks. *arXiv preprint arXiv:1704.06176*, 2017.

[35] Ali Hamadi Dicko, Tiantian Liu, Benjamin Gilles, Ladislav Kavan, Francois Faure, Olivier Palombi, and Marie-Paule Cani. Anatomy transfer. *ACM Trans. on Graphics (SIGGRAPH 2013)*, 32(6):188:1–188:8, 2013.

[36] D. E. Discher, D. J. Mooney, and P. W. Zandstra. Growth factors, matrices, and forces combine and control stem cells. *Science*, 324(5935):1673–1677, 2009.

[37] Daniel Charles Drucker. A definition of stable inelastic material. Technical report, DTIC Document, 1957.

[38] James S Duncan and Nicholas Ayache. Medical image analysis: Progress over two decades and the challenges ahead. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(1):85–106, 2000.

[39] M. S. Farvid, T. W. K. Ng, D. C. Chan, P. H. R. Barrett, and G. F. Watts. Association of adiponectin and resistin with adipose tissue compartments, insulin resistance and dyslipidaemia. *Diabetes, Obesity and Metabolism*, 7(4):406–413, 2005.

[40] A. Fenster and D. B. Downey. 3-d ultrasound imaging: a review. *IEEE Engineering in Medicine and Biology Magazine*, 15(6):41–51, 1996.

[41] Carlos Garre, Fernando Hernández, Antonio Gracia, and Miguel A Otaduy. Interactive simulation of a deformable hand for haptic rendering. In *IEEE World Haptics Conference (WHC)*, pages 239–244. IEEE, 2011.

[42] Thomas Gietzen, Robert Brylka, Jascha Achenbach, Katja zum Hebel, Elmar Schömer, Mario Botsch, Ulrich Schwanecke, and Ralf Schulze. A method for automatic forensic facial reconstruction based on dense statistics of soft tissue tissue thickness. *PLOS ONE*, 14:1, 2019.

[43] B. Gilles and N. Magnenat-Thalmann. Musculoskeletal mri segmentation using multi-resolution simplex meshes with medial representations. *Med. Image Anal.*, 14(3):291–302, 2010.

[44] Benjamin Gilles and Nadia Magnenat-Thalmann. Musculoskeletal mri segmentation using multi-resolution simplex meshes with medial representations. *Medical image analysis*, 14(3):291–302, 2010.

[45] Benjamin Gilles, Laurent Moccozet, and Nadia Magnenat-Thalmann. Anatomical modelling of the musculoskeletal system from mri. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*, pages 289–296, 2006.

[46] Benjamin Gilles, Lionel Reveret, and Dinesh Pai. Creating and animating subject-specific anatomical models. *Computer Graphics Forum*, 29(8):2340–2351, 2010.

[47] Leo Grady. Random walks for image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[48] Leo Grady. Random walks for image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[49] Agneta Gustus and Patrick van der Smagt. Evaluation of joint type modelling in the human hand. *Journal of Biomechanics*, 49(13):3097 – 3100, 2016.

[50] Shangchen Han, Beibei Liu, Robert Wang, Yuting Ye, Christopher D Twigg, and Kenrick Kin. Online optical marker-based hand tracking with deep labels. *ACM Transactions on Graphics (SIGGRAPH 2018)*, 37(4):166, 2018.

[51] Hang Si. TetGen: A Quality Tetrahedral Mesh Generator and a 3D Delaunay Triangulator, 2011.

[52] Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. *ACM Trans. on Graphics (SIGGRAPH 2018)*, 37(4):60:1–60:14, 2018.

[53] Jin Huang, Yiying Tong, Kun Zhou, Hujun Bao, and Mathieu Desbrun. Interactive shape interpolation through controllable dynamic deformation. *IEEE Trans. on Visualization and Computer Graphics*, 17(7):983–992, 2011.

[54] Z. Huang, N. A. Carr, and T. Ju. Variational implicit point set surfaces. *ACM Trans. on Graphics (SIGGRAPH 2019)*, 38(4), 2019.

[55] A.E. Ichim, P. Kadlecek, L. Kavan, and M. Pauly. Phace: Physics-based face modeling and animation. *ACM Trans. on Graphics (SIGGRAPH 2017)*, 36(4), 2017.

[56] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. *ACM Trans. on Graphics (SIGGRAPH 2005)*, 24(3):1134–1141, 2005.

[57] G. Irving, J. Teran, and R. Fedkiw. Invertible Finite Elements for Robust Simulation of Large Deformation. In *Symp. on Computer Animation (SCA)*, pages 131–140, 2004.

[58] ITK-SNAP. ITK-SNAP, 2018. http://www.itksnap.org/pmwiki/pmwiki.php.

[59] A. Jacobson, I. Baran, J. Popović, and O. Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Trans. on Graphics (TOG)*, 30(4):78, 2011.

[60] Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*, 2014.

[61] Petr Kadlecek, Alexandru-Eugen Ichim, Tiantian Liu, Jaroslav Krivanek, and Ladislav Kavan. Reconstructing personalized anatomical models for physics-based body animation. *ACM Trans. Graph.*, 35(6), 2016.

[62] A.I. Kapandji. *The physiology of the joints, 6th Edition, Vol. 1: The Upper Limb*. Elsevier Exclusive, 2009.

[63] L. Kavan, S. Collins, J. Zara, and C. O'Sullivan. Geometric skinning with approximate dual quaternion blending. *ACM Trans. on Graphics*, 27(4), 2008.

[64] Ladislav Kavan, Dan Gerszewski, Adam W. Bargteil, and Peter-Pike Sloan. Physics-inspired upsampling for cloth simulation in games. *ACM Trans. Graph.*, 30(4), 2011.

[65] Ali Emre Kavur, M. Alper Selver, Oğuz Dicle, Mustafa Barış, and N. Sinem Gezer. CHAOS - Combined (CT-MR) Healthy Abdominal Organ Segmentation Challenge Data, 2019.

[66] Baris Kayalibay, Grady Jensen, and Patrick van der Smagt. Cnn-based segmentation of medical imaging data. *arXiv preprint arXiv:1701.03056*, 2017.

[67] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. on Graphics (TOG)*, 32(3), 2013.

[68] Junggon Kim and Nancy S Pollard. Fast simulation of skeleton-driven deformable body characters. *ACM Trans. on Graphics (TOG)*, 30(5):121, 2011.

[69] Jonathan P King, Dominik Bauer, Cornelia Schlagenhauf, Kai-Hung Chang, Daniele Moro, Nancy Pollard, and Stelian Coros. Design. fabrication, and evaluation of tendon-driven multi-fingered foam hands. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, pages 1–9, 2018.

[70] Paul G. Kry, Doug L. James, and Dinesh K. Pai. EigenSkin: Real Time Large Deformation Character Skinning in Hardware. In *Proc. of the Symp. on Comp. Animation 2002*, pages 153–160, 2002.

[71] Tsuneya Kurihara and Natsuki Miyata. Modeling deformable human hands from medical images. In *Symp. on Computer Animation (SCA)*, pages 355–363, 2004.

[72] LeapMotion, 2017. https://www.leapmotion.com.

[73] S. H. Lee, E. Sifakis, and D. Terzopoulos. Comprehensive Biomechanical Modeling and Simulation of the Upper Body. *ACM Trans. on Graphics*, 28(4):99:1–99:17, 2009.

[74] Seunghwan Lee, Ri Yu, Jungnam Park, Mridul Aanjaneya, Eftychios Sifakis, and Jehee Lee. Dexterous manipulation and control with volumetric muscles. *ACM Transactions on Graphics (SIGGRAPH 2018)*, 37(4):57:1–57:13, 2018.

[75] J. P. Lewis, Matt Cordner, and Nickson Fong. Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. In *Proc. of ACM SIGGRAPH 2000*, pages 165–172, July 2000.

[76] Duo Li, Shinjiro Sueda, Debanga R Neog, and Dinesh K Pai. Thin skin elastodynamics. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 32(4):49:1–49:9, 2013.

[77] Hao Li, Robert W. Sumner, and Mark Pauly. Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum (Proc. SGP'08)*, 27(5), July 2008.

[78] Libin Liu, KangKang Yin, Bin Wang, and Baining Guo. Simulation and control of skeleton-driven soft body characters. *ACM Trans. on Graphics (SIGGRAPH Asia 2013)*, 32(6):215, 2013.

[79] N. Magnenat-Thalmann, R. Laperrire, and D. Thalmann. Joint-dependent local deformations for hand animation and object grasping. In *Proc. of Graphics Interface*, 1988.

[80] Joe Mancewicz, Matt L. Derksen, Hans Rijpkema, and Cyrus A. Wilson. Delta mush: Smoothing deformations while preserving detail. In *Proceedings of the Fourth Symposium on Digital Production*, DigiPro '14, pages 7–11, 2014.

[81] G. Marai, D. Laidlaw, J. Coburn, M. Upal, and J. Crisco. A 3d method for segmenting and registering carpal bones from ct volume images. In *Proc. of Annual Meeting of the American Society of Biomechanics*, 2003.

[82] A. McAdams, Y. Zhu, A. Selle, M. Empey, R. Tamstorf, J. Teran, and E. Sifakis. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. on Graphics (SIGGRAPH 2011)*, 30(4), 2011.

[83] T. McInerney and D. Terzopoulos. Deformable models. In I. Bankman, editor, *Handbook of Medical Image Processing and Analysis (2nd Edition)*, chapter 8, pages 145–166. 2008.

[84] Tim McInerney and Demetri Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2):91–108, 2008.

[85] Fernand Meyer. Color image segmentation. In *International Conf. on Image Processing and its Applications*, pages 303–306. IET, 1992.

[86] Eder Miguel, Derek Bradley, Bernhard Thomaszewski, Bernd Bickel, Wojciech Matusik, Miguel A. Otaduy, and Steve Marschner. Data-driven estimation of cloth simulation models. *Computer Graphics Forum (Proc. of Eurographics)*, 31(2), may 2012.

[87] Aslan Miriyev, Kenneth Stack, and Hod Lipson. Soft material for soft actuators. *Nature Communications*, 8(596), 2017.

[88] N. Miyata, M. Kouch, M. Mochimaru, and T. Kurihara. Finger joint kinematics from mr images. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2750–2755, 2005.

[89] M. Müller and M. Gross. Interactive Virtual Materials. In *Proc. of Graphics Interface 2004*, pages 239–246, 2004.

[90] M. Müller, B. Heidelberger, M. Teschner, and M. Gross. Meshless Deformations Based on Shape Matching. In *Proc. of ACM SIGGRAPH 2005*, pages 471–478, Aug 2005.

[91] G. Niculescu, J.L. Nosher, M.D. Schneider, and D.J. Foran. A deformable model for tracking tumors across consecutive imaging studies. *Int. J. of Computer-Assisted Radiology and Surgery*, 4(4):337–347, 2009.

[92] NimbleVR, 2012. http://nimblevr.com.

[93] James F. O'Brien, Adam W. Bargteil, and Jessica K. Hodgins. Graphical modeling and animation of ductile fracture. In *Proceedings of ACM SIGGRAPH 2002*, pages 291–294, 2002.

[94] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical recipes: The art of scientific computing*. Cambridge University Press, Cambridge, UK, third edition, 2007.

[95] RadiologyInfo. Radiation Dose in X-Ray and CT Exams, 2018. https://www.radiologyinfo.org/en/pdf/safety-xray.pdf.

[96] T. Rhee, U. Neumann, J. Lewis, and K. S. Nayak. Scan-based volume animation driven by locally adaptive articulated registrations. *IEEE Trans. on Visualization and Computer Graphics*, 17(3):368–379, 2011.

[97] Taehyun Rhee, J.P. Lewis, and Ulrich Neumann. Real-time weighted pose-space deformation on the gpu. In *Proc. of Eurographics 2006*, volume 25, 2006.

[98] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Trans. on Graphics (SIGGRAPH Asia 2017)*, 36(6):245:1–245:17, 2017.

[99] Alexandru Rusu. Segmentation of bone structures in magnetic resonance images (mri) for human hand skeletal kinematics modelling. Master's thesis, German Aerospace Center, 2011.

[100] Prashant Sachdeva, Shinjiro Sueda, Susanne Bradley, Mikhail Fain, and Dinesh K. Pai. Biomechanical simulation and control of hands and tendinous systems. *ACM Trans. Graph.*, 34(4):42:1–42:10, 2015.

[101] Yusuf Sahillioğlu and Ladislav Kavan. Skuller: A volumetric shape registration algorithm for modeling skull deformities. *Medical image analysis*, 23(1):15–27, 2015.

[102] Shunsuke Saito, Zi-Ye Zhou, and Ladislav Kavan. Computational bodybuilding: Anatomically-based modeling of human bodies. *ACM Trans. on Graphics (SIGGRAPH 2015)*, 34(4), 2015.

[103] Cornelia Schlagenhauf, Dominik Bauer, Kai-Hung Chang, Jonathan P King, Daniele Moro, Stelian Coros, and Nancy Pollard. Control of tendon-driven soft foam robot hands. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, pages 1–7, 2018.

[104] J. Schmid, E. Gobbetti J. A. I. Guitián, and N. Magnenat-Thalmann. A gpu framework for parallel segmentation of volumetric images using discrete deformable models. *The Visual Computer*, 27:85–95, 2011.

[105] J. Schmid, A. Sandholm, F. Chung, D. Thalmann, H. Delingette, and N. Magnenat-Thalmann. Musculoskeletal simulation model generation from mri data sets and motion capture data. *Recent Advances in the 3D Physiological Human*, pages 3–19, 2009.

[106] Jérôme Schmid, Jinman Kim, and Nadia Magnenat-Thalmann. Robust statistical shape models for mri bone segmentation in presence of small field of view. *Medical image analysis*, 15(1):155–168, 2011.

[107] Jérôme Schmid and Nadia Magnenat-Thalmann. Mri bone segmentation using deformable models and shape priors. In *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention*, pages 119–126, 2008.

[108] Jérôme Schmid and Nadia Magnenat-Thalmann. Mri bone segmentation using deformable models and shape priors. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, pages 119–126, 2008.

[109] Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Trans. on Graphics (SIGGRAPH 2005)*, 24(3):417–425, August 2005.

[110] Breannan Smith, Fernando De Goes, and Theodore Kim. Stable neo-hookean flesh simulation. *ACM Trans. Graph.*, 37(2):12:1–12:15, 2018.

[111] Ole Vegard Solberg, Frank Lindseth, Hans Torp, Richard E. Blake, and Toril A. Nagelhus Hernes. Freehand 3d ultrasound reconstruction algorithms: A review. *Ultrasound in Medicine and Biology*, 33(7):991 – 1009, 2007.

[112] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H-P Seidel. Laplacian surface editing. In *Symp. on Geometry processing*, pages 175–184, 2004.

[113] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symp. on Geometry Processing*, volume 4, pages 109–116, 2007.

[114] Stephcavs. KidneyFullBody 1.0.0: Full CT scan of body, 2019. https://www.embodi3d.com/files/file/26389-kidneyfullbody/.

[115] Georg Stillfried. *Kinematic modelling of the human hand for robotics*. PhD thesis, Technische Universität München, 2015.

[116] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle. A material point method for snow simulation. *ACM Trans. on Graphics (SIGGRAPH 2013)*, 32(4):102:1–102:10, 2013.

[117] Shinjiro Sueda, Andrew Kaufman, and Dinesh K. Pai. Musculotendon simulation for hand animation. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 27(3), 2008.

[118] Robert W Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Trans. on Graphics (SIGGRAPH 2004)*, 23(3):399–405, 2004.

[119] J. Sylvester. Sur l'equations en matrices p x = x q. *C. R. Acad. Sci. Paris.*, 99(2):67–71, 115–116, 1884.

[120] G. Székely, A. Kelemen, C. Brechbühler, and G. Gerig. Segmentation of 2-D and 3-D objects from MRI volume data using constrained elastic deformations of flexible Fourier contour and surface models. *Medical Image Analysis*, 1(1):19–34, 1996.

[121] Tissue. Weta Digital: Tissue Muscle and Fat Simulation System, 2013.

[122] Turbosquid, 2019. www.turbosquid.com.

[123] G. Turk and J. O'Brien. Shape transformation using variational implicit functions. In *Proc. of ACM SIGGRAPH 1999*, pages 335–342, 1999.

[124] C. Twigg and Z. Kačić-Alesić. Point cloud glue: constraining simulations using the procrustes transform. In *Symp. on Computer Animation (SCA)*, pages 45–54, 2010.

[125] Christopher D. Twigg and Zoran Kačić-Alesić. Optimization for sag-free simulations. In *Proc. of the 2011 ACM SIGGRAPH/Eurographics Symp. on Computer Animation*, pages 225–236, 2011.

[126] U.S. National Library of Medicine. The visible human project, 1994. http://www.nlm.nih.gov/research/visible/.

[127] Rodolphe Vaillant, Gäel Guennebaud, Loïc Barthe, Brian Wyvill, and Marie-Paule Cani. Robust iso-surface tracking for interactive character skinning. *ACM Trans. on Graphics (SIGGRAPH Asia 2014)*, 33(6):189:1–137:11, 2014.

[128] Patrick van der Smagt and Georg Stillfried. Using mri data to compute a hand kinematic model. In *Conf. on Motion and Vibration Control (MOVIC)*, 2008.

[129] Pascal Volino, Nadia Magnenat-Thalmann, and Francois Faure. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph.*, 28(4), September 2009.

[130] VTK. VTK, 2018. https://www.vtk.org/.

[131] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[132] Andrea Walther and Andreas Griewank. Getting started with adol-c. In *Combinatorial scientific computing*, pages 181–202, 2009.

[133] Bin Wang, Longhua Wu, KangKang Yin, Uri Ascher, Libin Liu, and Hui Huang. Deformation capture and modeling of soft objects. *ACM Transactions on Graphics (TOG) (SIGGRAPH 2015)*, 34(4):94, 2015.

[134] Bohan Wang, George Matcuk, and Jernej Barbič. Hand modeling and simulation using stabilized magnetic resonance imaging. *ACM Trans. on Graphics (SIGGRAPH 2019)*, 38(4), 2019.

[135] R. Y. Wang, S. Paris, and J. Popović. 6d hands: Markerless hand tracking for computer aided design. In *ACM User Interface Software and Technology (UIST)*, 2011.

[136] Robert Y. Wang and Jovan Popović. Real-time hand-tracking with a color glove. *ACM Trans. on Graphics (SIGGRAPH 2009)*, 28(3):63:1–63:8, 2009.

[137] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. Linear subspace design for real-time shape deformation. *ACM Transactions on Graphics (TOG) (SIGGRAPH 2015)*, 34(4):57, 2015.

[138] Robert E. Watson. Lessons learned from mri safety events. *Current Radiology Reports*, 3(10):37, Aug 2015.

[139] C. Wex, S. Arndt, A. Stoll, C. Bruns, and Y. Kupriyanova. Isotropic incompressible hyperelastic models for modelling the mechanical behaviour of biological tissues: a review. *Biomedical Engineering / Biomedizinische Technik*, 60(6):577–592, 2015.

[140] Nkenge Wheatland, Yingying Wang, Huaguang Song, Michael Neff, Victor Zordan, and Sophie Jörg. State of the art in hand and finger modeling and animation. In *Computer Graphics Forum*, volume 34, pages 735–760, 2015.

[141] Max A. Woodbury. Inverting modified matrices. *Memorandum Rept. 42, Statistical Research Group, Princeton University, Princeton, NJ*, page 4pp, 1950.

[142] Wrap3. Nonlinear iterative closest point mesh registration software, 2018. https://www.russian3dscanner.com.

[143] Shanxin Yuan, Qi Ye, Björn Stenger, Siddhant Jain, and Tae-Kyun Kim. Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2605–2613, 2017.

[144] FE Zajac. Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control. *Critical reviews in biomedical engineering*, 17(4):359—411, 1989.

[145] Hao Zhang. Discrete combinatorial laplacian operators for digital geometry processing. In *Proceedings of SIAM Conference on Geometric Design and Computing*, pages 575–592. Nashboro Press, 2004.

[146] Ziva Dynamics. Male Virtual Human "Max", 2019. http://zivadynamics.com/ziva-characters.

[147] Zygote. Zygote body, 2016. http://www.zygotebody.com.

Appendices

A Plastic strain Laplacian and its nullspace

Let $\mathbf{L}^{\text{sc}} \in \mathbb{R}^{m \times m}$ denote the discrete mesh Laplacian for *scalar* fields on mesh tetrahedra [145],

$$\mathbf{L}^{\text{sc}}{}_{i,j} = \begin{cases} \#\text{adjacent tets} & \text{if } i = j, \\ -1 & \text{if } i \neq j, \text{ and } i, j \text{ are adjacent}, \\ 0 & \text{otherwise.} \end{cases} \tag{A.1}$$

Given a plastic strain state $\mathbf{s} = [\mathbf{s_1}, \mathbf{s_2}, \mathbf{s_3}, \mathbf{s_4}, \mathbf{s_5}, \mathbf{s_6}] \in \mathbb{R}^{6m}$, where $\mathbf{s_i} \in \mathbb{R}^m$, define the *plastic strain Laplacian*

$$\mathbf{Ls} = [\mathbf{L}^{\text{sc}}\mathbf{s_1}, \sqrt{2}\mathbf{L}^{\text{sc}}\mathbf{s_2}, \sqrt{2}\mathbf{L}^{\text{sc}}\mathbf{s_3}, \mathbf{L}^{\text{sc}}\mathbf{s_4}, \sqrt{2}\mathbf{L}^{\text{sc}}\mathbf{s_5}, \mathbf{L}^{\text{sc}}\mathbf{s_6}], \tag{A.2}$$

where the $\sqrt{2}$ were added to account for the fact that s_2, s_3, s_5 control two entries in the symmetric matrix $F_p \in \mathbb{R}^{3\times 3}$.

Lemma: Assume that the tet mesh $\mathscr{M}$ has a single connected component. Then, the nullspace of $\mathbf{L}$ is 6-dimensional and consists of vectors $\psi_i = [\mathbf{s_1}, \ldots, \mathbf{s_6}]$ where $\mathbf{s_j} \in \mathbb{R}^m$ is all ones when $j = i$, and all zeros otherwise.

Proof: First, observe that $\mathbf{L}^{\text{sc}}$ is symmetric positive semi-definite with a single nullspace vector, namely the vector of all 1s. This follows from the identity

$$\mathbf{x}^T\mathbf{L}^{\text{sc}}\mathbf{x} = \sum_{i \text{ and } j \text{ adjacent}} (x_i - x_j)^2, \tag{A.3}$$

i.e., $\mathbf{x}^T\mathbf{L}^{\text{sc}}\mathbf{x} = 0$ is only possible if all x_i are the same.

We have $0 = \mathbf{s}^T\mathbf{L}\mathbf{s} = \sum_{i=1}^{6} \xi_i \mathbf{s_i}^T\mathbf{L}^{\text{sc}}\mathbf{s_i}$, where $\xi_i = \sqrt{2}$ for $i = 2,3,5$; and 1 otherwise. Because $\mathbf{L}^{\text{sc}}$ is symmetric positive semi-definite, each $\mathbf{s_i}$ must either be $\mathbf{0}$ or a non-zero nullspace vector of $\mathbf{L}^{\text{sc}}$, i.e., a vector of all 1s. A linearly independent orthonormal nullspace basis emerges when we have a vector for all 1s for exactly one i. There are 6 such choices, giving the vectors ψ_i; we normalize them by dividing with $\sqrt{m}$. ■

B First and second derivatives of elastic energy with respect to plastic strain

For convenience, we denote $F_{p,i}$ as $i-$th entry of the vector $\text{vec}(F_p) \in \mathscr{R}^9$. The first-order derivatives are

$$\frac{\partial \mathscr{E}}{\partial x_i} = V\frac{\partial \psi}{\partial F_e} : \frac{\partial F_e}{\partial x_i} = VP : \frac{\partial F_e}{\partial x_i}, \tag{B.1}$$

$$\frac{\partial \mathscr{E}}{\partial F_{p,i}} = V\frac{\partial \psi}{\partial F_{p,i}} + \frac{\partial V}{\partial F_{p_i}}\psi = VP : \frac{\partial F_e}{\partial F_{p,i}} + \frac{\partial V}{\partial F_{p_i}}\psi, \quad \text{where} \tag{B.2}$$

$$\frac{\partial F_e}{\partial x_i} = \frac{\partial F}{\partial x_i}F_p^{-1}, \quad \frac{\partial F_e}{\partial F_{p,i}} = F\frac{\partial F_p^{-1}}{\partial F_{p,i}}, \quad \text{and} \tag{B.3}$$

$$\frac{\partial V}{\partial F_{p,i}} = \frac{\partial \left|F_p\right|}{\partial F_{p_i}}V_0. \tag{B.4}$$

Here, P is the first Piola-Kirchhoff stress tensor and $\partial F/\partial x$ is a constant matrix commonly used in the equations for FEM simulation. For the second-order derivatives, we first compute $\partial^2\mathscr{E}/\partial x^2$. This is the tangent stiffness matrix in the FEM simulation under a fixed F_p. It is computed as

$$\frac{\partial^2 \mathscr{E}}{\partial x_i \partial x_j} = V\frac{\partial F_e}{\partial x_j}^T : \frac{\partial P}{\partial F_e} : \frac{\partial F_e}{\partial x_i}. \tag{B.5}$$

Here, $\partial P/\partial F_e$ is a standard term in FEM nonlinear elastic simulation; it only depends on the strain-stress law (the material model). Next, we compute $\partial^2\mathscr{E}/(\partial x \partial F_p)$,

$$\frac{\partial^2 \mathscr{E}}{\partial x_i \partial F_{p,j}} = \frac{\partial V}{\partial F_{p,j}} \left(P : \frac{\partial F_e}{\partial x_i} \right) + \tag{B.6}$$

$$V \frac{\partial F_e}{\partial F_{p,j}}^T : \frac{\partial P}{\partial F_e} : \frac{\partial F_e}{\partial x_i} + VP \frac{\partial^2 F_e}{\partial x_i \partial F_{p,j}}, \quad \text{where} \tag{B.7}$$

$$\frac{\partial^2 F_e}{\partial x_i \partial F_{p,j}} = \frac{\partial F}{\partial x_i} \frac{\partial F_p^{-1}}{\partial F_{p,j}}. \tag{B.8}$$

Finally, we have

$$\frac{\partial^2 \mathscr{E}}{\partial F_{p,i} \partial F_{p,j}} = \frac{\partial^2 V}{\partial F_{p,i} \partial F_{p,j}} \psi + \tag{B.9}$$

$$V \left(P : \frac{\partial^2 F_e}{\partial F_{p,i} \partial F_{p,j}} + \frac{\partial F_e}{\partial F_{p,j}}^T : \frac{\partial P}{\partial F_e} : \frac{\partial F_e}{\partial F_{p,i}} \right) + \tag{B.10}$$

$$\frac{\partial V}{\partial F_{p,j}} \frac{\partial \psi}{\partial F_{p,i}} + \frac{\partial V}{\partial F_{p,i}} \frac{\partial \psi}{\partial F_{p,j}}, \quad \text{where} \tag{B.11}$$

$$\frac{\partial^2 V}{\partial F_{p,i} \partial F_{p,j}} = \frac{\partial^2 |F_p|}{\partial F_{p,i} \partial F_{p,j}} V_0, \tag{B.12}$$

$$\frac{\partial^2 F_e}{\partial F_{p,i} \partial F_{p,j}} = F \frac{\partial^2 F_p^{-1}}{\partial F_{p,i} \partial F_{p,j}}. \tag{B.13}$$

The quantities ψ, P and $\partial P/\partial F_e$ are determined by the chosen elastic material model. After computing the above derivatives, there is still a missing link between F_p and s. Because we want to directly optimize s, we also need the derivatives of $\mathscr{E}(F_p(s), x)$ with respect to s. From Equation 5.2 we can see that F_p is linearly dependent on s. Therefore, so we can define a matrix Y such that $\text{vec}(F_p) = Ys$. Then all the derivatives can be easily transferred to derivation by s by multiplying with Y.

C Proof of singular lemma

Statement (i) follows from well-known linear algebra facts $\mathscr{R}(A) = \mathscr{N}(A^T)^\perp$ and $\dim(\mathscr{N}(A)) + \dim(\mathscr{R}(A)) = p$, and the symmetry of A. As per (ii), A maps $\mathscr{R}(A)$ into itself, and no vector from $\mathscr{R}(A)$ maps to zero, hence the restriction of A to $\mathscr{R}(A)$ is invertible, establishing a unique solution to $Ax = b$ with the property that $x \perp \psi_i$ for all $i = 1, \ldots, k$. This unique solution is the minimizer of

$$\min_x \frac{1}{2} x^T A x - b^T x \tag{C.1}$$

$$\text{s.t.} \quad \psi_i^T x = 0 \text{ for all } i = 1, \ldots, k. \tag{C.2}$$

When expressed using Lagrange multipliers, this gives Equation 5.15. Suppose $x = n + r$ is another solution and $n \in \mathscr{N}(A)$ and $r \in \mathscr{R}(A)$. Then $b = Ax = Ar$ and hence r is the unique solution from Equation 5.15. The vector n can be an arbitrary nullspace vector, proving the last statement of (ii). As per (iii), suppose we have $0 = B(n + r) = Ar + \sum_{i=1}^k \frac{\lambda_i}{\alpha_i} (\psi_i^T n) \psi_i$. Observe that the first summand is in $\mathscr{R}(A)$ and the second in $\mathscr{N}(A)$. Hence, $B(n+r)$ can only be zero if both summands are zero. $Ar = 0$ implies $r = 0$. The second summand can only be zero if $n \perp \psi_i$ for each i, which implies that $n = 0$. Hence, B is invertible. The last statement of (iii) can be verified by expanding $\left(A + \sum_{i=1}^k \alpha_i \psi_i \psi_i^T\right)\left(x + \sum_{i=1}^k \frac{\lambda_i}{\alpha_i} \psi_i\right)$. ■

D Proof of nullspace lemma

We are trying to prove that $\mathbf{K}(\mathbf{F_p}(\mathbf{s}), \mathbf{x})$ is 6-dimensional for any $\mathbf{x}$ that solves $\mathbf{f}_\mathrm{e}(\mathbf{s}, \mathbf{x}) = 0$. First, if $\mathbf{x}$ is a solution, then translating all vertices of the object by the same constant 3-dimensional vector is also a solution. This means that vector $\psi_i := (e_i, e_i, \ldots, e_i)$ is in the nullspace of $\mathbf{K}$, where $e_i \in \mathbb{R}^3$ is the i-th standard basis vector, for $i = 1, 2, 3$. Now, suppose we rotate the object with an infinitesimal rotation $X \mapsto X + e_i \times X$.

Observe that for general plastic strains $\mathbf{s}$, the elastic forces in each individual tet are not zero even in the equilibrium $\mathbf{x}$; but the contributions of elastic forces on a tet mesh vertex from all adjacent tets sum to zero. As we rotate the object, the forces contributed by adjacent tets to a specific tet mesh vertex rotate by the same rotation in each tet. Therefore, as these forces sum to zero, they continue to sum to zero even under the rotation (see Figure D.1). This means that the vector of infinitesimal displacements $\psi_{3+i} := [e_i \times x_1, e_i \times x_2, \ldots, e_i \times x_n]$ induced by the infinitesimal rotation is in the nullspace of $\mathbf{K}$, for each $i = 1,2,3$. Here, x_i are the components of $\mathbf{x} = [x_1, x_2, \ldots, x_n]$. The vectors ψ_i, $i = 1,2,3,4,5,6$, form the nullspace of $\mathbf{K}$. ■

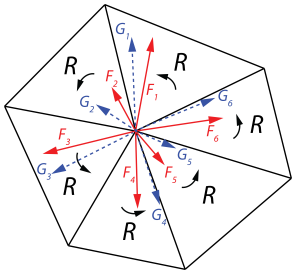


Figure D.1: **Illustration of the nullspace proof.** The original forces F_i sum to zero. We have $G_i = RF_i$; note that R is the same for all tets. Therefore, the rotated forces G_i also sum to zero. Hence, there is no change in the internal elastic force under a rotation—that is, infinitesimal rotations are in the nullspace of $\mathbf{K}$.

Finally, we inform the reader that the nullspace of $\mathbf{K}(\mathbf{F_p}(\mathbf{s}), \mathbf{x})$ is only 3-dimensional if $\mathbf{x}$ is *not* an elastic equilibrium. In this case, only translations are in the nullspace. Infinitesimal rotations are not in the nullspace because under an infinitesimal rotation, the nonzero elastic forces $\mathbf{f}_e$ rotate, i.e., they do not remain the same. The assumption of $\mathbf{x}$ being the equilibrium shape is therefore crucial (and is satisfied in our method).

E Second derivative of polar decomposition

To compute the second-order derivatives, we differentiate

$$\frac{\partial F}{\partial F_i} = \frac{\partial R}{\partial F_i} S + R \frac{\partial S}{\partial F_i}, \tag{E.1}$$

$$\frac{\partial^2 F}{\partial F_i \partial F_j} = \frac{\partial^2 R}{\partial F_i \partial F_j} S + \frac{\partial R}{\partial F_i}\frac{\partial S}{\partial F_j} + \frac{\partial R}{\partial F_j}\frac{\partial S}{\partial F_i} + R\frac{\partial^2 S}{\partial F_i \partial F_j}, \tag{E.2}$$

$$\frac{\partial^2 R}{\partial F_i \partial F_j} = \left(-R\frac{\partial^2 S}{\partial F_i \partial F_j} - \frac{\partial R}{\partial F_j}\frac{\partial S}{\partial F_i} - \frac{\partial R}{\partial F_i}\frac{\partial S}{\partial F_j} \right) S^{-1}. \tag{E.3}$$

To compute $\partial^2 R/(\partial F_i \partial F_j)$, we need to compute $\partial^2 S/(\partial F_i \partial F_j)$ first. This can be derived in the same way as for $\partial S/\partial F_i$. Starting from Equation 5.22, we have

$$\frac{\partial^2 F^T F}{\partial F_i \partial F_j} = \frac{\partial^2 S}{\partial F_i \partial F_j} S + \frac{\partial S}{\partial F_i}\frac{\partial S}{\partial F_j} + \frac{\partial S}{\partial F_j}\frac{\partial S}{\partial F_i} + S\frac{\partial^2 S}{\partial F_i \partial F_j}. \tag{E.4}$$

We can now solve a similar Sylvester equation

$$\mathrm{vec}(\frac{\partial^2 S}{\partial F_i \partial F_j}) = (S \oplus S)^{-1}\,\mathrm{vec}(C), \tag{E.5}$$

$$C = \frac{\partial^2 F^T F}{\partial F_i \partial F_j} - \frac{\partial S}{\partial F_i}\frac{\partial S}{\partial F_j} - \frac{\partial S}{\partial F_j}\frac{\partial S}{\partial F_i}. \tag{E.6}$$

F Formulas for $\mathbf{A_k}$,$\mathbf{b_k}$,$\mathbf{c_k}$ (Equation 5.6)

Let the attachment k be embedded into a tetrahedron t_k with barycentric weights $\left[w_1^k, w_2^k, w_3^k, w_4^k\right]$. We have

$$\mathbf{A_k} = \begin{bmatrix} w_1^k I_3 & w_2^k I_3 & w_3^k I_3 & w_4^k I_3 \end{bmatrix} S^k \in \mathbb{R}^{3\times 3n} \tag{F.1}$$

$$\mathbf{b_k} = -y_k \in \mathbb{R}^3, \tag{F.2}$$

where $y_k \in \mathbb{R}^3$ is the attachment's target position, $I_3 \in \mathbb{R}^{3\times 3}$ is the identity matrix, and $S^k \in \mathbb{R}^{12\times 3n}$ is a selection matrix that selects the positions of vertices of t_k. The scalar $\mathbf{c_k}$ is the weight of the attachment k. Equivalent formulas apply to landmarks and ICP markers. The attachment energy and forces are,

$$\mathscr{E}_\mathrm{a}(\mathbf{x}) = \frac{1}{2}\sum_k \mathbf{c_k}\|\mathbf{A_k x} + \mathbf{b_k}\|^2, \tag{F.3}$$

$$f_\mathrm{a}(\mathbf{x}) = \frac{d\mathscr{E}_\mathrm{a}}{d\mathbf{x}} = \sum_k \mathbf{c_k}\mathbf{A_k^T}(\mathbf{A_k x} + \mathbf{b_k}). \tag{F.4}$$

G Meeting constraints by scaling the elastic stiffness

In this section, we demonstrate that scaling the elastic stiffness cannot be used to "better" meet the constraints. We illustrate this on a simple toy problem of linear elasticity with linear constraints. Denote the vertex displacements by $u \in \mathbb{R}^{3n}$ and the stiffness matrix by $K \in \mathbb{R}^{3n \times 3n}$.

First, consider scaling the stiffness ($K \to \alpha K$) when using hard constraints:

$$\min_u < Ku, u > \tag{G.1}$$

$$\text{s.t. } Cu = b \tag{G.2}$$

v.s.

$$\min_u < (\alpha K)u, u > \tag{G.3}$$

$$\text{s.t. } Cu = b \tag{G.4}$$

produces same u for any scalar $\alpha > 0$.

Similarly, consider scaling stiffness ($K \to \alpha K$) when using soft constraints:

$$\min_u < (\alpha K)u, u > +\beta ||Cu - b||^2 \tag{G.5}$$

produces same u as

$$\min_u < Ku, u > +\beta/\alpha ||Cu - b||^2. \tag{G.6}$$

Therefore, scaling the stiffness does not provide any more expressive shape deformation power than tweaking β in:

$$\min_u < Ku, u > +\beta ||Cu - b||^2. \tag{G.7}$$

As one tweaks β in the above, one can choose between (A) meeting the constraints well (when $\beta >> 1$), or (B) produce smooth deformation (when $\beta << 1$). In option (A), one obtains sharp non-smooth deformation (”spikes”), and in option (B) one does not meet the constraints. In the middle territory, one doesn't meet either of these goals well, as seen in Figure 1.3,f. So, in summary, merely increasing the elastic stiffness does not help with better satisfying the constraints.