Universidad Rey Juan Carlos

Tesis Doctoral

Data-driven design of expressive reduced models for deformation and contact

Autor:

Cristian Romero Garcia

Directores:

Miguel A. Otaduy Tristán

Dan Casas Guix

Programa de Doctorado en Tecnologías de la Información y las Comunicaciones

Escuela Internacional de Doctorado

2024

Abstract

Deformable materials are widely extended in our daily interactions. They are the source for the manufacturing of many everyday objects, in addition to playing a very important role in our bodies, considering their deformable nature. In this context, computational modeling of deformable objects has been widely explored in computer graphics, with a wide variety of applications in simulators and virtual environments. During the last decades, the community has researched from efficient to very precise methods, but unfortunately, there is yet no general deformation model that is suitable for all practical use cases in the field. The goal of this thesis is to develop useful simulation methods for deformable objects that are fast and accurate, with special focus on efficiently capturing proper dynamic behaviors, external interactions and deformations produced by contact. We achieve this goal thanks to an intelligent combination of data-driven techniques with fast physics-based methods. Throughout the thesis we also address common limitations of data-driven methods by designing novel problem representations. This enables us to obtain useful models for the efficient simulation of deformable objects and materials.

Acknowledgements

Me gustaría dar las gracias a todos los que me habéis acompañado en esta intensa aventura. Mi motivación inicial para empezar el doctorado fue comprender todos los detalles de este apasionante mundo que es la informática gráfica. Indudablemente, ahora sé más de lo que sabía cuando empecé, pero aquel objetivo inicial se ha ido matizando, y como mucha otra gente antes que yo, cada vez tengo más presente que lo importante es el camino. Sin duda, lo que más me enorgullece de estos años es la evolución que he tenido como persona, y no tengo dudas de que en gran medida esto es un logro colectivo.

Miguel, gracias por guiarme con brazo firme durante todo este tiempo y hacer que las cosas sean más fáciles de lo que parecen. No sé cuál fue tu criterio para darme esta oportunidad, pero valoro mucho la confianza que has depositado en mí durante todo este viaje.

Dan, has estado presente en gran parte de mi progreso, siempre aportando algo para que los proyectos salgan adelante. Tu capacidad resolutiva es envidiable.

Jesús, admiro tu dedicación y ayuda incondicional. Has sido mi figura fraterna en lo que a investigación se refiere. Aunque no se si puede tener que ver con que los dos seamos rubios y que compartamos el mismo "padre". No lo sé.

Thanks to my hosts Maurizio and Bernd, and to all the people who have worked with me at Meta and Google during my internships. Thank you for giving me the opportunity to continue my education and allowing me to be part of these great companies during some months. I take with me good memories of interns and employees. Many exceptional people.

Gracias a los compañeros del MsLab. A los que ya se fueron, a los que siguen y a los que acaban de llegar. Habéis hecho que el ambiente de compañerismo en los despachos sea inigualable. Muchas conversaciones, comidas, paseos, cenas, lunchs, escaladas, post cenas, rituales de microondas, anécdotas, sesiones de pizarra,... Sin duda uno de los recuerdos más cálidos de toda esta experiencia. Se os echa de menos.

Gracias a los compañeros de piso. Me alegro de que nuestros caminos se cruzaran y también me alegro cuando pienso que la vida sigue para cada uno de nosotros. Continúa asombrándome cuanto se puede llegar a apreciar a la gente con la que se convive de cerca, incluso cuando no existe mucha afinidad, o aún peor, cuando alguno aspira a ser músico. Muchos recuerdos dentro de esas cuatro paredes.

Gracias a los compañeros del norte. Tampoco me voy a engañar, los de verdad no sois muchos y nuestros encuentros han sido poco frecuentes desde que bajé. Aun así, las veces que nos juntamos son una bocanada de aire fresco, y eso me basta.

Y por último, muchas gracias a mis padres, que me habéis dado todo y lo seguís haciendo. No creo que alargarme ayude a plasmar lo que siento, pero que este austero agradecimiento sirva para que lecturas futuras me hagan recordar el cariño que os mereceis.

Contents

Figures

Tables

1 Introduction

Many of the objects we use in our daily life are deformable. In fact, deformable objects and materials are a fundamental part in the manufacturing of fashion, furniture and tools, among many other things. One of the main reasons for being so widely used is their inherent mechanical behavior. Soft materials can change their shape, tend to be more comfortable, ergonomic, and are better suited for interactions. Not only in the manufacturing of objects, deformable matter is also very common in living creatures, like humans. Except for bones, our bodies are mostly made up of soft-tissues such as muscle and fat. Indeed, we could say that our bodies are one of the most important deformable objects in our daily interactions. When we change our pose, make a facial expression or grasp an object with our hands, our soft tissue deforms due to muscular activations and external contacts.

Therefore, a proper understanding and modeling of the deformable materials is essential for the progress of different applications in computer graphics, mechanical engineering and biomechanics. Beyond applications in the real world, modeling deformable objects is also important in virtual environments such as games, visual effects, training simulators or virtual reality. In this context, realistic virtual worlds demand complex deformable interactions, with avatars, items and clothing requiring to be properly modeled as deformable objects for a complete immersive experience.

In the area of computer graphics, which is the one covered by this thesis, various methods have been proposed for modeling deformable objects, with different scope and application domains. Mathematical models describing the behavior of continuous media have been known for a long time, but until recently they could only be used for very simple examples. This has changed since the advent of modern computers, and nowadays we have a wide variety of computational models and simulations available for this task, ranging from geometric deformation models to physically-based methods.

Unfortunately, there is no general deformation model that is suitable for all use cases in computer graphics. Some methods can compute accurate simulations with realistic object dynamics and detailed deformations produced by contacts and forces. Although very useful in applications demanding high accuracy, they can be difficult to setup, they offer no guarantees of stability, and they are not practical in interactive applications due to their high computational cost. Other methods succeed in reducing the computational cost, simplifying the models and approximating the deformations. These alternatives tend to be fast, easy to implement, and can be applied in interactive contexts, but the necessary simplifications often mean that the generated results suffer limitations. This is especially true if we are looking for expressive dynamic movements, accurate mechanical behavior or highly detailed deformations.

Motivated by these limitations, in this thesis we use data-driven methods to bridge the gap between accurate and interactive simulation models of deformation. Recently, data-driven methods and machine learning techniques have dramatically transformed computer science, showing that a wide variety of problems can be solved with the right data and learning models. The goal of this thesis is to develop useful simulation methods for deformable objects that are fast and accurate, with special focus on efficiently capturing proper dynamic behaviors, external interactions and contact produced deformations. To do so, we investigate the combination of physically-based simulations and data-driven techniques.

1.1 Objectives

As stated in the introduction, our main goal is designing useful models for the efficient simulation of deformable objects and materials. Here we enumerate the most important objectives that deformable models have to fulfill to be applicable in general computer graphics applications:

- **Expressiveness.** We are targeting deformation models that generate realistic and perceptually accurate results. To obtain plausible interactions, adequate mechanical response and proper dynamics are also relevant properties. Importantly, in computer graphics applications the final quality assessment is usually done by our senses, and this allows approximations in the models that might not be acceptable in other engineering fields. Therefore, we look for deformation models whose results are correct enough to feel realistic.

- **Efficiency.** Efficient models can be used in different environments, facilitate exploration, and allow interactive simulation framerates. In particular in computer graphics applications, we need efficient deformation models both in terms of computational resources and time resources. The reason is that multiple applications in this context need to run interactively in virtual environments, either to allow immersive experiences or provide previews of final results. Moreover, these models often run on devices with limited capabilities such as personal computers, mobile devices or headsets, motivating further the need for efficiency.

- **Generalization.** Useful deformation models are stable and robust, enable accurate interactions, and support general contact deformations. Also, to be applicable in unpredictable environments, deformation models need to respond robustly to a wide variety of situations. This is especially relevant with contact interactions between objects, where the generated deformations can be very diverse. Therefore, models have to generalize appropriately to this diversity.

Current methods for deformable object simulations do not meet all of the objectives mentioned above. Precise physically-based methods produce accurate results [SB12; Kim*17] and generalize properly to complex contact deformations [Li*20], however, this is at the expense of efficiency. On the other hand, models focused on interactive applications are efficient, but suffer limited expressiveness and generalization when it comes to capturing realistic material responses [KB18; LLK19] and contact deformations of high spatial frequency [PW89; BJ05; Hah*12; Hah*13; BEH18].

In this situation, we propose the extension of physics-based reduced simulations with data-driven methods to build models with all the desired objectives. Relying on physics-based simulations provides us robustness, generalization, intuitive force-based interactions, and expressive damping-free dynamics. Additionally, using novel data-driven techniques, we can improve the generalization capability of the fast coarse deformation models and preserve most of the accuracy and realism of high-resolution models.

1.2 Challenges

Useful deformable models should be expressive, efficient and generalize to diverse interactions. Next we discuss the main challenges we have faced in order to achieve practical simulation methods with these properties, including some highlights on how we solve them:

- **Complexity of real objects.** Building physical models for real deformable objects is challenging. Real objects have intricate internal structures, with heterogeneous material properties and variable thickness. Living creatures and human bodies are specially complex, with multiple bones, soft-tissue layers, and coupled articulated mechanisms. Therefore, efficiently modeling the behavior of these systems just based on physical assumptions is near impossible. To solve this challenge, we design data-driven techniques to reduce the complexity of the physical models, replacing this complexity with data. Among other things, we optimize deformable material parameters and thickness from real data, design expressive material models, compute elastic deformation in unposed configurations and combine simulated dynamics with quasi-static deformations captured from the real world.

- **Slow physical simulations.** Physical simulations are still the method of choice for computing general-purpose deformations, with accurate physical behavior and vivid dynamics. However, to obtain precise results, high resolution simulations are required, and they tend to be slow. In addition, they may suffer instabilities if extra actions are not taken. To avoid these limitations, we use reduced models. We combine the fast dynamics of subspace simulations with the highly detailed deformations of learning-based methods, presenting models that are efficient, dynamic and naturally respond to external interactions. Our models use practical handle-based subspaces and build efficient nonlinear corrections from data, decoupling the internal and external contact-driven deformations.

- **Overfitting in data-driven methods.** Models based on data tend to overfit if enough deformation examples are not provided. Supporting general interactions and arbitrary contact deformations in data-driven models is especially challenging due to the large number of possible interactions and collider shapes. However, obtaining exhaustive datasets can be very expensive or even impossible, and the resulting models can be large and inefficient. To tackle this problem, we present methods to reduce the required data, with good quality results and still demanding low computational resources. Our approach is based on the generation of deformation examples with an efficient covering of the interaction space, and encoding the deformation models in a suitable representation for the problem at hand, e.g, with contact-centric models, local collider descriptors, continuous representations and sparse deformation maps.

1.3 Contributions

Here we gather the contributions of this thesis, along with the key results for each of them, accompanied by the resulting publications:

- **Accurate soft avatars with hybrid dynamic deformations.** We present an animated avatar model with hybrid soft-tissue deformations generated as a combination of a data-driven statistical model and FEM mechanical simulation. As a key contribution, we define deformation mechanics in the reference space of the statistical model, retaining as much as possible of the accurate data-driven deformations. We also present a custom anisotropic nonlinear material for accurate skin dynamics, automatically optimizing skin thickness and material properties from 4D human captures. Finally, to achieve interactive frame-rates, we augment the avatar model with a careful choice of reduced simulation subspace. These contributions are discussed in Chapter 3 and have resulted in the following publications:

- Cristian Romero, Miguel A. Otaduy, Dan Casas and Jesús Pérez. "Modeling and Estimation of Nonlinear Skin Mechanics for Animated Avatars". *Computer Graphics Forum (Proc. of Eurographics)* (2020)

- Javier Tapia, Cristian Romero, Jesús Pérez and Miguel A. Otaduy. "Parametric Skeletons with Reduced Soft-Tissue Deformations". *Computer Graphics Forum* (2021)

- **Fast reduced simulation with learned deformation corrections.** We design a novel subspace method for the simulation of dynamic deformations, augmenting linear handle-based subspaces with nonlinear learned corrections. This combines the fast dynamics of subspace methods with the highly detailed deformations of learning-based methods. We apply the nonlinear corrections on the local undeformed setting, decoupling internal and external contact-driven corrections for an improved generalization. Our dynamic simulations are efficient, thanks to a simple mapping of corrections to the global setting. Additionally, we design a training pipeline to generate examples that efficiently cover the interaction space. These contributions are discussed in Chapter 4 and have been collected in the following publication:

  - Cristian Romero, Dan Casas, Jesús Pérez, and Miguel A. Otaduy. "Learning Contact Corrections for Handle-Based Subspace Dynamics". *ACM Transactions on Graphics (SIGGRAPH)* (2021)

- **Efficient representation for learning contact deformations.** We introduce an efficient method to machine-learn highly detailed, nonlinear contact deformations for real-time dynamic simulation, using a novel representation suited for contact problems. We depart from previous deformation-learning strategies, modeling contact deformations in a contact-centric manner. Also, we learn a continuous vector field of contact deformations, instead of a discrete approximation. To improve the generalization, we sparsify the mapping between the contact configuration and contact deformations. These contributions are discussed in Chapter 5 and have given rise to the following publication:

- Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. "Contact-Centric Deformation Learning". *ACM Transactions on Graphics (SIGGRAPH)* (2022)

- **Generalized learning for arbitrary contact deformations.** We formulate a learning-based method for the simulation of general rich contact deformations, generalizing the previously introduced deformation model to arbitrary colliders. We improve earlier limitations by designing a neural model that supports general rigid collider shapes. This is achieved thanks to a novel collider descriptor that characterizes local geometry in a region of interest. To validate out approach, we showcase our method on interactive dynamic simulations with animation of rich deformation details and contact.

  These contributions are discussed in Chapter 6 and have culminated in the following publication:

  - Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. "Learning Contact Deformations with General Collider Descriptors". *SIGGRAPH ASIA Conference Proceedings* (2023)

Background 2

Since the pioneering work of Terzopoulos and Witkin [TW88], the computer graphics community has been increasingly interested in the physics-based animation of deformable objects. In the last decades, different methods have been presented to improve the efficiency and expressiveness of the produced deformations, with many existing articles and courses discussing the evolution and current development of deformable object simulation, e.g., [Nea*06; SB12; BS19].

Different rigging techniques have been proposed for the controllable deformation of soft objects, such as linear blend skinning or handle based methods [Jac*14], improving the deformations with pose-dependant correctives [LCF00]. In addition, the simulation of soft articulated bodies has motivated many works that enrich rigging techniques with dynamics [Cap*02; Cap*05] and add two-way coupling between the soft-tissue and the articulated body [Liu*13].

Significant efforts have been devoted to improving the efficiency of deformable simulations [Gal*07; McA*11; LLK19]. With particular relevance, subspace models have been used to approximate the equations of motion by ignoring high-frequency deformations [SB12], using modal analysis [PW89] or principal component analysis of deformation examples [KLM01]. A common issue with subspace simulations is that contact deformations are not resolved in high detail, specially for contact-based interactions. Some works have addressed this limitation, enriching the subspace with a local basis [HZ13] or with a local submesh [Ten*15].

In recent years, data-driven methods have been applied to learn different types of deformation models. They are used for synthesizing high-dimensional deformation detail conditioned by some low-dimensional code [Lop*15; Bai*18], automatically infer compact nonlinear subspaces for dynamic deformations [Ful*19], and learning representations of the full dynamic interaction between a dynamic object and some collider(s) [Hol*19]. Other works have addressed the personalization of the physics-based models by estimating geometric and material properties from observed data [Kad*16; KK19] or optimize material from local measurements [Pai*18].

The following review of related work offers a more complete description of the already mentioned advances, focusing especially on those works related to the main contributions in this thesis.

2.1 Rig-Based Deformers

Rigging techniques are ubiquitous in software animation packages which often include traditional linear deformers and various other nonlinear alternatives. They consist of the transformation of the vertices of a surface mesh by mapping a low-dimensional space of rig parameters.

Nowadays the classic approach to implement deformable shapes is through linear blend skinning (LBS) [MLT89; Jac*14], where an underlying skeleton is used to parameterize the pose of an articulated object, and linear blending of individual bone transformations deforms the shape surface. This technique is also known as skeleton subspace deformation (SSD) [MLT89], and has been extensively used for the animation of skeletal characters and objects, with many extensions and improvements.

Pose-space deformation. To mitigate well-known LBS artifacts and prevent unnatural deformations, it is common to use a pose-space deformation (PSD) [LCF00] method. PSD methods define a reference shape of the skin, apply a pose-space deformation to it, and then transform the deformed skin through rigging. While Lewis et al. [LCF00] computed the PSD through scattered-data interpolation of deformation samples, many other solutions are possible. The data-driven models SMPL [Lop*15] and MANO [RTB17] define the PSD using blend shapes governed by the skeletal pose as well as shape parameters. Such blend shapes are fitted to accurately depict the deformation of parametric bodies and hands respectively under static poses.

Aggregating deformations. Extending beyond skeletal rigs, other animation methods use more diverse definitions of global pose, aggregating local surface deformation for faces [Bic*08] or cloth [Kav*11; ZBO13]. The recent work of Song et al. [SSR20] uses an animation rig as a generalization of pose, and learns both global and local deformation as a function of the rig parameters. If we look at dynamic simulations, two prominent examples of aggregate global-local deformation methods are Eulerian-on-Lagrangian simulation [Fan*13] and multifarious hierarchies [Mal*15]. In this area, the focus is not necessarily on dimensionality reduction. Separation into global and local deformations may have other advantages, such as better modeling of mechanical phenomena or faster solvers.

Deformable human models. The simulation of soft articulated bodies has motivated many works that enrich rigging techniques with simulated dynamics, e.g., [Cap*02; Cap*05]. These works aim at modeling an anatomically-inspired representation of the human body that can be deformed in a physics-based simulation framework to reproduce real-world human body behavior. Some simulation methods approximate the skeletal structure using an articulated rig, and model the soft flesh as a continuum, coupled to the skeleton [Liu*13]. Kim et al. [Kim*17] combined the SMPL model with a soft-tissue layer simulated in full space. Anatomical methods work inside out to represent the musculoskeletal elements of the body [LST09], and then place a soft layer of skin, which provides the final appearance. Some early works focus on specific body parts, including the modeling of head [Käh*02; SNF05], neck [LT06], hands [AHS03], torso [Zor*04], or upper-body [Ter*05; LST09]. Other works have placed emphasis on the properties of the flesh models, including robustness [SGK18] or anisotropy [KDI19]. Others have placed emphasis on the efficiency of the model, as it provides a cost-effective solution for responsive animated characters [KB18; LLK19].

2.2 Subspace Simulations

Subspace simulation methods assume that a high-resolution deformable object is given, and find a low-dimensional subspace that represents accurately the range of deformations of the object. These methods are also known as model order reduction in various fields of engineering and applied mathematics [SB12].

In the context of dynamic simulation, the creation of subspace models allows fast approximation to the equations of motion. These methods replace the degrees of freedom (DoFs) of a chosen discretization with a much smaller set of DoFs expressed in a subspace.

Linear vs nonlinear. Designing an appropriate linear subspace basis has been a research interest since the early works [OHS03]. Modal analysis finds a good subspace based on the mechanical properties of the object [PW89], and principal component analysis does it based on deformation examples [KLM01]. Modal derivatives can improve the basic linear subspace of these two approaches [BJ05].

Other model order reduction approaches use nonlinear subspaces, such as animation rigs [Hah*12; Hah*13] or rotation-strain coordinates [PBH15]. The recent approach of Lan et al. [Lan*20] uses the object's medial axis to find an expressive geometry-motivated subspace. Some authors have looked at complementing artist-defined subspaces with model order reduction to augment them with fast dynamics, such as pose-based subspaces [XB16; Hah*14].

Locality. Traditional subspaces usually have global support, e.g., [KLM01; BJ05], meaning every DoF in the full space depends kinematically on every subspace DoF. This is not necessarily realistic and may cause that novel interactions, which were not originally considered in the design of the subspace, produce unrealistic global artifacts. Some works have alleviated this burden using domain decomposition techniques [KJP02; BZ11; KJ11; WMW15], while others directly address the design of local subspaces.

Local methods are generally based on skinning transformation handles, spatially distributed in regions of deformation interest [Wan*15]. Some examples include the use of interpolated frames [Gil*11; Fau*11], automatic sparse matrix computation [Neu*13], and the more recent work of Brandt et al. [BEH18], which interpolates affine transformations skinned using geodesic radial basis functions.

Subspaces and rigs. Model reduction has been employed in combination with rigging models. The early work by [KJP02] augmented SSD with locally supported eigen-displacement basis functions to achieve quasi-static deformations. Many posterior works further improved this approach for the dynamic simulation of characters [Gal*09; Ten*15] or cloth [Hah*14]. These methods build their basis in an unposed configuration of the object. This might result in inaccurate deformations when skeletal configurations are far from the neutral position. Some methods like [Gal*09] and the more recent [XB16] address this problem by considering several poses for the construction of the basis. While the former constructs a single basis with per-pose PSD corrections, the latter constructs several bases at different poses and interpolates them at simulation time.

Contact enrichment. A common issue with model order reduction is that contact deformations are not resolved in high detail. The variety of contact deformations is too large to be captured by the subspace, and the resulting simulations appear overly smooth.

In this context, a different approach to increase the accuracy of global subspace simulation methods is to locally enrich the simulation model. Harmon and Zorin [HZ13] enrich a linear subspace model with locally supported basis functions precomputed using a Boussinesq contact model. Teng et al. [Ten*15] select submeshes that are simulated with nodal degrees of freedom, while the rest of the object uses a linear subspace representation. Both regions are coupled accurately and efficiently using a condensation method.

Cubature. An additional challenge for simulations with subspace methods is the efficient yet accurate evaluation of the different forces. When the subspace deformations are smooth, this cost can be further alleviated using cubature [AKJ08; Tyc*13]. Cubature methods approximate integrals across the mesh using a small set of representative elements. This usually involves a precomputation where the optimal set is selected to match forces for some training examples. The accurate evaluation of contact forces can be particularly challenging, with some works presenting methods for these specific cases with contacts or self-contacts [TOK14].

2.3 Data-Driven Models

Data-driven methods use data to build the deformation models, avoiding the sometimes complicated or non-existent fundamental-principle approaches. To a large extent, the recent success of machine learning techniques has made data-driven methods gain relevance in the last few years. A great variety of proposals make use of data, combining learning methods with more traditional deformation models such as rig deformations or subspace simulation.

Learning deformations. Data-driven aproaches have been applied to learn different types of deformation models. Prior to the explosion of neural-network methods, de Aguiar et al. [Agu*10] designed a learning-based second-order model of cloth deformation with stability guarantees and Kim et al. [Kim*13] showed how to encode complex dynamics of cloth using motion graphs. More recently, and using artificial neural-networks, many works synthesize rich deformation details conditioned by some low-dimensional code, e.g., linear deformation driving nonlinear deformation [Luo*20], coarse deformation driving numerical coarsening (i.e., high-order shape functions) [Ni*23], upscaling of dynamics to objects of different topology [Zhe*21], skeletal motion driving cloth deformation [SOC22; BME22], upscaling of low-resolution hair simulation [Lyu*22] or convolutional networks for mesh-based deformations [Che*20].

Many works have been proposed to extend pose-space deformation PSD in multiple ways, learning the pose-dependant correctives from data [Lop*15]. Bailey et al. [Bai*18] use multiple neural networks to approximate the rig's nonlinear deformation components. Some works have also learned correctives for modeling specific body parts, such as faces [SSR20] or hands [RTB17]. Garment deformations have also been learned from simulations, using shape correctives [Wan*10; SOC19; PLP20; Pon*17; Ma*20] or even neural features [Zha*21]. Recent works leverage machine-learning methods to learn dynamic corrections as a function of pose and its time evolution. The approach has been applied to bodies [Pon*15] and cloth [SOC19]. As an alternative to learning corrections, the method of Wang et al. [WPP07] learns deformation gradients and then reconstructs the deformation.

Despite the realism of the deformations showcased by these methods, due to their self-driven deformation strategy (i.e., deformations depend only on rig pose or motion), they are unable to model the effects of external interactions.

Learning subspaces. Subspace simulations are an efficient solution when external interactions are required, and data-driven learning techniques are becoming an important alternative for building these efficient and expressive subspace models. Fulton et al. [Ful*19] introduced the use of variational autoencoders to automatically infer compact nonlinear subspaces for dynamic deformations. Recently, Shen et al. [She*21] improved the differentiability of deep autoencoders for their use in deformable simulation with learning-based cubature, and Lee and Carlberg [LC21] showed how to enforce physical conservation laws in the learned subspaces. Very recent work has also placed the attention on training the reduced representation without example data [Sha*23]. Learned reduced models are even used for modeling muscle activation of soft characters [Yan*22].

Learning physics. Some methods try to learn directly the physics of deformable objects. NNWarp [Luo*20] learns the correction between linear and nonlinear materials as a warping function, and thus simplifies the simulation of complex nonlinear materials. Beyond computer graphics, recent efforts on machine learning look at representations of the common invariants and/or processes involved in mechanics. Some of the examples include modeling collisions and deformations using graph representations [Bat*16], producing generic neural-network representations of mechanical evolution using composable objects and their interactions [Cha*17], modeling multi-physics phenomena through learned particle-based models [Li*19], or modeling physical processes by learning invariants and training with measurable functions of these invariants [GDY19].

Learning contact. Other methods learn the contact response in deformable object simulations. Holden et al. [Hol*19] learned the dynamic update of a reduced simulation upon contact. Aigerman et al. [Aig*22] applied a more general methodology for learning Jacobian fields. Learning has also been used for resolving self-collisions, e.g., by computing differentiable collision classifiers as a function of mesh deformation [Tan*22a], or configuration-space distances conditioned by the deformation state of reduced models [Cai*22]. In the case of cloth deformation, particular attention has been placed on solving contact, with self-supervised learning of a latent space free of body collisions [San*21], by adding a repulsive force unit to the network architecture [Tan*22b], or untangling multiple cloth layers through projection operations on implicit representations [San*22]. All these works learn collider-specific models, which do not scale to the combinatorial complexity of object interactions.

Learning human models. Purely data-driven methods aim at finding a model to represent the surface deformations of the human body directly from data. Initial works leverage static 3D scans to build linear models that represent upper torso [ACP02] and full body [All*03] static deformations as a function of body shape. The seminal work of SCAPE [Ang*05] went one step further and learned an articulated human body model parameterized by shape and pose. Later, SMPL [Lop*15] learned pose and shape correctives from a large dataset of 4D scans. Subsequent works have leveraged the learning capabilities of neural networks to extend SMPL to model soft-tissue dynamic deformations [Pon*15; CO18; San*20].

Dynamic deformations caused by soft tissue and muscles have also been attempted from a data-driven perspective. Pioneering works used a marker-based tracking system to capture the trajectory of a few hundred markers to reconstruct [PH06] and model [PH08] soft-tissue deformations. With 3D scanning technologies becoming more accessible and precise, it is nowadays possible to reconstruct full-body sequences exhibiting highly nonrigid deformations [Dou*15; Pon*17; Rob*17; Bog*17]. Such detailed and dense reconstructions of human performances have been very recently leveraged to build data-driven models capable of learning soft-tissue dynamics [Pon*15; CO18], but with no interaction capabilities.

Setting up a physics-based model of a character provides interaction capabilities and a plausible skin behavior. But personalization requires, in addition, the estimation of geometric and material properties from observed data. Some works have addressed the estimation of full-body models [Kad*16], specific models for particularly complex areas such as the face [KK19], or soft-tissue layers from local measurements [Pai*18], and some have even addressed learning of motor control [Nak*18]. Kim et al. [Kim*17] estimate a soft-tissue model to augment a parametric human model.

2.4 Neural Representations

Machine learning techniques, in addition to using data, are characterized by the use of novel model representations. Among these representations, the most widely used are the so-called artificial neural networks, and models that use these neural network as the main mechanism to operate are known as *neural representations* [Xie*21]. When modeling spatial quantities, these neural representations have proved to have interesting features, such as their property of being universal function approximators, their parameterization flexibility, or the easiness of evaluating them for arbitrary input without requiring discretizations.

Neural fields. The high potential of learning field representations has only recently been identified, and it has quickly extended to address many different problems in Computer Vision and Computer Graphics [Che*21]. One prominent example is Neural Radiance Fields (NeRF) [Mil*20], which learn to synthesize novel views of complex scenes by optimizing a continuous volumetric scene function using a sparse set of input views. Other popular examples are methods that learn to reconstruct 3D shapes from images by conditioning an implicit representation on local features extracted from images [Sai*19; Sai*20].

Neural shapes. Recent trends learn implicit neural representations to encode 3D shapes. Initial works learn to approximate the surface of 3D meshes by predicting a binary occupancy of arbitrary 3D points [Mes*19; CZ19]. Since fully-connected neural networks are used, the learned representation is continuous, memory-efficient, and easily differentiable, which brings many benefits in simulation, computer vision, and geometry processing frameworks. For example, these representations enable differentiable inside/outside queries, which are tricky to implement with traditional representations such as polygonal meshes. Follow-up research [Par*19; CZ19; AL20] demonstrated that neural networks are also capable of learning distance to surface, which is also a fundamental building block for many methods in Computer Graphics. Such learning-based encoding is often referred to as *implicit neural representations* or *neural distance fields* [CMP20].

Beyond rigid surfaces, the advantages that learned implicit representations bring have been leveraged to model more complex objects, such as articulated shapes. Deng et al. [Den*20] model an articulated human body using a piecewise implicit representation. Subsequent works learn fully-parametric body models [AXS21; Mih*21], hands [Kar*21], hand-and-object interactions [Kar*20], garments [Cor*21], and personalized dressed humans [Sai*21; Tiw*21].

To train these representations, existing methods often require direct 3D supervision in form of a known or pre-computed implicit representation of the target shape [Par*19; CZ19]. Interestingly, more recent methods are able to train directly from raw point clouds (i.e., without supervision at the zero level set) [Gro*20; AL20] or open surfaces [CMP20], something that is not possible with traditional representations of signed distance fields (SDF).

Beyond surfaces. Other methods explore more general uses of learned fields, and beyond using them to approximate implicit functions, leverage them to expand surface properties to 3D points. Subsequent learning-based methods expand surface properties, such as skinning weights, outside the body surface. This strategy has been used for registering 3D scans to meshes [Bha*20], and for articulating raw scans of dressed humans [Hua*20]. Santesteban et al. [San*21] go one step further and learn neural fields to diffuse pose-and-shape surface correctives to $\mathbb{R}^3$. Learned fields are used for projecting garments to a canonical shape, which enables highly-efficient handling of body-garment collisions.

Neural shape descriptors. Neural representations for unstructured point clouds, starting with PointNet [Qi*17a], have opened the possibility to encode rich latent shape information. In a nutshell, they aggregate individual point features into a global signature. PointNet++ [Qi*17b] extends the original PointNet with a hierarchical structure, allowing the representation of local features at different scales. PCPNet [Gue*18] learns local features robust to common point imperfections (e.g., varying noise level, sampling density, level of detail, missing data), by enforcing perturbations in the training data. Point2Sequence [Liu*19] aggregates the information of different local regions thanks to an attention mechanism. EdgeConv [Wan*19] arranges dynamic graphs on point clouds to enable more powerful operations. Such shape descriptors are mostly used for problems such as object detection, shape classification, or part segmentation.

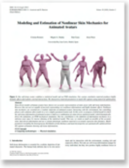
Other works use SDF data to construct shape descriptors, such as probabilistic directed distance fields [Aum*22] or neural omnidirectional distance fields [Hou*22]. However, these representations are more complex than the actual SDF and are designed for ray queries. Deep Local Shapes [Cha*20] reconstruct large surfaces from continuous local deep SDFs. Neural Descriptor Fields [Sim*22] encode object manipulation conditioned by pose, and are used for inverse modeling of contact manipulation tasks. Interestingly, Chun et al. [Chu*23] recently generalized Neural Descriptor Fields to unseen objects by utilizing local descriptors.

As a related problem, several works have studied the neural computation of vector quantities on surfaces, which poses a challenge due to the choice of reference frame on the tangent plane. Multi-Directional Geodesic CNNs [PO18] address the challenge by computing quantities on multiple frames. Harmonic Surface Networks [WEH20], on the other hand, extend harmonic nets to surfaces and achieve rotation-equivariance of vector quantities.

3 Modeling Soft Avatars with Hybrid Deformations

Data-driven models of human bodies have shown very accurate representations of static poses with soft-tissue deformations. However they are not yet capable of precisely representing very nonlinear deformations and highly dynamic effects. Nonlinear skin mechanics are essential for a realistic depiction of animated avatars interacting with the environment, but controlling physics-only solutions often results in a very complex parameterization task.

In this chapter, we propose a hybrid model in which the soft-tissue deformation of animated avatars is built as a combination of a data-driven statistical model, which kinematically drives the animation, and an FEM mechanical simulation. Our key contribution is the definition of deformation mechanics in a reference pose space. This way, we retain as much as possible of the accurate static data-driven deformation and use a custom anisotropic nonlinear material to accurately represent skin dynamics. Model parameters including the heterogeneous distribution of skin thickness and material properties are automatically optimized from 4D captures of humans showing soft-tissue deformations. To achieve interactive frame-rates, we also augment our avatar model with simulated reduced deformations. We succeed to do so in a highly efficient manner, thanks to a careful choice of reduced model for the subspace deformation. The work on model reduction has been published in a paper whose first author is Javier Tapia, but this thesis has contributed extensively in the design of the deformation model, the cubature approach, and the experiments; therefore the full work is discussed here for completeness. Overall, the contributions presented in this chapter have led to the following publications:



Modeling and Estimation of Nonlinear Skin Mechanics for Animated Avatars

- Cristian Romero, Miguel A. Otaduy, Dan Casas and Jesús Pérez. “Modeling and Estimation of Nonlinear Skin Mechanics for Animated Avatars”.
Computer Graphics Forum (Proc. of Eurographics) (2020)

Parametric Skeletons with Reduced Soft-Tissue Deformations

- Javier Tapia, Cristian Romero, Jesús Pérez and Miguel A. Otaduy. “Parametric Skeletons with Reduced Soft-Tissue Deformations”.
Computer Graphics Forum (2021)

3.1 Introduction

Soft-tissue deformation is essential for a realistic depiction of animated characters. The human body deforms due to its own movement and its interaction with the environment, creating rich and expressive effects. Not only are soft-tissue deformations unique for every individual, but they also produce highly nonlinear forces in response to the interaction with surrounding objects. A jiggling belly, the quick transition of skin from soft to stiff when we pull from it, or the bulging induced by tight apparel, are familiar examples to all of us. Finding accurate and inexpensive methods for the animation of soft-tissue characters in highly dynamic and contact-intensive scenarios has been a long-term goal in the Computer Graphics community, with numerous applications in VFX, video games or garment design, among others. In this chapter, we present a model to animate personalized characters with rich soft-tissue dynamics produced by both skeletal motion and external interactions.

There are two distinct approaches to produce expressive animated characters: physics-based and data-driven. Physics-based solutions model the mechanical response of flesh, and simulate deformations by solving the equations of elasticity [Liu*13; Kad*16]. Their advantage is the ability to respond in a plausible way to arbitrary interactions; they are fully generative by construction. Their challenge, on the other hand, is to design materials that mimic the properties of skin, and to parameterize them to match the behavior of each individual. Data-driven solutions learn nonlinear mappings from skeletal motion to soft-tissue deformation [Lop*15; Pon*15; CO18]. Their advantage is the ability to reproduce observed data with high accuracy, thanks to high-dimensional parameterizations and rich nonlinear building blocks, not constrained by physics laws. Their challenge, on the other hand, is the lack of response to external interactions or, more generally, the need for large training data and the difficulty to generalize to unseen situations.

We propose a model for avatar animation that combines the advantages of physics-based and data-driven approaches. The model responds to external interactions, yet it reproduces accurately observed deformations. The high potential of the combined approach is evidenced by the work of Kim et al. [Kim*17]. However, our deformation model enjoys major contributions that enhance notably the response to external interactions and the ability to reproduce observed deformations (See Fig. 3.1).

We design a deformation model that reproduces with very high accuracy static deformations, matches well dynamic deformations, and exhibits plausible and robust response to unseen interactions. We build the deformation model by extending the SMPL parametric human model [Lop*15] within the volume of the body, and adding a parametric smooth soft-tissue

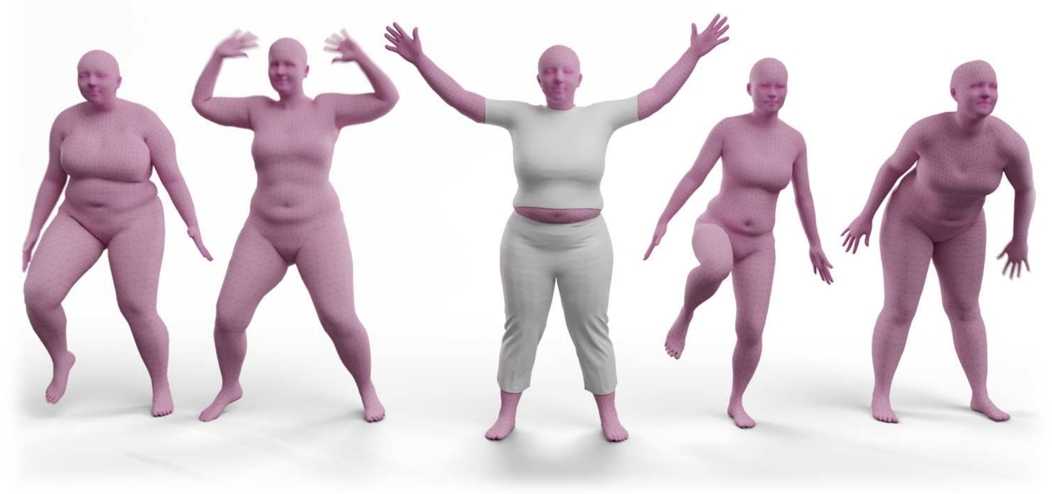


Figure 3.1: Our soft-tissue avatar combines a statistical model and an FEM simulation. Our custom constitutive material produces highly dynamic effects and realistic external interactions. We characterize material parameters to match 4D captures using numerical optimization.

layer in neutral shape and pose. We achieve effective personalization of the soft-tissue layer through shape-and-pose-dependent transformation plus adjustment of its thickness parameters.

The key to match static deformations with very high accuracy is a formulation of mechanics in neutral pose. With this formulation, soft-tissue mechanics ignore static deformations already considered by the parametric model, and focus instead on accurately capturing highly dynamic deformations. We introduce a sound and simple derivation of this mechanical formulation, and we show how animated characters converge exactly to the output of the parametric model when they reach a static pose.

For plausible and robust interaction, we design a custom nonlinear skin material. Furthermore, by observing human performance data, we have identified strong anisotropy in the motion of skin. Therefore, we augment the custom material with anisotropy, and define anisotropic error metrics for the estimation of material parameters from observed motions.

Following the definition of the deformation model and the skin material, we personalize them by estimating soft-tissue thickness and heterogeneous material properties from 4D human performance data. We execute the estimation as a numerical optimization, and to optimize efficiently we design acceleration strategies for gradient-based optimization methods. As a result of this optimization, our method accurately reproduces observed deformations. While our model does not intend to match real material stress, it shows realistic behaviors under external interactions.

Finally, to achieve interactive simulation frame rates, we present a model of reduced kinematics that combines our data-driven skeletal model and a handle-based subspace simulation. We leverage recent developments on handle-based subspace methods [Wan*15] that provide intuitive ways to define expressive subspaces, simply by selecting handles on the target object, without the need for deformation examples. For the successful conservation of accurate data-driven deformations, we build the subspace basis in the unposed reference shape. The resulting model produces effective skeletal dynamics and data-driven static deformation, enriched with subspace soft-tissue deformations.

The current chapter is structured as follows. We start describing our volumetric parametric human model (Section 3.2), we introduce our deformation model (Section 3.3), and then we discuss our method for model estimation (Section 3.4). Finally, we construct our reduced soft skeletal model to accelerate simulations (Section 3.6) and present the formulation of forces and dynamics (Section 3.7).

3.2 Construction of a Soft-Tissue Avatar

Our method creates soft-tissue deformations as a combination of both data-driven and physics-based components. Our deformable avatars are composed of a soft-tissue layer with varying thickness throughout the body that is simulated using a nonlinear finite element method (FEM). The data-driven model is used to define the overall shape of the avatar, to kinematically drive the inner vertices of the soft-tissue layer and to formulate deformation mechanics in an unposed reference shape. This provides us with a personalized simulation model that retains as much as possible of the data-driven deformation for static poses and only uses mechanics to precisely model the response to external interactions as well as highly dynamic effects. In this section, we first briefly review SMPL, a statistical model, which is capable of creating static deformations of the body surface for some input shape and pose parameters. Then, we first describe how we create a volumetric mesh of the soft-tissue layer with varying thickness, and adapt SMPL to also account for volumetric deformations. This pipeline results in a parametric soft-tissue avatar ready to be used in simulation.

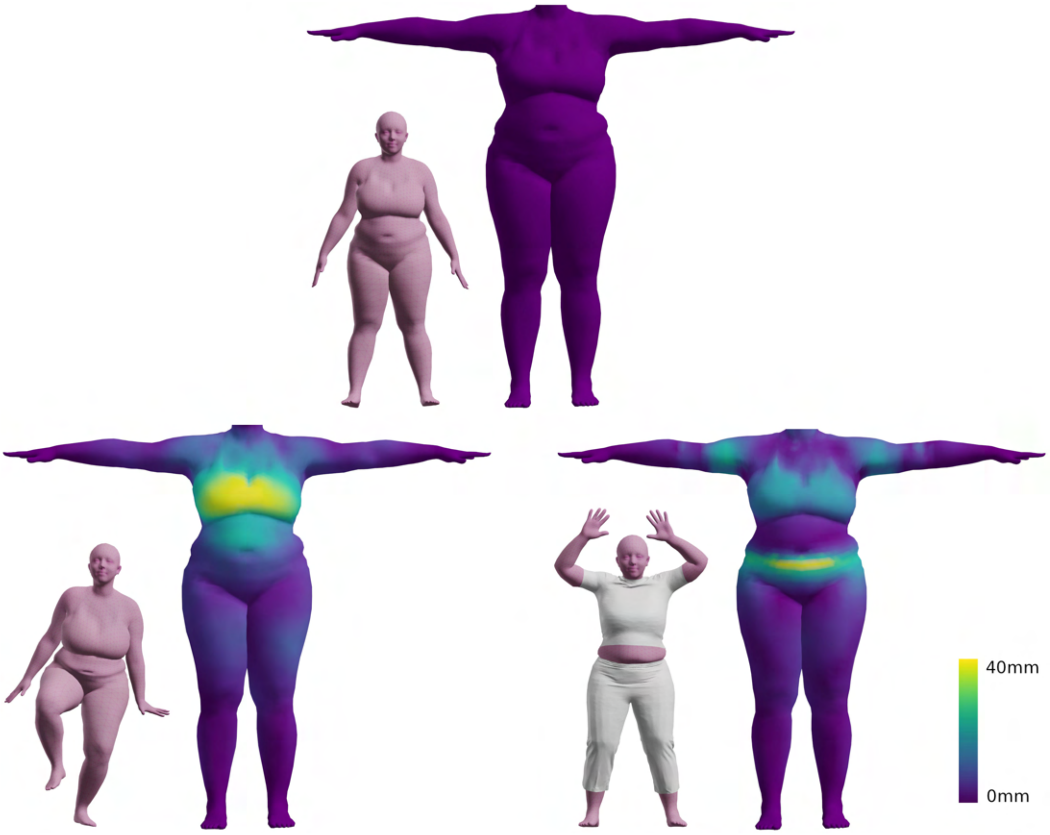


Figure 3.2: These are three examples of posed avatars (inset) subject to various soft-tissue deformations expressed in unposed space through a color map. For a static pose (top), the unposed shape is undeformed and identical to the reference shape. For highly dynamic (left) or external interaction (right) scenarios, the unposed shape shows high-frequency local deformations.

3.2.1 Body Surface Model

Our model relies on SMPL [Lop*15], a statistical model of body surface static deformations that modifies a rigged template mesh $\mathcal{T}$, with $N = 6890$ vertices and $K = 24$ skeletal joints. Vertex positions are adapted depending on two sets of parameters: i) the *pose*, $\theta, |\theta| = 75$, encoding the translation and rotations of the skeletal chain; and ii) the *shape*, $\beta, |\beta| = 10$, representing a series of identity-dependent features that model aspects such as height, slenderness or muscularity. Given these parameters, SMPL produces shape and pose dependent surface deformations following the expression:

$$\begin{aligned} S(\beta,\theta) &= W(\bar{S}, J(\beta), \theta, \mathbf{W}), \\ \bar{S}(\beta,\theta) &= \mathcal{T} + S_s(\beta) + S_p(\theta). \end{aligned} \tag{3.1}$$

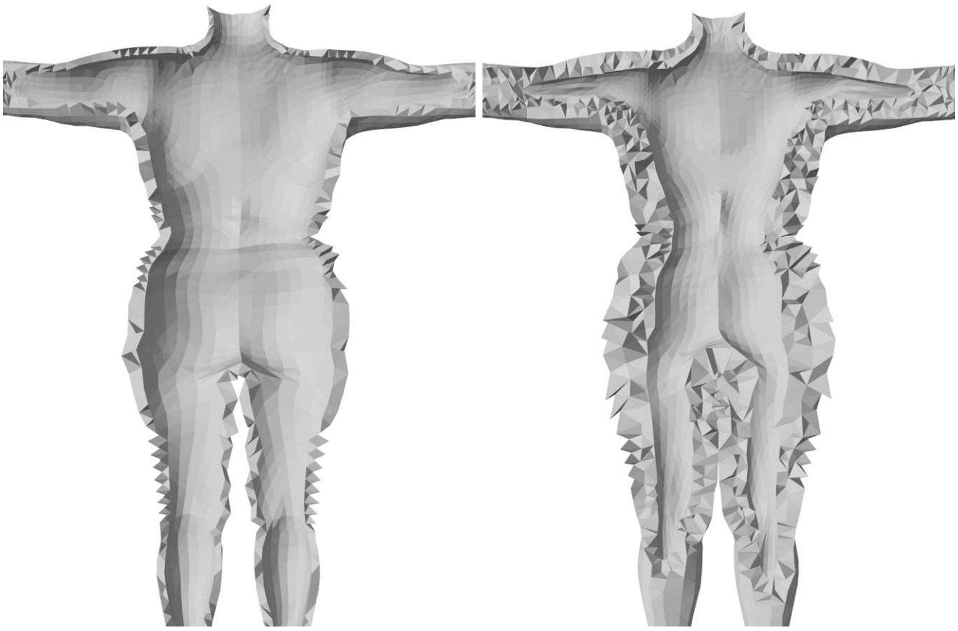


Figure 3.3: We generate a volumetric discretization of the soft-tissue layer with smoothly vayring thickness throughout the body. This avoids the negative effect that an irregular discretization of the inner surface might have on the outer surface deformation.

Here, $W(\bar{S}, J(\beta), \theta, \mathbf{W})$ is a linear blend skinning function [MLT89] that takes as input the surface vertices in a reference T-pose $\bar{S}(\beta, \theta) \in \mathbb{R}^{3N}$, the joint locations $J(\beta) \in \mathbb{R}^{3K}$, the pose parameters $|\theta| = 75$ and some blend weights $\mathbf{W} \in \mathbb{R}^{3NxK}$. To create the reference T-pose $\bar{S}$, SMPL modifies the template in an additive way through per-vertex 3D offset functions called *blend shapes*. While shape blend shapes, $S_s(\beta)$, model changes due to identity body features, pose blend shapes, $S_p(\theta)$, deform the template to compensate for skinning artifacts and features such as muscle bulging. Both blend shapes and joint location functions are learnt from 3D human captures in different poses. We refer to the original work [Lop*15] for a more detailed explanation.

3.2.2 Soft-Tissue Mesh

Simulating the soft-tissue layer requires creating a volume mesh with varying thickness throughout the body. One straightforward solution used in [Kim*17] would be to compute a volumetric discretization of the template mesh, $\mathcal{T}$, and then select those elements within a given distance of the surface. However, this leads to irregular and jaggy discretizations of the inner surface. This might potentially affect the robustness of the simulation against element inversion and the smoothness of the outer surface deformation due to the highly nonlinear response to external interactions. For this reason, we propose an alternative method that first creates smooth outer and inner boundaries of the flesh layer and then generates the volumetric discretization using a standard meshing software.

The outer surface of the soft-tissue layer corresponds to the template mesh $\mathcal{T}$. To define the inner surface $\mathcal{B}$, each vertex of the template mesh $\mathbf{t}_i \in \mathcal{T}, i = 1, \ldots, N$ is initially projected to the closest bone of the kinematic chain leading $\hat{\mathbf{t}}_i$. This mapping often results on failure cases where neighboring vertices in the surface correspond to points on the skeleton which are far from each other. To correct this issue, we refine this initial solution by iteratively applying a Laplacian smoothing of vertex projections, which slide along the skeleton until convergence. This yields a point distribution where each pair of neighboring surface vertices also have neighboring skeleton projections. In practice, this allows us to effectively shrink the body surface moving each vertex along the projection direction $\mathbf{u}_i = (\hat{\mathbf{t}}_i - \mathbf{t}_i)/|\hat{\mathbf{t}}_i - \mathbf{t}_i|$, while preventing face inversion. Consequently, given some thickness distribution defined at each vertex of the template mesh, $\mathbf{h} = \{h_1, \ldots, h_N\}$, the inner surface $\mathcal{B}(\mathbf{h})$ can be easily generated by setting each vertex $\mathbf{b}_i \in \mathcal{B}, i = 1, \ldots, N$ as $\mathbf{b}_i = \mathbf{t}_i + \mathbf{u}_i h_i$. Here, thickness values are kept in a feasible range $h_i \in [h_i^{min}, h_i^{max}]$. The lower bound h_i^{min} is equal for all surface vertices and corresponds to a minimum soft-tissue layer thickness. The upper bound h_i^{max} varies for each of the vertices and can be computed as $h_i^{max} = |\hat{\mathbf{t}}_i - \mathbf{t}_i| - b$, for some common minimum bone thickness b.

Once both the outer and inner meshes are created, we generate the volumetric mesh using the Tetgen software package [Si15], making sure we retain the original topology for the boundaries. This leads to a parameterized volumetric mesh $\mathcal{V}(\mathbf{h})$ with M vertices and D elements. Notice that none of the soft-tissue deformations affects extremities (i.e., hands, feet and head). In practice, we remove those parts from the volumetric mesh and enforce thickness to slowly decay to zero at the boundaries. As it can be seen in Fig. 3.3, our way of generating the volumetric mesh produces smooth discretizations that are better suited for simulation.

3.2.3 Volumetric Statistical Model

Our deformation method relies on the statistical model to constrain the kinematics of the inner surface of the soft-tissue layer, as well as to formulate deformation mechanics in an unposed reference state. However, the model defined in (3.1) only handles surfaces and thus we must adapt the SMPL method to account for volumetric deformations. This is done by interpolating blend shape functions and blending weights from the surface points to the volume mesh through Laplacian interpolation. More precisely, as the outer boundary of $\mathcal{V}$ is conformal with the template mesh $\mathcal{T}$, we simply contrain the values at the outer vertices and compute a Laplacian interpolation matrix $\mathbf{L} \in \mathbb{R}^{3M \times 3N}$ using the cotangent approximation [Sor05]. Thus, our volumetric SMPL function simply results

$$\begin{aligned} V(\beta, \theta) &= Z(\bar{V}, \mathcal{J}(\beta), \theta, \mathbf{Z}), \\ \bar{V}(\beta, \theta) &= \mathcal{V}(\mathbf{h}) + V_s(\beta) + V_p(\theta), \end{aligned} \tag{3.2}$$

where $V_s = \mathbf{L} \cdot S_s$, $V_p = \mathbf{L} \cdot S_p$, $\mathbf{Z} = \mathbf{L} \cdot \mathbf{W}$. We notice that this approximation is not exactly equal to effectively applying Laplacian interpolation to blend shape functions as done in [Kim*17]. However, we have not detected any significant difference in the performance of the method.

3.2.4 Parametric Soft-Tissue Avatar

The statistical method described in the previous section provides us with a parametric model of a soft-tissue animated avatar ready to be simulated. The model has a personalized shape β and varying skin thickness $\mathbf{h}$ and can be statically deformed given some time-dependent pose θ. The separation between fixed and time-varying properties defines two clearly different parametric shapes that are used in combination with the FEM simulation:

- The *reference shape* $\bar{\mathbf{x}}(\mathbf{h}, \beta) = \mathcal{V}(\mathbf{h}) + V_s(\beta), \bar{\mathbf{x}} \in \mathbb{R}^{3M}$, which depends only on character-based properties that are fixed through the animation and define a natural rest configuration.

- The *skinned shape* $\mathbf{p}(\bar{\mathbf{x}}, \theta) = Z(\bar{\mathbf{x}} + V_p(\theta), J(\beta), \theta, \mathbf{Z}), \mathbf{p} \in \mathbb{R}^{3M}$, which conversely depends on the time-changing pose and provides an accurate approximation to soft-tissue static deformation. Note this function can be expressed in a compact way as an affine skinning transformation $\mathbf{P}(\theta)$ of the reference shape $\mathbf{p} = \mathbf{P}(\theta) \cdot \bar{\mathbf{x}}$.

The first expression defines the rest shape of the FEM simulation. In Section 3.3, we leverage the second expression to drive the kinematics of the inner surface vertices of the soft-tissue layer, as well as formulating mechanics in an unposed deformed shape.

3.3 Soft-Tissue Avatar Mechanics

The volumetric SMPL method introduced in the previous section provides us with a tool to parametrically deform a volumetric soft-tissue model given some thickness $\mathbf{h}$, shape

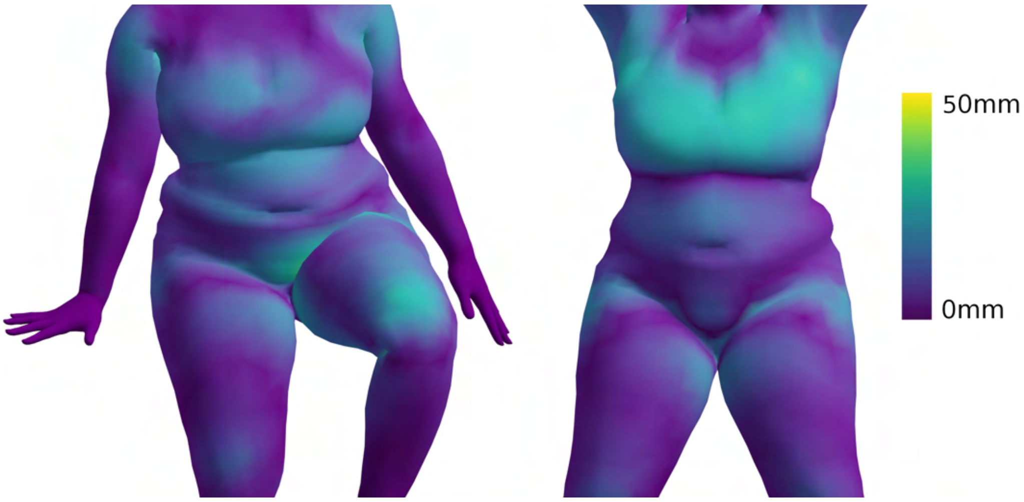


Figure 3.4: This picture shows the resulting deformation for two static poses when defining the deformation gradient in world space as in [Kim*17]. The color map highlights where the result differs from the data-driven static deformation of SMPL.

β, and pose θ parameters. We aim to build a hybrid data-driven and physically based model such that it retains as much as possible of the data-driven pose-dependent static deformation, while taking advantage of the capabilities of FEM simulation to capture nonlinear skin dynamics. Overall, this is done by using the volumetric SMPL to impose kinematic boundary conditions on the internal surface of the soft-tissue layer, while letting the rest of the vertices deform freely. However, two design decisions are key for an accurate depiction of the skin behavior. First, we formulate deformation mechanics in an unposed reference space to ignore all pose-dependent deformations already captured by the SMPL model. And second, we design a custom constitutive material that is capable of representing two essential features of the skin: anisotropy and nonlinearity.

In this section, we first explain how we formulate deformation mechanics and discuss its advantages with respect to other alternatives. Then, we describe in detail our custom material. Finally, we formalize the dynamics equation that is solved to simulate the deformation of the soft-tissue layer.

3.3.1 Skin Mechanics Formulation

Continuum mechanics define the elastic response to deformation in terms of a strain energy density $\Psi(\mathbf{F})$ that depends on the deformation gradient $\mathbf{F}$, a tensor field defined at each point of the volume as the partial derivative of the deformed position w.r.t. the rest configuration. The total elastic potential is then computed as the volumetric integral $U = \int \Psi(\mathbf{F}) \cdot dV$.

FEM provides an interpolation framework to define $\mathbf{F}$ at each element of the volumetric discretization and approximate the total potential integral as a summation [ITF06].

One key concept that is going to enable our deformation mechanics formulation is that of skinning transformation $\mathbf{P}(\theta)$. In the previous section, we introduced the transformation to map the reference shape $\bar{\mathbf{x}}$ to the skinned shape $\mathbf{p} = \mathbf{P}(\theta) \cdot \bar{\mathbf{x}}$. However, this transformation can be used to map arbitrary deformed configurations. For an arbitrary simulated configuration, let's denote the mesh in world space $\mathbf{x} \in \mathbb{R}^{3M}$ as *deformed shape*, and the mesh in unposed space $\mathbf{u} \in \mathbb{R}^{3M}$ as *unposed deformed shape*. These two shapes are related through the skinning transformation $\mathbf{x} = \mathbf{P}(\theta) \cdot \mathbf{u}$, and through the inverse skinning trasnformation $\mathbf{u} = \mathbf{P}(\theta)^{-1} \cdot \mathbf{x}$. Thanks to this transformations, we have the option of expressing the simulated configurations in two different spaces or coordinates systems, being able to transform from one to the other when necessary.

The *unposed deformed shape* $\mathbf{u}$ has some desirable properties to measure deformation. In absence of acceleration or external forces, it is identical to the reference shape $\mathbf{u} \equiv \bar{\mathbf{x}}$. In other cases, the resulting shape differs from the reference only in local high-frequency deformations due to external interactions and dynamic effects. This opens the possibility to formulate soft-tissue mechanics so that only deformations different from the static pose defined by the skinning transformation produce an actual elastic response.

Following this observation, we propose an alternative deformation gradient definition as the derivative of the unposed deformed shape w.r.t. the rest configuration, $\mathbf{F} = \nabla_{\bar{\mathbf{x}}}\mathbf{u}$. This way, our local strain metric ignores large global deformations that are accounted for by the pose-dependent SMPL model and focuses solely on high-frequency local deformations that arise from inertial effects and external interactions. With this formulation, it is not necessary to consider gravity forces because their effect is already included in the statistical model. Inertia effects appear naturally, since the method does not modify the definition of kinetic energy. Finally, note that our deformation gradient $\mathbf{F}$ measures deviation w.r.t. the unposed reference state and this might affect the expected stress response for a given strain. However, we do not aim to model anatomically accurate skin and so strain magnitudes are not exact anyway. Instead, we seek to produce realistic behaviors through a simplified model and, to this end, we optimize material parameters to best fit captured data.

Alternatively, [Kim*17] use the standard definition of the deformation gradient, i.e., the derivative of deformed shape expressed in world coordinates $\mathbf{F} = \nabla_{\bar{\mathbf{x}}}\mathbf{x}$. However, this solution does not take advantage of the deformation data provided by the statistical model and does not preserve pose-dependent static deformations in absence of external interactions. As it can be seen in Fig. 3.4, deformations resulting from the definition of the deformation gradient in world space significantly differ from SMPL parametric shapes even for static poses. There is an additional way of defining a deformation metric such that static deformations are preserved, the so-called *rest-state retargetting* [KDI19]. In that case, the deformation gradient is computed in world coordinates but the reference shape is incrementally updated to match the static deformation produced by the statistical model, i.e., $\mathbf{F} = \nabla_{\mathbf{p}} \mathbf{x}$, with $\mathbf{p} = \mathbf{P}(\theta) \cdot \bar{\mathbf{x}}$. However, this alternative comes with the burden of having to compute all rest-state-dependent magnitudes that remain constant in other cases. Plus, generating the volumetric mesh might produce meshing errors for some extreme poses and affect the solution of the dynamics problem.

Notice, that even if our deformation gradient is computed in unposed space, we can represent the state of the simulation both in world space or unposed space. For the full model introduced in this section we chose to encode it in world space. In this way, we don't need to modify the inertial terms and the simulation equations are solved as we would do for any other deformable model. However, as explained later in Section 3.7.1, simulating in world space complicates the construction of a static pose-preserving subspace and the computation of the required derivatives. To circumvent this problem, for the reduced model case, we simulate deformations directly in unposed space.

Having defined our local measure of strain based on the unposed reference shape $\mathbf{F} = \nabla_{\bar{\mathbf{x}}}(\mathbf{P}^{-1}\mathbf{x})$, the response of the material to deformation is fully goberned by the properties of the elastic energy density $\Psi(\mathbf{F})$. In the next section, we will describe in detail how we define such elastic potential to capture deformation properties that are key for an accurate depiction of skin deformation.

3.3.2 A Nonlinear Skin Material

We aim to design an elastic energy function that accurately captures the complexity of skin dynamics. More precisely, we want to support two deformation properties: *anisotropy* and *nonlinearity*. Skin is a complex structure composed of several layers of heterogeneous materials. It has been shown experimentally that this structure results in a highly nonlinear anisotropic response to tangential and normal components of the deformation. After extensive experimentation, we designed a custom elastic potential to support each of these properties as mixture of an orthotropic *StVK* material [LB14], combined with a *Fung* type

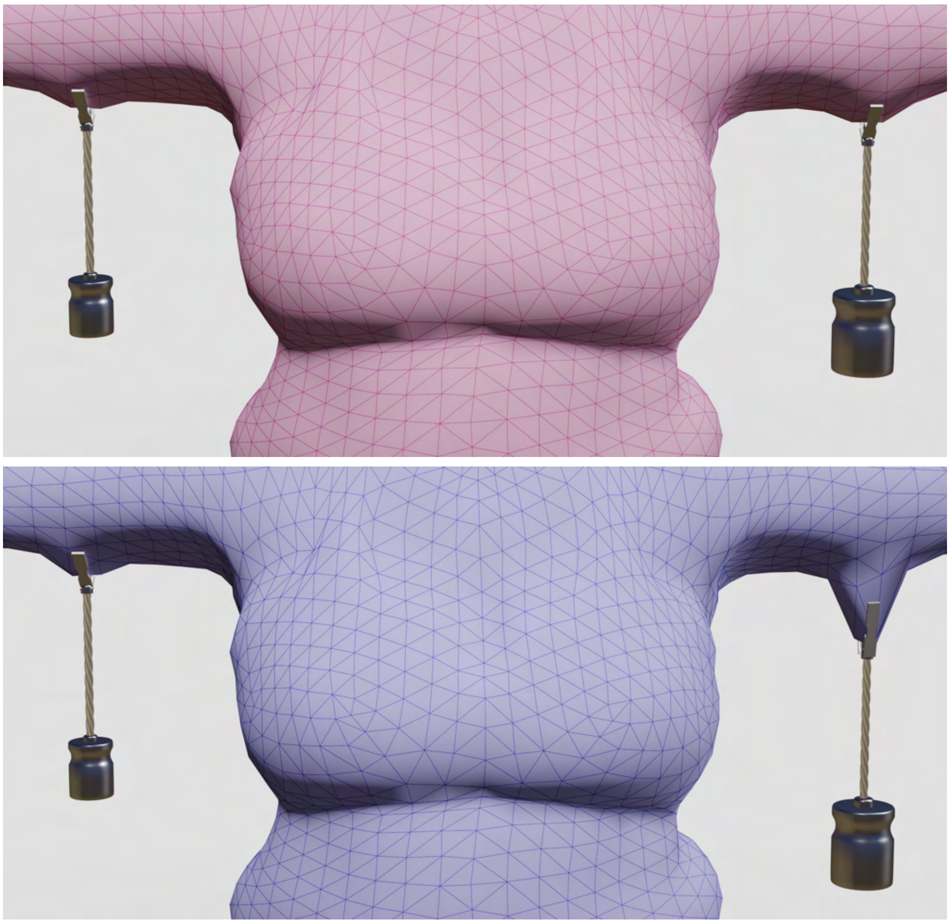


Figure 3.5: This picture shows the deformation of our material (top) in comparison with the nonlinear flesh material defined in [SGK18] (bottom), for two different weights under gravity: 0.5Kg (left) and 2.0Kg (right). It can be clearly seen that our material provides a higher nonlinear response.

exponential saturation to increase the nonlinear response [SGK18]. The elastic energy density $\Psi(\mathbf{F})$ results

$$\begin{aligned} \Psi(\mathbf{F}) =& \mu(\exp(\eta \cdot \sigma) - 1)/\eta + \frac{\lambda}{2}(J-1)^2, \\ \sigma =& \sum_{i=1}^{3}\sum_{j=1}^{3} \tau_{ij} \cdot {E_{ij}}^2, \quad J = \det(\mathbf{F}). \end{aligned} \tag{3.3}$$

Here, E_{ij} are the coefficients of the Green strain tensor $\bar{\mathbf{E}}$ expressed in an orthonormal basis $\mathbf{B}$ that aligns the axes in a meaningful direction for modeling skin anisotropic response, $\bar{\mathbf{E}} = \mathbf{B}^T \cdot \mathbf{E} \cdot \mathbf{B}$, with $\mathbf{E}$ the Green strain tensor evaluated at unposed deformed shape. The first term of (3.3) models material response to stretch and compression and is controlled through the first Lamè parameter $\mu \in [0, \inf)$, together with a saturation parameter $\eta \in (0, \inf)$ that regulates the nonlinearity of the elastic response. Anisotropy is achieved by individually weighting the contribution of each quadratic component in σ using weights $\tau_{ij} \in [0, 1]$. The second term depends on the relative volume change $J = \det(\mathbf{F})$ and modulates material respose to incompressibility through the second Lamè parameter λ.

Fig. 3.5 shows a comparison between our material and the nonlinear flesh material defined in [SGK18]. In both cases, material parameters have been estimated from data to best fit target 4D human captures. The saturation component in (3.3) produces a highly nonlinear response preventing very large deformation from happening even under big external loads. To prove the capabilities of our method to handle external interactions, we have integrated the *ArcSim* cloth simulator [NSO12] in our pipeline as well as incorporated collision handling. Fig. 3.6 shows realistic soft-tissue deformations as a result of the avatar wearing tight cloths. In addition, handling self-contacts improves the default SMPL solution even in the absence of other external interactions.

3.3.3 Mechanics Parameterization

Our custom constitutive material requires the characterization of Lamé parameters and several weigths, $\{\mu, \lambda, \tau_{ij}\}$. In practice, we estimate the elastic modulus for the normal and tangential directions $Y_N, Y_T \in [0, \inf)$, the Poisson's ratio $\nu \in [0, 0.5)$ and the saturation parameter $\eta \in (0, \inf)$. We compute Lamè parameters as

$$\mu = \frac{Y_{max}}{2(1+\nu)}, \quad \lambda = \frac{Y_{max}\nu}{(1+\nu)(1-2\nu)}, \tag{3.4}$$

where $Y_{max} = \max(Y_N, Y_T)$. In a similar way as done in [LB14], we set the coefficients corresponding to the main stretch directions relative to elastic moduli, $\tau_{00} = Y_N / Y_{max}$ and $\tau_{11} = \tau_{22} = Y_T / Y_{max}$, and automatically compute the value of shear direction coefficients as $\tau_{ij} = \max(\tau_{ii}, \tau_{jj}), i \neq j$ to ensure a reasonable behavior.

Overall, the mechanical behavior of our animated avatar is completely determined by the thickness of the soft-tissue layer at each point of the surface mesh $\mathbf{h}$; and the material parameters $\mathbf{m}_i = \{Y_N, Y_T, \nu, \eta\}$ and anisotropy basis $\mathbf{B}_i$ at each element of the volumetric mesh. In practice, we fix Poisson's ratio $\nu = 0.2$ and the saturation parameter $\eta = 4$ to constant values throughout the volume that were experimentally found to produce reasonable volume preservation and nonlinear effects. Anisotropy basis $\mathbf{B}_i$ are defined in the surface mesh to be aligned with the surface normals of the reference shape $\bar{\mathbf{x}}$ defined in Section 3.2.4, and interpolated through the volume using Laplacian interpolation.

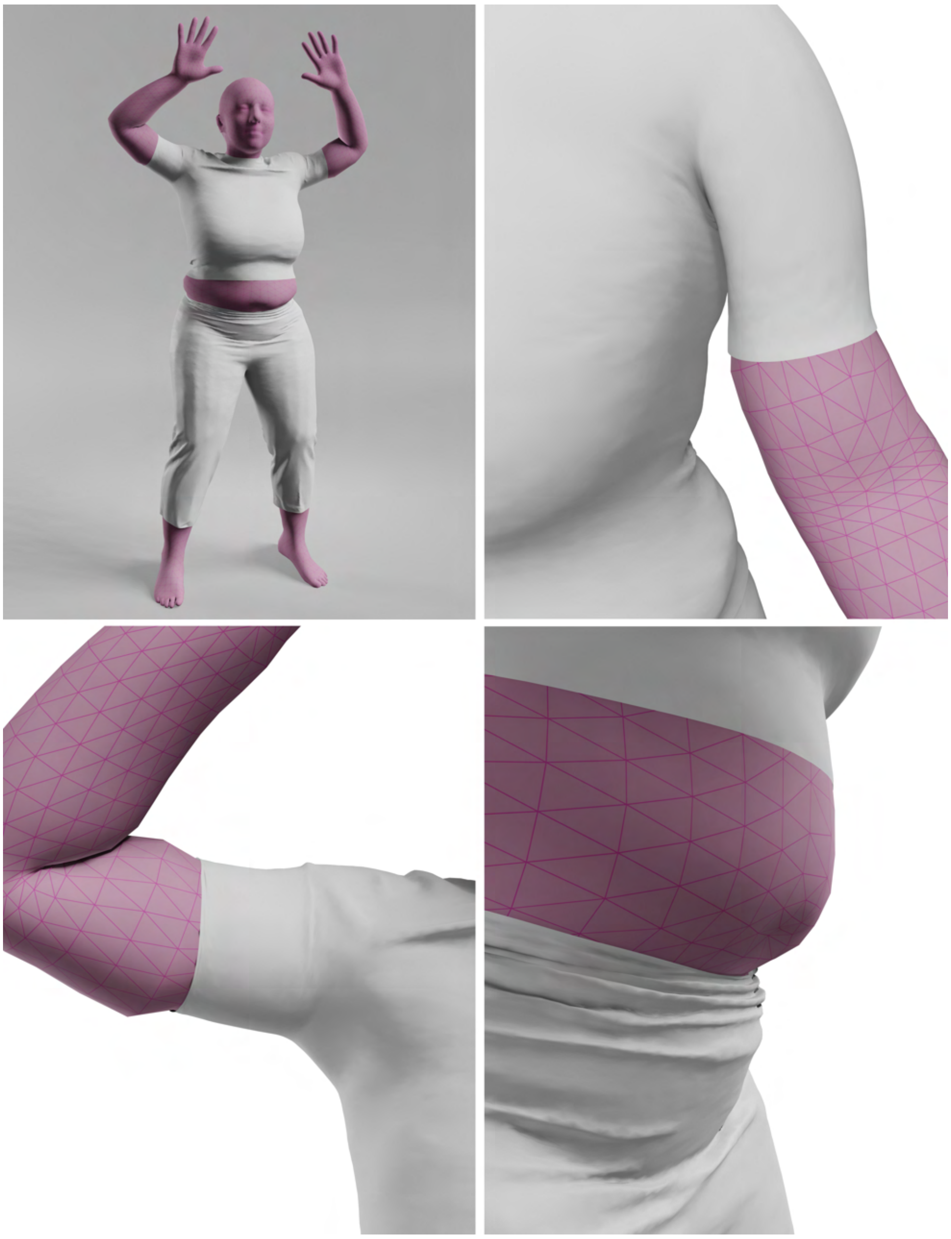


Figure 3.6: This picture shows four instances of soft-tissue deformation due to the effect of tight cloth, contact and friction.

In the next section, we will present how we characterize the thickness, $\mathbf{h}$, and material, $\{Y_N, Y_T\}_i$, parameters from 4D human captures using numerical optimization. To reduce the computational burden of solving this problem, we consider the material distribution is symmetric along the longitudinal axis, thus we only care about half of the body. Plus, parameters are defined at a reduced number $C = 42$ of control points distributed throughout the surface and interpolated to surface vertices using biharmonic interpolation [Jac*11]. Finally, material parameters are interpolated from surface points to volume vertices using again Laplacian interpolation. To ensure that these properties do not decay with the distance to the surface, values at the vertices of the inner surface are set equal to the value at vertices in the outer surface. This yields a reduced parameter space of $\mathbf{d} \in \mathbb{R}^{3C}$.

3.3.4 Full Space Dynamics

Simulating the deformation of the FEM model due to skeletal-driven animation and external interactions requires solving the system of nonlinear differential equations defined by discrete Newtonian dynamics. Acceleration and external forces add a deformation offset to the skinned shape $\mathbf{p}(\bar{\mathbf{x}}, \theta)$ defined in Section 3.2.4, resulting in the *deformed shape* configuration $\mathbf{x} \in \mathbb{R}^{3M}$ of the volumetric mesh. Then, the dynamics problem is formulated as:

$$\begin{aligned} &\mathbf{M}\ddot{\mathbf{x}} - \mathbf{f} - \mathbf{f}_e = 0 \\ &\mathbf{C}_{\mathcal{B}} \cdot \mathbf{x} = \mathbf{C}_{\mathcal{B}} \cdot \mathbf{p}(\bar{\mathbf{x}}, \theta). \end{aligned} \tag{3.5}$$

Here, $\mathbf{C}_{\mathcal{B}}$ is a matrix selecting only those vertices in the inner boundary of the soft-tissue layer, which are kinematically constrained to the skinned configuration defined by the statistical model $\mathbf{p}(\bar{\mathbf{x}}, \theta) \in \mathbb{R}^{3M}$ for a given reference shape $\bar{\mathbf{x}} \in \mathbb{R}^{3M}$ and time-dependent pose θ. Additionally, $\mathbf{M}$ is the mass matrix, $\mathbf{f}_e$ groups all external forces produced by environmental interactions and $\mathbf{f}$ is the sum of all internal forces generated by skin elastic response. Internal forces can be computed from the derivative $\mathbf{f} = -\nabla_{\mathbf{x}} U$ of the scalar elastic potential $U(\mathbf{x})$ introduced in Section 3.3.1.

3.4 Data-Driven Estimation

In the previous sections, we have introduced our hybrid method for the deformation of soft-tissue animated characters. The performance of this method heavily depends on having a good characterization of all the parameters affecting the deformation, which can be an arbitrarily complex task. This includes shape and pose parameters used by the statistical model in (3.1) but, more importantly, the geometry and material constituting the soft-tissue layer. In this section, we will describe in detail how these parameters are automatically estimatated from 4D captures of people using numerical optimization. Our key contribution is the definition of a novel error metric based on the anisotropic motion variance of surface vertices. This is required to accurately estimate the value of the anisotropic material

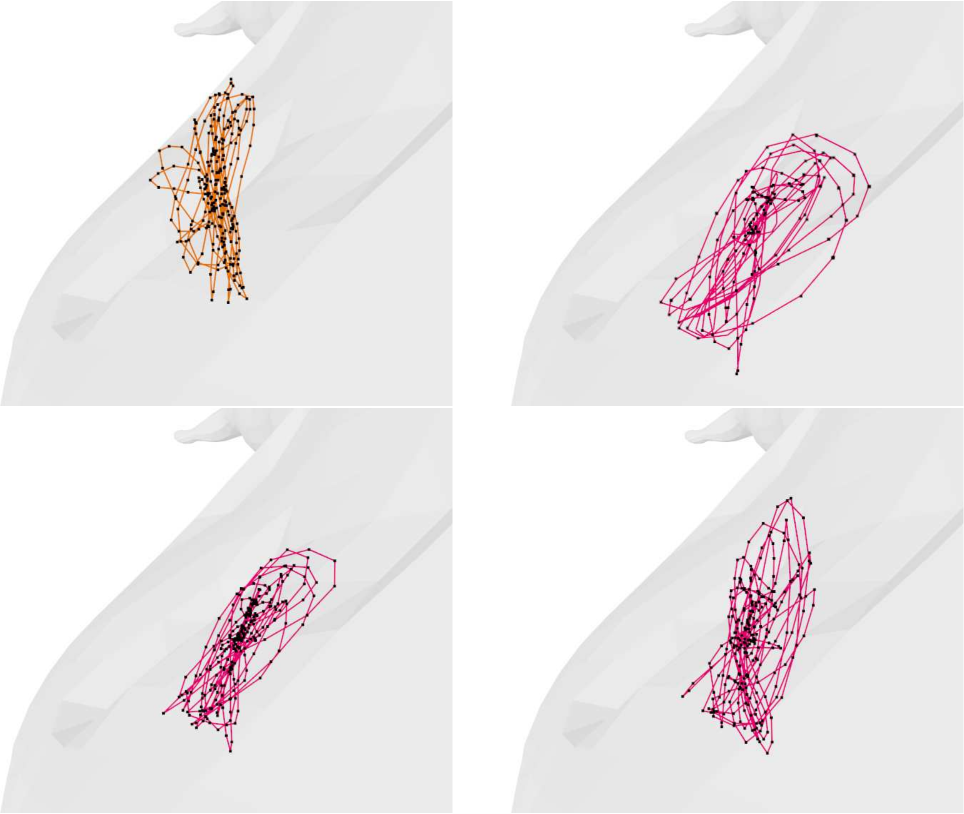


Figure 3.7: Picture showing the trajectory of a vertex for different configurations of material and error metric (from left to right, top to bottom): captured data, isotropic-isotropic, isotropic-anisotropic, anisotropic-anisotropic. It can be seen that the last configuration clearly outperforms the rest.

parameters. In addition, we use sparsity acceleration to make gradient-based optimization feasible.

The estimation of SMPL shape β and pose θ parameters that best approximate an input 4D capture can be formulated as a least-squares problem and has been previously described. We refer to e.g. [CO18] for a more detailed explanation. In the following sections, we will focus on the estimation of the parameters affecting the mechanical model. First, we will describe the input data and introduce metrics that can be used to characterize the avatars. And then, we will pose the optimization problem and describe in detail the methods used for solving it.

3.4.1 Input Data

Our input data consists of several sequences of animated 4D meshes captured from 3 different subjects performing various tasks [Pon*15]. For a given subject with V sequences, the i-th sequence $\mathcal{Z}_i$ is composed by a series of frames $\mathcal{Z}_i = \{\mathcal{Z}_i^1, \dots, \mathcal{Z}_i^{K_i}\}$ captured at 60 FPS. Each frame is a triangular mesh with $N = 6890$ vertices and exactly the same topology as the template mesh $\mathcal{T}$ described in Section 3.2.1. Sequences are initially grouped by subject and preprocessed to estimate SMPL parameters as described in [CO18]. This results in a unique shape parameter vector β per subject and as many pose parameter vectors as the total number of frames across all sequences $\{\theta_1^1 \dots \theta_1^{K_1}, \theta_2^1 \dots \theta_2^{K_2}, \dots, \theta_V^1 \dots \theta_V^{K_V}\}$.

3.4.2 Motion Variance Metrics

We aim to minimize some distance metric between the soft-tissue deformation produced by our hybrid method and the 4D captures under the same input animation. One straightforward choice would be to use the L2 distance between the captured and simulated data, but we are not interested in matching exact trajectories. Instead, we seek to approximate the characteristic movement of the captured data and, for that purpose, L2 distance between trajectories has some undesirable properties. In particular, for static deformations, the error is not zero because SMPL does not exactly match captured data. In situations with low dynamic behavior, the base error caused by the static deformation might hide the contribution of dynamic effects. For these reasons, we look for a simplified descriptive metric of the surface motion in order to approximately match the overall deformation behavior of the skin. We explore the use of the statistical variance of vertex trajectories.

Through early observation of captured data we noticed a skewed distribution of vertex trajectories, where the resulting point cloud takes the approximated shape of an ellipsoid aligned with the surface normal as shown in Fig. 3.7. This is caused by the highly nonlinear and anisotropic nature of skin dynamics. As a consequence, in our initial estimation experiments, the isotropic motion variance of the deformed mesh approximately matched that of the input captures but still the resulting trajectories differed significantly. This anomaly motivated the creation of a custom anisotropic material as well as exposed the need for a different motion variance metric that takes into account anisotropy.

For notation simplicity, in the following we will drop the subscripts corresponding to subjects and sequences. The trajectory of the i-th vertex of a mesh within a sequence $\mathcal{Z}$ follows a discrete path $\pi_{\mathcal{Z},i} = \{\mathbf{z}_i^1, \dots, \mathbf{z}_i^K\}$, where the subscript and superscript indicate respectively the index of the vertex and the frame. This trajectory includes large displacements due to the large differences in input poses. This might result in a very high dispersion that would not correctly represent the local behavior of the skin. For this reason, we first transform vertex positions to unposed reference space by applying the inverse skinning operation $\mathbf{P}^{-1}(\theta)$ as described in Section 3.3.1. Therefore, we define the anisotropic motion variance of a vertex as

$$\begin{aligned} M_N(\pi_{\mathcal{Z},i}) &= \frac{1}{K}\sum_{j=1}^{K} |\mathbf{N}_i(\mathbf{P}_i^{-1}(\theta^j)\cdot \mathbf{z}_i^j - \bar{\mathbf{z}}_i)|^2, \\ M_T(\pi_{\mathcal{Z},i}) &= \frac{1}{K}\sum_{j=1}^{K} |(\mathbf{I}-\mathbf{N}_i)(\mathbf{P}_i^{-1}(\theta^j)\cdot \mathbf{z}_i^j - \bar{\mathbf{z}}_i)|^2, \\ \bar{\mathbf{z}}_i &= \frac{1}{K}\sum_{j=1}^{K} \mathbf{P}_i^{-1}(\theta^j)\cdot \mathbf{z}_i^j, \end{aligned} \tag{3.6}$$

where $\mathbf{N}_i$ is the projection matrix to the subspace defined by the normal of the surface at the i-th vertex, $\mathbf{P}_i^{-1}(\theta^j)$ is the inverse skinning transformation corresponding to the i-th vertex at the j-th frame, and $\bar{\mathbf{z}}_i$ is the average vertex posistion across all frames. This allows us to define an anisotropic variance residual between the simulated and captured sequences:

$$D(\mathcal{X},\mathcal{Z}) = M(\mathcal{X}) - M(\mathcal{Z}), \tag{3.7}$$

where $M(\mathcal{X}) = \{M_N(\pi_{\mathcal{X},i}), M_T(\pi_{\mathcal{X},i})\}$, $i = 1,\ldots,N$, is a vector containing the normal and tangential motion variances corresponding to all the vertex trajectories of a sequence. Note that here we only consider those vertices in the outer surface of the soft-tissue layer. In the following sections, we will pose an optimization problem to minimize an error based on this residual and show that our anisotropic distance metric leads to a more precise description of skin behavior and ultimately better results.

3.4.3 Optimization Procedure

For a given subject with V sequences, we pose the optimization as a nonlinear least squares problem based on the residual defined in (3.7)

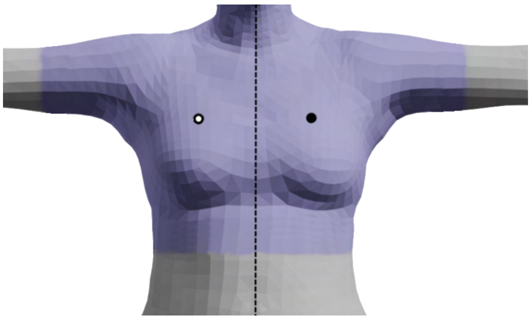
$$\min_{\mathbf{d}} \sum_{i=1}^{V} D(\mathcal{X}_i(\mathbf{d}),\mathcal{Z}_i)^T D(\mathcal{X}_i(\mathbf{d}),\mathcal{Z}_i), \tag{3.8}$$

where $\mathbf{d}\in\mathbb{R}^{3C}$ is the vector that concatenates all free parameters described in Section 3.3.3. For notation simplicity and without loss of generality we will assume we are optimizing just one sequence. We aim to solve this problem using a gradient-based method to take advantage of all the available machinery in standard optimization packages. This requires the computation of the gradient as $\nabla_{\mathbf{d}} D \cdot D$ which is highly nontrivial. Each evaluation of the residual D requieres the simulation of the complete animated sequence, which means solving a large number of nonlinear systems of equations defined in Section 3.3.4. Moreover, the solution of each dynamic problem depends on all the previous frames of the sequence which complicates the computation of the analytic Jacobian $\nabla_{\mathbf{d}} D$. Consequently, we opt to estimate the gradiend using finite differences, which requires one full evaluation of the simulation per optimized parameter.

To alleviate this large computational cost, we take advantage of the fact that the Jacobian $\nabla_{\mathbf{d}} D$ is sparse in practice. The trajectory of each of the deformed vertices depends only on the thickness and material distribution in a close neighborhood. For instance, it is very unlikely that the material of the leg and the mechanical behavior of the chest are related in any way. We solve the optimization problem using the off-the-shelf trust-region nonlinear least squares solution offered in *SciPy* optimization package that takes advantage of a sparsity pattern to estimate the Jacobian using finite-differences. The sparsity is defined through a matrix $\mathbf{S} \in \mathbb{R}^{N \times C}$ such that $S_{ij} \neq 0$ if and only if the mechanical behavior of the i-th vertex is affected by the parameters of the j-th control point. We build such matrix heuristically:

1. We first compute the biharmonic interpolation matrix from control points to surface vertices $\mathbf{H} \in \mathbb{R}^{N \times C}$. Then, we build the p-Influence matrix $\mathbf{H}_p$ by taking the p higher coeffiencients of each row and rounding them up to 1, leaving the rest of them to 0. This relates each surface vertex to its p most influential control points.

2. The adjacency matrix $\mathbf{A}_p = \mathbf{H}_p^T \cdot \mathbf{H}_p$ defines neighborhoods of control points sharing surface vertices for which such control points are among the p most influential. This adjacency criterion can be finally used to build k-ring expansions of the influence matrix $\mathbf{S}(p, k) = \mathbf{H}_p \cdot \mathbf{A}_p^k$.



In our experiments, the values $p = 2$ and $k = 2$ result in a sufficiently sparse pattern and a good trade-off between the quality of the estimation and computation time. The inset figure shows the dependency area corresponding to one of the points in the chest and its corresponding symmetric point. To further alleviate the cost of the optimiza-

tion and avoid redundant effects of material
stiffness and layer thickess we separate both sets of parameters and follow and alternating optimization scheme. First, we optimize material parameters to convergence using a simple distribution of $C = 12$ points. Then we alternate optimization steps of the thickness and material parameters until no further improvement is possible. Finally, we take this initial result and refine it using the total number of control points $C = 42$, directly on an alternating scheme. In the following section, we will show the results of fitting material and thickness parameters from data for 3 different subjects with various input sequences and discuss the performance of the optimization procedure.

3.5 Estimation Experiments

The optimization procedure described in the previous section is capable of characterizing the parameters of our mechanical model. As it can be seen in Fig. 3.1, this produces soft-tissue characters that realistically deform under highly dynamic animations and external interactions. In this section, we describe in more detail our parameter estimations experiments and discuss our optimization scheme.

We estimated the parameters of our soft-tissue avatar model for three different subjects that we denote S_A, S_B and S_C (Fig. 3.8). For each subject, we use two animation sequences of $1.5s$ as the training set and three animation sequences of $[3, 4]s$ as the test set. We focus our main analysis in this section on subjects S_A and S_B, as those show stronger dynamic behavior. Fig. 3.9 shows the distribution of the normal and tangential variance error defined

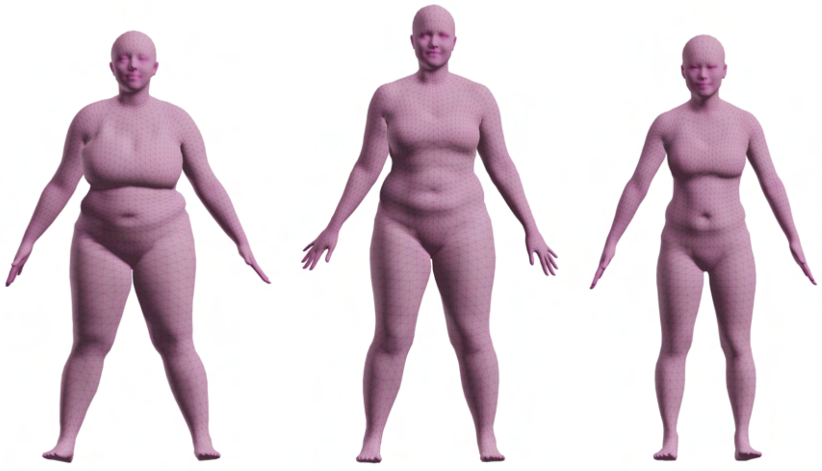
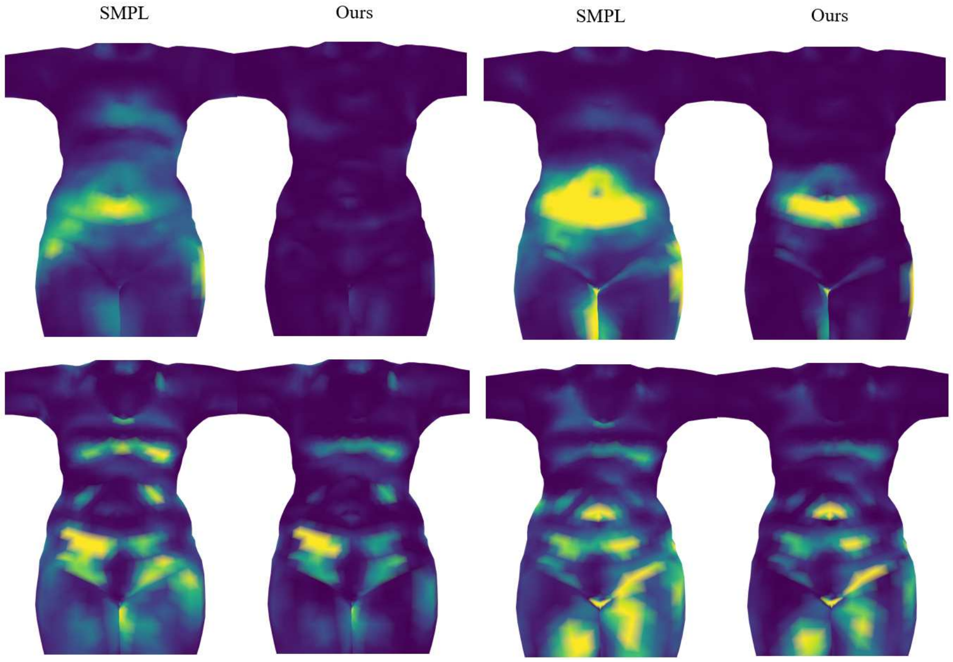
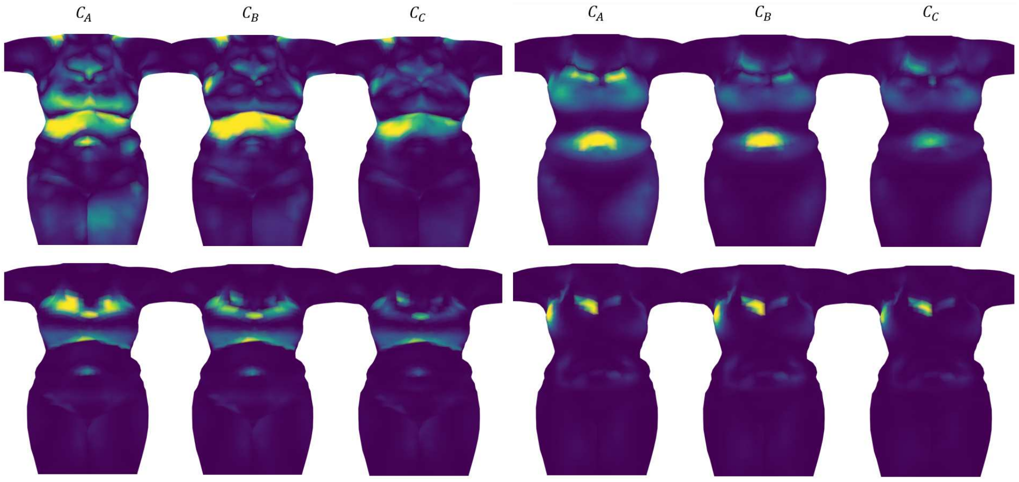


Figure 3.8: The three test subjects: S_A, S_B and S_C.



(a) Running on spot (Training) **(b)** One leg jump (Test)

Figure 3.9: This figure shows a comparison of the tangential (top) and normal (bottom) variance errors for subject S_B, between SMPL and our soft-tissue model. It can be seen that our method approximates the overall dynamic behavior better for both training and test sequences.



(a) Jumping jacks (Training) **(b)** Shake shoulders (Test)

Figure 3.10: This figure shows a comparison of the tangential (top) and normal (bottom) variance errors for subject S_A, using the three different configurations of the optimization. It can be seen that for C_C the error w.r.t. the captured data is lower for both training and test sequences.

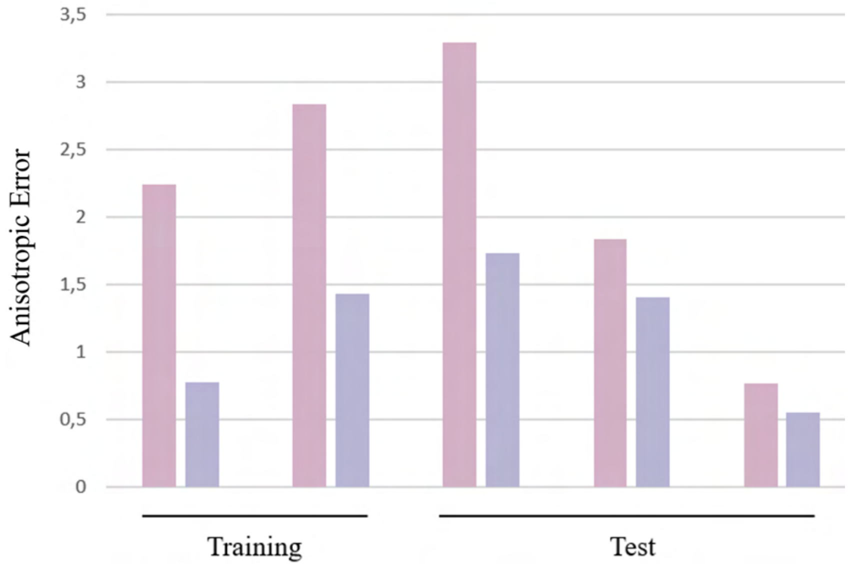


Figure 3.11: This graph shows a comparison of the anisotropic variance error for each of the five sequences, averaged across the three different subjects, between SMPL (pink) and our hybrid data-driven and physics based method (purple). It can be seen that our soft-tissue avatar clearly outperforms the SMPL solution.

in (3.8), for subject S_B in two different animation sequences. It can be seen that our method significantly reduces the error of the SMPL model and approximates the dynamic behavior of the captures from just two training sequences. Thanks to the formulation of the error metric in terms of the anisotropic motion variance, we notice that most of it is concentrated in the normal part of the residual. We believe that this is caused by an existing trade-off in our material definition between compliance in the normal direction and volume presservation (which is a desired property when modeling skin mechanics). This is expected as the skin has inherently limited stretch in the tangential direction, thus constraining the deformation in the normal direction if volume must be presserved. We found that using a Poisson's Ratio of $\mu = 0.2$ was a reasonable compromise solution. For future work, we would like to explore more complex soft-tissue models (e.g., multi-layer) that might alleviate this limitation. Overall, our method reduces the error of the statistical model by 56.8% in the case of training sequences and 31.4% in the case of test sequences (Fig. 3.11). For some particular tests that notably differ from the training set, the optimization is not fully capable of generalizing the behavior of the soft-tissue. In such cases, we have observed that the solution provided by SMPL produces smaller variance errors. We found that this happens because the anisotropic metric is particularly sensitive to the misalignment between the captured and simulated vertex trajectories. However, our method provides qualitatively better results.

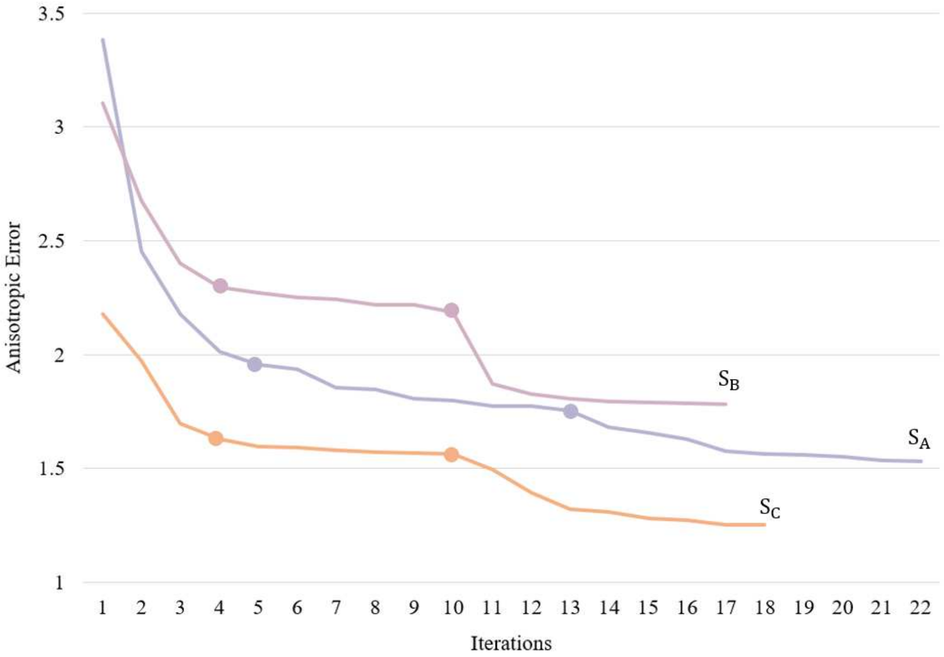


Figure 3.12: This graph shows the convergence of the optimization considering the anisotropic material and error metric for each of the three subjects tested. The dots indicate the iteration separating the three stages of the optimization: i) 12 points, material parameters; ii) 12 points, alternating material and thickness; and iii) 42 points, alternating material and thickness.

We repeated the optimization of subject S_A several times to compare the performance of our custom constitutive material when considering isotropic and anisotropic elastic moduli. Additionally, we explore the effect that our anisotropic error metric has on the results of the optimization . Overall, this result in three configurations:

- C_A: isotropic material with isotropic error metric.
- C_B: isotropic material with anisotropic error metric.
- C_C: anisotropic material with anisotropic error metric.

Fig. 3.7 shows a comparison between the trajectories obtained for one of the training sequences, for each of the mentioned configurations. It can be seen that the C_C produce vertex trajectories that fit much better with the captured one than C_A. While the anisotropic error metric used in C_B helps to improve the overall quality of the fit, the isotropic constitutive material cannot capture skin behavior accurately. Fig. 3.10 shows in detail the spatial distribution of the anisotropic variance for each of the considered configurations. It can be seen that the configuration C_C clearly outperforms the other two in approximating the captured data.

Overall, the complexity of the problem complicates convergence and suffers risk of getting stuck in local minima. Our optimization scheme notably alleviates the burden and makes solving the problem feasible. As shown in Fig. 3.12, it is effective to provide an initialization for the full optimization using the solution obtained with the reduced parameter space of 12 control points. However, the big computational cost remains the main limitation and prevents us from further exploring other options that might lead to better results. All our simulations and optimizations were run in a desktop machine Intel Core i7-8700 @ 3.2GHz, with 32GiB RAM. We use a time step of $1/120s$. Each time step converges in 1 or 2 Newton iterations, and the cost per iteration is around $0.7s$. We estimate the parameters of each subject using 2 sequences of $1.5s$ each. Alltogether, simulating the two sequences takes about $6min$. In each optimization step, we estimate the Jacobian of error terms using finite differences. With 42 control points, this implies running 43 simulations per thickness iteration and 85 simulations per material iteration. Our sparsity acceleration approximately divides the number of simulations by 2. Overall, this leads to a total optimization time of around 30-40 hours, depending on the subject and the chosen training set.

3.6 Construction of a Reduced Soft Skeletal Avatar

In this section, we describe the DoFs of our reduced skeletal soft model. The model combines the parametric soft avatar presented so far and a handle-based reduced deformable model. We first describe the handle-based reduction, and then conclude the section discussing how we merge it with the parametric avatar.

3.6.1 Handle-Based Reduced Model

Several reduced models build compact and sparse bases by *skinning* the deformations of a few handle points or frames. They differ in the choice of DoFs and the construction of the basis. Some possibilities are to use rigid frames skinned with harmonic weights [Gil*11], or affine transformations skinned using geodesic radial basis functions [BEH18]. We opt for the biharmonic generalized barycentric coordinates (BGBC) of Wang et al. [Wan*15], which interpolate both handle points and affine frames in a smooth, shape-aware manner. The BGBC are obtained by solving a quadratic minimization on a volumetric mesh subject to interpolation constraints at handles. Wang et al. propose an objective energy based on mass-weighting a linearly precise Laplacian, and we refer to their paper for details.

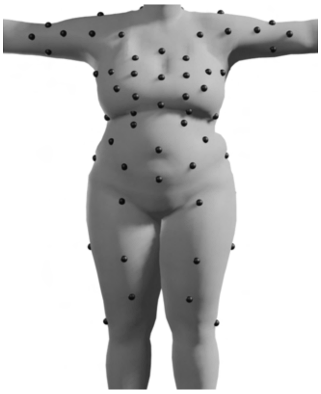
We discard BGBC values below a small threshold (in practice, we smoothly decay values between 0.1 and 0.03), and thus enforce local support of the basis.

Given a skeletal model, the BGBC reduced model allows us to naturally choose as handles the bones of the skeleton plus a sparse set of surface points. The model is well suited for fitting the shape (i.e., skin) of the skeletal model, as well as anatomical constraints (i.e., the bones and joints), while providing a compact deformation basis. To compute the BGBC basis, we mesh the soft-tissue volume between the outer skin surface and the inner skeleton, as shown in Fig. 3.3. Note that we only use the complete volumetric meshes during preprocessing, to compute the BGBC weights as well as cubature weights (Section 3.7.2).

Below, we show the computation of skin deformations $S_B \in \mathbb{R}^{3n}$ by linear combination of the positions $\phi \in \mathbb{R}^{3h}$ of h handle points and the vectorized affine transformations $A(\theta) \in \mathbb{R}^{12b}$ of b handle bones with pose θ.

$$S_B(\phi, \theta) = U_\phi \, \phi + U_\theta \, A(\theta). \tag{3.9}$$

$U_\phi \in \mathbb{R}^{3n \times 3h}$ and $U_\theta \in \mathbb{R}^{3n \times 12b}$ represent, respectively, the basis vectors of handle points and handle bones, and store the BGBC of all handles.



The reduced model (3.9) can be used as kinematic representation in dynamic simulation, endowed with an FEM elasticity model, as well as joint and contact forces. The inset figure shows the distribution of handle points manually selected all over the surface of the body for the presented reduced model examples. The major advantage of the method is that contact produces both skeletal reaction and soft-tissue deformation, with natural two-way coupling. However, deformations are governed solely by physics, making it very hard to match real deformation examples.

3.6.2 Combined Model

Next, we present our skeletal model augmented with reduced soft-tissue deformations, by combining the two models presented before. We represent the effect of bone transformations using the parametric skeletal model, i.e., through the pose blend-shapes $V_p(\theta)$ and the skinning transformation $W(\cdot)$. This provides a pose-dependent soft-tissue deformation that realistically depicts data-driven static deformation and with skeletal dynamics due to contact. In addition, we aim to achieve flesh deformations due to contact or inertia, while still preserving the accurate data-driven static deformation of the parametric skeletal model. Thus, we formulate reduced soft-tissue deformations S_B in the unposed reference shape, to capture deviations from the parametric skeletal model instead of full deformations. Our model enables simple construction of anthropomorphic models that match both flesh and bone geometry, while supporting large joint rotations. This is a complex goal for typical models that evaluate world-space deformations, because the narrow space between bones is heavily deformed under joint rotations. With our unposed deformation model, joint rotations do not cause any difficulty.

Our design choices result in ignoring the bone transformations in the reduced deformation model, i.e., $A(\theta) = 0$ in (3.9), since the bones remain fixed in the reference shape, but their effect is captured by the parametric skeletal model. As a result, our augmented skeletal model defines soft-tissue deformations $S \in \mathbb{R}^{3n}$ as

$$\begin{aligned} S(\phi,\theta) &= W(\bar{V}, J(\beta_0), \theta, \mathbf{W}), \\ \bar{V}(\phi,\theta) &= \mathcal{V} + V_s(\beta_0) + V_p(\theta) + U_\phi\,\phi, \end{aligned} \tag{3.10}$$

The DoFs of the model are the handle points ϕ and the skeletal pose θ. The reduced deformation $U_\phi\,\phi$ is evaluated on a volumetric mesh on the unposed reference shape. However, as already discussed in Section 3.6.1, in our implementation the rest of the model, and hence S, is evaluated only on the surface . We fix the shape parameters $\beta = \beta_0$ to create the model for a given (but arbitrary) skeletal shape. Then, the joint locations and the shape-based blend shapes remain fixed during the simulation. Note also that, even though the bone basis vectors U_θ can be ignored in the runtime simulation, the BGBC basis must be computed with both point and bone basis vectors U_ϕ and U_θ. This ensures that bones do not suffer soft-tissue deformation.

In the next section, we discuss how we endow the combined kinematics representation (3.10) with elastic and inertial models for dynamic simulation. As shown in Fig. 3.13, our model enjoys the advantages of both the parametric skeletal model and the reduced deformation model; it matches static contact-free deformations accurately, and contact produces both skeletal reaction and soft-tissue deformation, with natural two-way coupling.

3.7 Reduced Soft Skeletal Mechanics

The augmented skeletal model described in the previous section is capable of generating skeletal animations with soft-tissue deformations. However, to generate these deformations

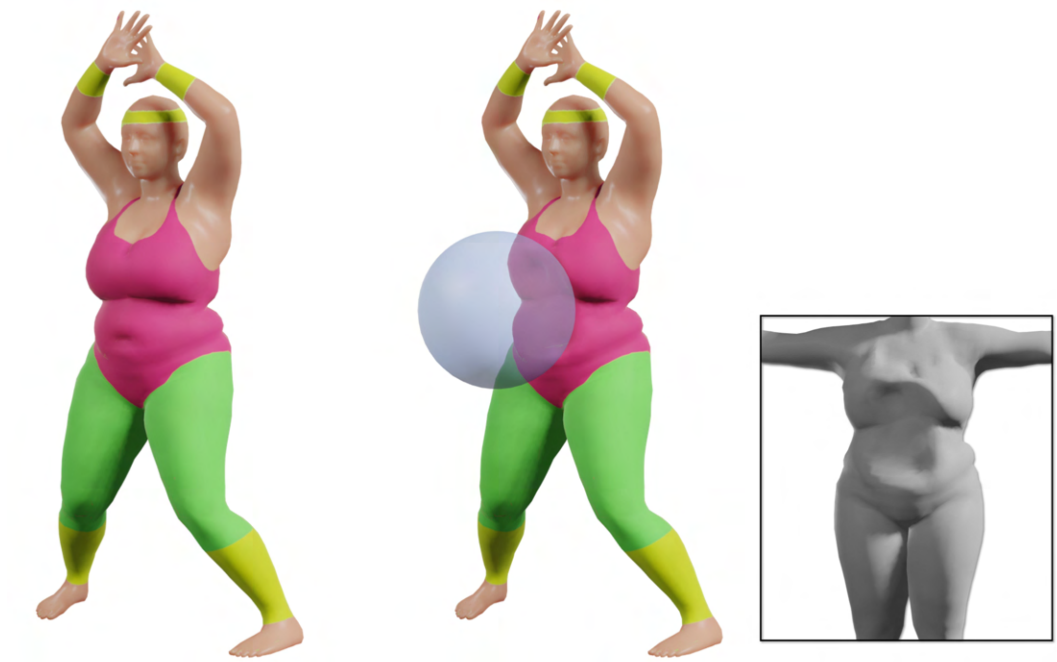


Figure 3.13: Our soft skeleton body model matches static data-driven deformations accurately for a given pose (left). When in contact with an external object it produces a dynamic response with two-way coupling (right). Soft-tissue deformation is clearly noticeable in the unposed reference shape (inset). Local static deformations due to the pose blend shape (e.g., arm bulging) apply independently of external interactions.

in a physics-based simulation, we need to define force models, as well as dynamic equations that relate those forces to the motion of the skeletal and soft-tissue DoFs. We start with our formulation of reduced deformation mechanics in unposed space, followed by a novel cubature scheme that enables efficient evaluation of energies, forces and Jacobians. We complement the deformation model with a description of skeletal forces, contact handling, and the formulation and solution of dynamics.

3.7.1 Unposed Deformation

We measure soft-tissue deformations as deviations from the parametric skeletal motion. To this end, we require two ingredients. One is to define displacements in unposed space as in (3.10). The other one is to define elastic energy solely based on these displacements.

Let us take a point $\bar{x}$ in reference shape as introduced in Section 3.2.4. We define the soft-tissue deformation as the unposed displacement $U_\phi\, \phi$ from this rest position, which yields the unposed deformation $u = \bar{x} + U_\phi\, \phi$. We compute an unposed deformation gradient as $F = \nabla_{\bar{x}} u = I + \nabla_{\bar{x}} U_\phi\, \phi$. This definition of the deformation gradient resembles the one proposed for the full model in Section 3.3.1. However, this time kinematics are discretized using unposed displacements, not world positions.

Given a generic elastic energy density of the form $\Psi(F)$, elastic forces on the handle points are computed as

$$\nabla_\phi \Psi = \nabla_F \Psi \cdot \nabla_{\bar{x}} U_\phi, \tag{3.11}$$

where $\nabla_F \Psi$ is the First Piola-Kirchhoff stress, and $\nabla_{\bar{x}} U_\phi$ denotes the gradient of basis vectors w.r.t. the reference shape. In our dynamics solver, we also require force Jacobians, which in turn use the Hessian of the elastic energy.

As evident from the expression above, our deformation model does not affect the skeletal pose θ. In other words, a static change of skeletal pose does not produce additional soft-tissue deformation, and the skin surface is controlled solely by the parametric model, as desired. However, as shown in Fig. 3.15 and Fig. 3.17, the presence of dynamic inertial terms and external contacts does produce an effect on the deformation of the soft-tissue.

In our examples, we have used the nonlinear material model introduced in Section 3.3.2. In practice, we have clamped the saturation term in the material model, to limit its highly nonlinear effect. We found that very high saturation values could require more than one Newton iteration in the backward Euler solve (see Section 3.7.4), and this had a significant impact on performance.

3.7.2 Data-Oblivious Cubature

The deformation model described above must be integrated on the whole soft tissue to compute elastic forces on the handle points. To this end, we consider the volumetric tetrahedral meshes shown in Fig. 3.3 and use the finite element method (FEM) to discretize the continuous formulation in (3.11). With reduced models, the runtime cost is dominated by the computation and assembly of energy gradients and Hessians on all mesh elements and/or quadrature points, each of which must be projected onto the reduced space through the basis matrix as in (3.11). When the basis vectors are smooth, this cost can be alleviated thanks to cubature methods [AKJ08; Tyc*13]. They rely on the approximation of the energy, its gradient and Hessian as weighted sums of evaluations at a sparse set of points.

Typical cubature methods use deformation examples to optimize cubature point locations and cubature weights. However, the generation of representative training examples is a challenging task. Instead, we propose a data-oblivious cubature method based on the construction of an interpolation scheme on the skeletal model. We sample cubature points uniformly, construct an interpolation scheme for the elastic energy and its derivatives, and then derive cubature weights automatically by integrating the basis functions of the interpolation scheme.

Let us assume a set of cubature points $\{X_i\}$, sampled on the reference shape of the soft-tissue portion of the skeletal model, i.e., excluding the bones. We will discuss the sampling strategy shortly. We wish to construct an interpolation scheme for the elastic energy and its derivatives. Thanks to the reduced deformation model, both the energy and its derivatives are smooth, and a sufficiently dense interpolation scheme can provide a good approximation. We use BGBC as interpolation scheme, which boils down to computing the BGBC for the cubature points $\{X_i\}$ on the volumetric mesh of the soft-tissue portion of the skeletal model. Note that these BGBC are different from those of the handles introduced in Section 3.6.1.

Given the elastic energy density $\Psi(X)$ (and its derivatives) sampled at the cubature points, $\{\Psi(X_i)\}$, we interpolate the energy using BGBC $\{\alpha_i(X)\}$ as $\Psi(X) = \sum_i \alpha_i(X)\,\Psi(X_i)$. We can compute the full energy E by integrating this interpolated energy density on the whole reference shape:

$$E = \int_{\mathcal{V}} \sum_i \alpha_i(X)\,\Psi(X_i)\,d\mathcal{V} = \sum_i \left(\int_{\mathcal{V}} \alpha_i(X)\,d\mathcal{V} \right) \Psi(X_i). \tag{3.12}$$

Alternatively, we can define the full energy E using cubature, with cubature weights $\{w_i\}$:

$$E = \sum_i w_i\,\Psi(X_i). \tag{3.13}$$

Simply by equating both expressions, we see that the cubature weights can be obtained by integrating the BGBC on the reference shape.

$$w_i = \int_{\mathcal{V}} \alpha_i(X)\,d\mathcal{V}. \tag{3.14}$$

Our cubature method provides a good approximation of the energy and its derivatives, but it is also fast.

We have explored several strategies for sampling cubature points. One strategy was to co-locate cubature points and basis handles, and thus interpolate the internal energy and its derivatives from the same samples as the deformation field. However, internal forces are simply not uniform at the surface of rigid handles (i.e., bones); therefore, the interpolation scheme is not well defined. Another strategy was to apply importance sampling, favoring denser sampling at locations with potentially larger deformation. However, due to our approach to compute cubature weights in (3.14), some cubature samples with large

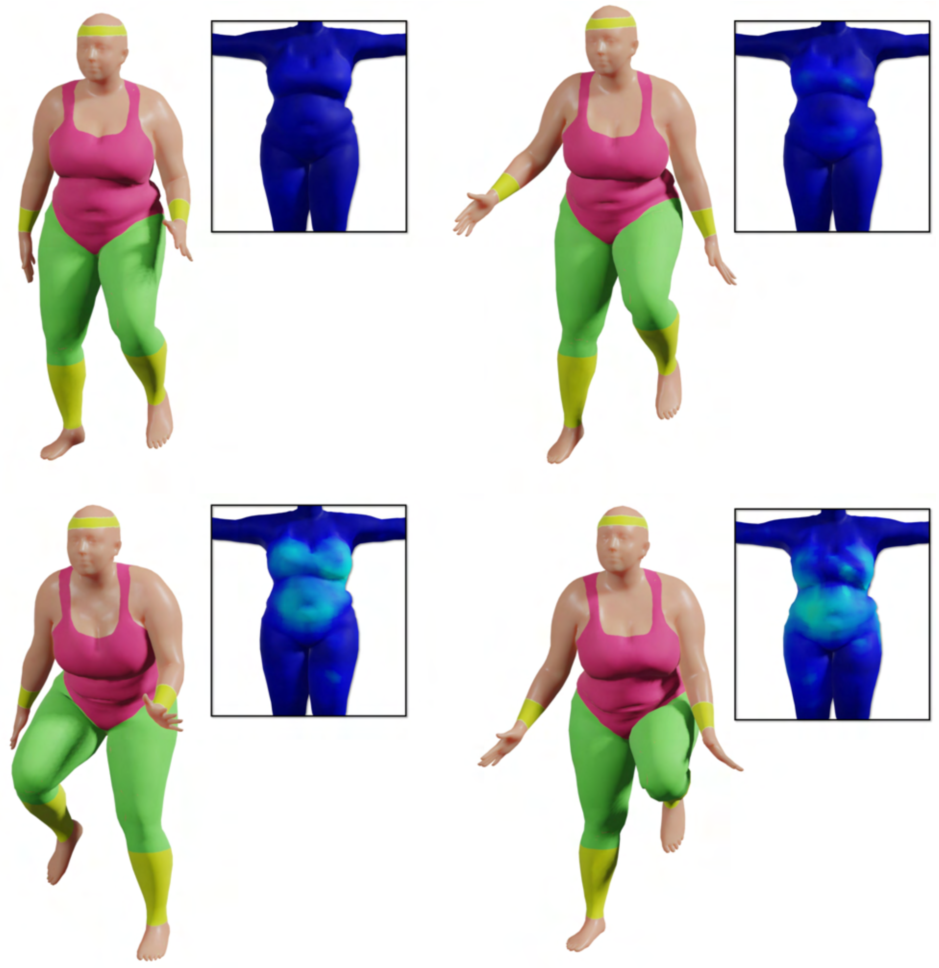


Figure 3.14: Our model incorporates skeletal tracking of target input animations through control forces. Note that even in the absence of external interactions, dynamic deformations appear as an effect of inertial components. The figure shows a sequence of four animation frames and a colormap with the corresponding vertex displacements in unposed space (insets).

deformation receive a large influence from regions with little deformation, leading to wrong force estimations. We concluded that uniform sampling was the safest strategy, and we found no apparent increase in quality after using 5 to 10 cubature points per basis handle.

3.7.3 Skeletal Constraints, Forces and Contact

Our augmented skeletal model (3.10) defines the skeletal DoFs θ, but we must define joint constraints and forces to correctly simulate skeletal dynamics. Furthermore, it is important that the skeletal pose remains within the observed subspace of the parametric model, otherwise the pose-dependent blend shapes will produce unnatural results. To this end, we implement the following types of skeletal forces: joint constraints to maintain the skeletal structure, joint stiffness to capture bending resistance, joint limits, and PD control forces (a.k.a. viscoelastic coupling) to enable interactive tracking and/or animation control as shown in Fig. 3.14.

We model all skeletal forces, including joint constraints, through a combination of translation and rotational springs. These forces act only on the skeletal pose θ, with no direct effect on soft-tissue deformation ϕ.

We evaluate contact forces on surface points $x \in S$, which are computed according to (3.10) as $x = W(\bar{V}(\phi, \theta), \theta)$. Recall that $W(\cdot)$ is the skinning transformation, and $\bar{V} = u(\phi) + V_p(\theta)$ is the unposed deformation of the reference shape. In our implementation, we define penalty contacts for colliding surface points. To transfer penalty contact forces to the DoFs of the model, we require the Jacobian of the surface position x w.r.t. the DoFs. The same Jacobian would be necessary for constraint-based contact handling.

Since surface positions depend on both the skeletal pose θ and the soft-tissue handle points ϕ, contact forces act on both the skeleton and the soft-tissue deformation. Effectively, this shared dependency plays the role of two-way coupling between skeleton and soft tissue. The Jacobians of the surface position can be computed as:

$$\frac{\partial x}{\partial \theta} = \frac{\partial W}{\partial \bar{V}} \frac{\partial V_p}{\partial \theta} + \frac{\partial W}{\partial \theta}, \qquad \frac{\partial x}{\partial \phi} = \frac{\partial W}{\partial \bar{V}} U_\phi. \tag{3.15}$$

We have observed that, in practice, the forces on the skeleton are dominated by the direct influence on the skinning transformation through $\frac{\partial W}{\partial \theta}$, with the influence through the pose blend-shapes $\frac{\partial V_p}{\partial \theta}$ negligible. To maximize performance, we have discarded the latter in our implementation. .

3.7.4 Reduced Dynamics

To simulate dynamic deformations of our augmented skeletal model, we integrate equations of motion of the form

$$M \begin{pmatrix} \ddot{\theta} \\ \ddot{\phi} \end{pmatrix} = \begin{pmatrix} f_\theta \\ f_\phi \end{pmatrix}. \tag{3.16}$$

We integrate the dynamic equations using backward Euler with one Newton iteration, and we achieve inertial soft-tissue deformations induced by skeletal motion, such as those in

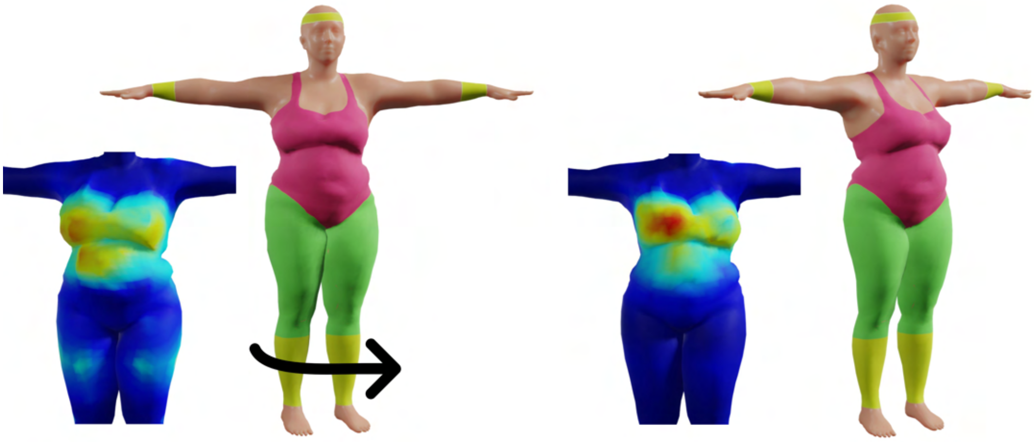


Figure 3.15: Our handling of inertial forces correctly captures soft-tissue deformations induced by the skeletal motion. The picture shows two frames of an animation with a character subject to rotational acceleration (left) and deceleration (right) in the vertical axis, and the resulting colormap of soft-tissue deformations.

Fig. 3.15. As discussed before, the Newton solve requires the computation of the various force Jacobians.

The bone forces f_θ include the skeletal forces described in Section 3.7.3. The reduced soft-tissue forces f_ϕ include the elastic forces described in Section 3.7.1. Contact forces affect both bone and soft-tissue forces, as described in Section 3.7.3. The mass matrix is derived by formulating the kinetic energy of the system. This boils down to:

$$M = \int_{\mathcal{V}} \frac{1}{2} \rho \left(\frac{\partial x}{\partial \theta} \; \frac{\partial x}{\partial \phi} \right)^T \left(\frac{\partial x}{\partial \theta} \; \frac{\partial x}{\partial \phi} \right) d\mathcal{V}, \tag{3.17}$$

with the Jacobians given in (3.15). We leverage the cubature method discussed in Section 3.7.2 to approximate the mass matrix, with one modification: we must add the inertia terms of the bones to the mass matrix computed on the volumetric mesh.

3.8 Reduction Experiments

The soft skeleton model presented above provides a general scheme to augment any arbitrary PSD with reduced soft-tissue deformations. The resulting method features realistic dynamics, highly non-linear materials, and two-way coupling between flesh and bones through frictional contact and inertia. As global static deformations are handled by the

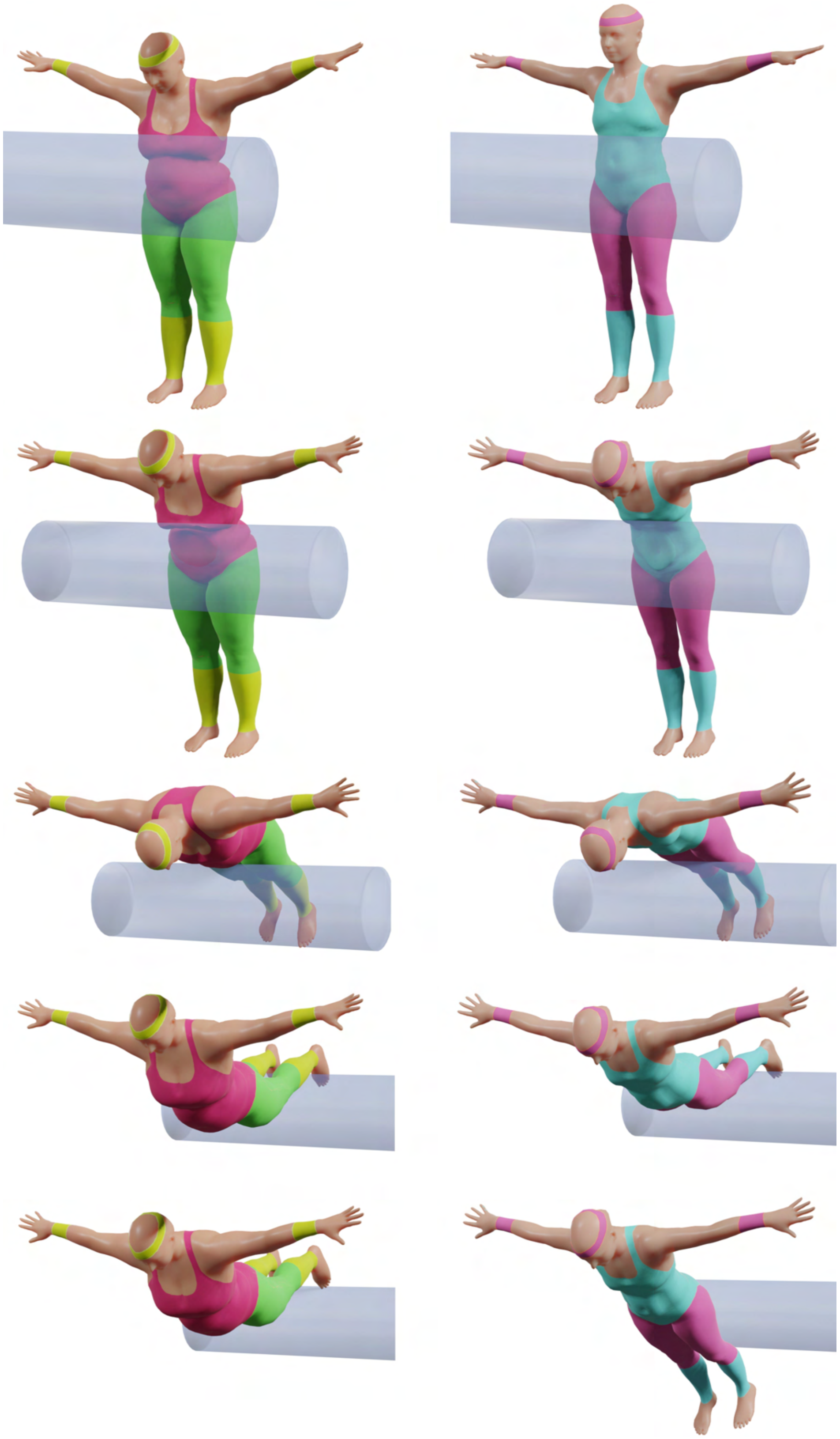


Figure 3.16: Each column presents a sequence of pictures showing two rag-doll characters with different shape parameters undergo the same simulation scenario. The bodies are slowly pushed by a cylinder while hanging from the wrists, resulting in different deformation behaviors.

data-driven pose blend shapes, the subspace basis is completely devoted to representing local deformations.

To define the soft-tissue layer, we follow the approach described in Section 3.2. The resulting tetrahedral mesh comprises a total of 10111 nodes and 41585 elements , 500 of which are selected as cubature integration points. We manually sample $h = 82$ handle points uniformly distributed throughout the surface of the body which leads to a total of 672 simulated DoFs.

As shown in Fig. 3.13, the resulting skeletal model accurately matches data-driven static deformations while at the same time realistically responding to external interactions. The deformation of the skeleton can be driven by an input animation through tracking forces (Fig. 3.14), and the soft-tissue dynamically deforms accordingly due to the effect of inertial forces (Fig. 3.15). Interestingly, as shown in Fig. 3.16, the parametric model allows us to reproduce the same simulation scenarios for different body shapes resulting in different deformation behaviors.

On average, simulations run at close to 100 FPS allowing us to interactively manipulate the avatars. Fig. 3.17 shows a comparison of the deformation behavior obtained by solving the same simulation scenario considering the full-space problem and our subspace approximation. While differences are noticeable, our soft skeleton model produces plausible behaviors while running in real-time. In comparison, solving the full space problem takes around 7 seconds per frame.

Table 3.1 shows a breakdown of timings, comparing a full-space simulation, our subspace method with the number of cubature points used in the examples, and our method with more cubature points. The subspace formulation provides a speed-up of over $1000\times$ on the linear system solve of each time step. On the other hand, cubature integration provides a speed-up of almost $400\times$ on the evaluation of forces and the energy Hessian. We confirm that the force and Hessian evaluations scale sublinearly with the number of cubature points.

	force	Hessian	solve	step
full space	882	2401	3947	7233
subspace 500	1.51	7.16	0.49	9.17
subspace 1400	1.99	11.63	0.47	14.1

Table 3.1: Performance comparison of a full-space simulation vs. our reduced method, with different numbers of cubature points (denotes as “subspace X”). The table shows average timings per time step, as well as broken down into the evaluation of forces, the evaluation of the energy Hessian, and the linear system solve (all in ms).

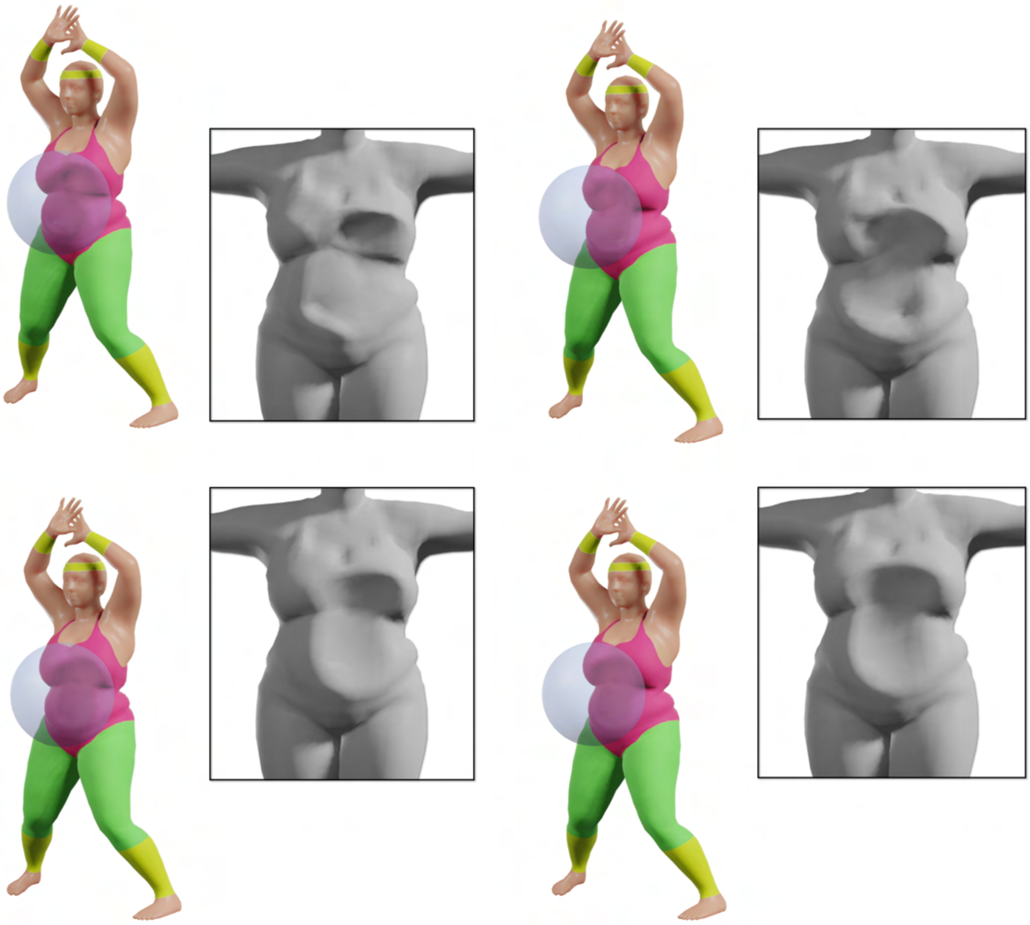


Figure 3.17: We compare the behavior of our method (top) against the ground-truth full-space problem (bottom). While the full-space solution is slightly smoother, our approximation produces a plausible behavior at a computational cost orders of magnitude lower.

3.9 Conclusions

We have presented a hybrid model of soft-tissue animated avatar that computes deformation as a combination of a data-driven statistical model and an FEM simulation. In addition, we have presented an optimization procedure to estimate the parameters of the model from 4D human captures. Unlike previous work, we ensure that two important properties are achieved. We reformulate strain computations such that, in absence of external interactions and dynamics effects, the deformation generated by our method matches exactly that of the data-driven statistical model. Plus, we model soft-tissue mechanics using a custom constitutive material that captures the nonlinearity and anisotropy of the skin, which are essential for representing complex interactions. To best explore the parameter space of our material model, we have formulated the estimation problem in terms of a novel anisotropic motion variance metric. Also, accompanying this soft-tissue avatar model, we have presented a reduced subspace simulation, effectively augmenting the model with interactive soft-tissue deformations.

In general terms, our method is capable of realistically modeling soft-tissue avatars at interactive framerates, under highly dynamic effects and with extreme external interactions. However, our work is not free from limitations.

By measuring deformations in the unposed reference space, we suffer errors in the estimation of the deformation gradient. Some of these errors are actually intended, and they help the model reach the data-driven pose without causing internal stress. But other errors may alter the expected behavior of the material model. This occurs particularly near joints, where the skinning transformation from unposed to posed space deviates most from a rigid transformation. With our nonlinear material, this deviation may induce overestimation or underestimation of the saturation term. We believe, however, that the benefits of the unposed deformation metric surpass its limitations. One additional benefit is the robustness under large joint rotations. The unposed formulation avoids mesh degeneracies that are otherwise commonplace near joints.

In addition, during our experiments we detected that there is a trade-off between the volume preservation and the anisotropic capabilities of the material, which limits the quality of the fitting to captured data. This might be solved using a more complex model of the soft-tissue that would consider several layers with different mechanical properties. On the other hand, characterizing material parameters requires solving a complex optimization problem with large computational cost. Even though our sparsity accelerated gradient alleviates part of the burden, using finite-differences for the computation of the Jacobian quickly becomes hardly feasible for a large number of control points. It would be interesting to find an appropriate reformulation of the dynamics problem such that the gradient can be analytically approximated. Solving both these problems would constitute interesting lines of future work.

When the subspace deformation model is used, there are evident limits on the range of possible deformations, of course with the benefit of computational efficiency. One of the the main restrictions found in the presented reduced model is the limited ability to resolve contact deformations with high spatial frequencies. To alleviate this limitation the subspace basis can be enriched at runtime. As a novel alternative, in the following chapters and in the remainder of this thesis, we describe new data-based methods to augment subspace contact simulations with learned corrections, improving the generated contact deformations while maintaining a reduced number of simulated DoFs.

As a minor note, in our particular choice of subspace basis, we have focused the placement of handle points on the surface of the model. However, one could consider adding handle points within the volume to capture internal heterogeneity. Also, in the construction of our subspace cubature integration, we have used uniform sampling, but a better cost-to-accuracy trade-off might be possible with the addition of importance sampling.

4

Reduced Deformation Modeling with Learned Corrections

This chapter introduces a novel subspace method for the simulation of dynamic deformations. The method augments existing linear handle-based subspace formulations with nonlinear learning-based corrections parameterized by the same subspace. Together, they produce a compact nonlinear model that combines the fast dynamics and overall contact-based interaction of subspace methods, with the highly detailed deformations of learning-based methods. We propose a formulation of the model with nonlinear corrections applied on the local undeformed setting, and decoupling internal and external contact-driven corrections. We define a simple mapping of these corrections to the global setting, an efficient implementation for dynamic simulation, and a training pipeline to generate examples that efficiently cover the interaction space. Altogether, the method achieves unprecedented combination of speed and contact-driven deformation detail. The contributions presented in this chapter have led to the following publication:

- Cristian Romero, Dan Casas, Jesús Pérez, and Miguel A. Otaduy. "Learning Contact Corrections for Handle-Based Subspace Dynamics". *ACM Transactions on Graphics (SIGGRAPH)* (2021)

To facilitate the understanding of the whole, in this and the remaining chapters, we have adapted the mathematical notation with respect to the separately published papers. In these last three chapters there is a strong connection in the problems we solve and how we solve them. To make the whole more cohesive and allow a smoother reading, we have unified the notation following some simple rules. We use lowercase fonts for scalar and vector quantities, and uppercase fonts for matrices. On the other hand, for scalar and 3D geometric quantities (positions, displacement vectors, transformations, ...) we use regular fonts, and for vectors and matrices of arbitrary dimensions with non-geometrical nature, we use bold fonts instead. The remaining ambiguities that may arise can be easily decided by context.

4.1 Introduction

Subspace simulation models define a compact space for the animation of complex objects, without the constraints of mesh resolution. They have demonstrated the ability to produce expressive simulations under low computational cost [PW89; KLM01; OHS03; BJ05; AKJ08]. They are not free of limitations though, as they suffer to produce high-frequency details, e.g., resulting from contact.

Learning methods find effective parameterizations to model complex nonlinear functions. Despite recent important breakthroughs [Bat*16; Cha*17; San*18; GDY19; WBT19; Li*19], it is yet unclear whether learning approaches can represent the generality of high-resolution dynamics under contact. However, they have succeeded at capturing high-resolution skeletal and rig-based animations [SOC19; San*20; PLP20; SSR20].

In our work, we want to combine the best features of subspace and learning models for dynamic simulation of deformable objects. To do so, we design a new subspace simulation model, presented in Section 4.2. The model aggregates a linear global deformation and a nonlinear local correction, both parameterized by a common set of subspace handle-based degrees of freedom. We also define an efficient mapping of the nonlinear corrections to the global setting, as a function of the subspace. Thanks to learning-based modeling of the local corrections, we attain highly detailed and accurate contact-driven deformations. At the same time, thanks to the subspace parameterization of the aggregate deformation, we attain fast dynamics and overall contact-based interaction. Combining the features of subspace and learning models, we achieve dynamic simulations with unprecedented combination of speed and contact-driven deformation detail.

Our method captures in a consistent way nonlinear corrections due to both internal deformations and external interactions. We have designed a data-generation and training pipeline that supports different types of corrections and interactions. As we describe in Section 4.3, this pipeline requires mapping interactions and full-space deformations to the linear subspace, and we discuss how the choice of subspace can simplify this task.

All in all, we introduce a simple method that allows the efficient simulation of many interesting phenomena. We showcase simulations where dynamics are efficiently resolved in a subspace, and they are enriched with accurate data-driven quasi-static corrections. As the human eye is less perceptive of detail under motion, and high-frequency oscillations tend to dampen quicker than low-frequency oscillations, we find that our approach produces simulations that are barely distinguishable from full-space results. We also tackle the simulation of local contact deformations, a classic challenge for subspace methods. We

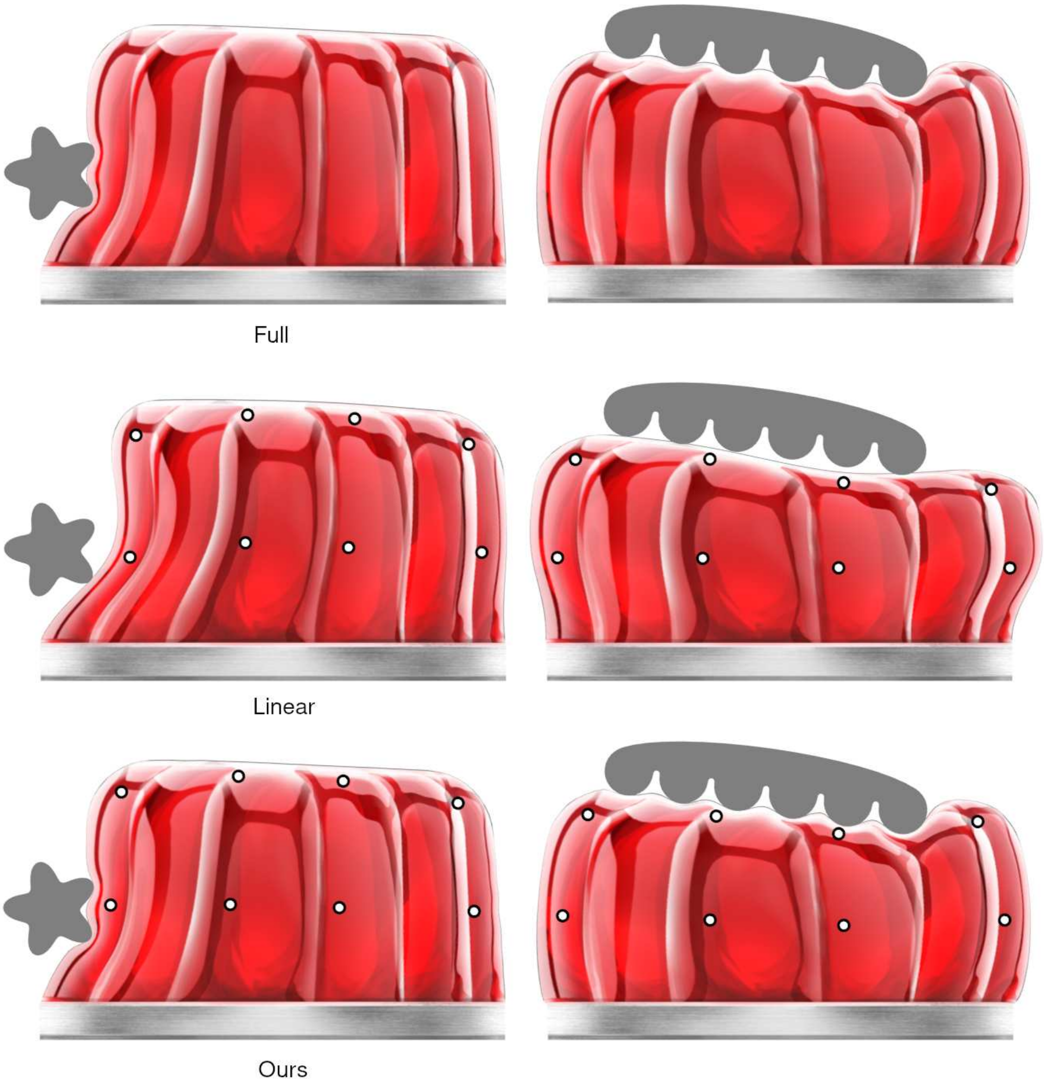


Figure 4.1: The top images show a dynamic simulation of an FEM Neo-Hookean jelly with 12,469 triangles. The deformation is rich but slow (20 fps). The central images show the same scene using a linear subspace model built with just 8 point handles. The simulation is fast (420 fps), but it misses all the detail and suffers distortion under moderate forces. The bottom images show the result with our model, which augments the linear model with nonlinear learning-based corrections. We retain fast dynamics close to the linear model (140 fps), but we recover the detailed contact-driven deformations of the full model.

showcase simulations where we learn these high-detail deformations as a function of the relative configuration between colliding objects and the subspace, and seamlessly aggregate the corrections with the subspace dynamics. We demonstrate the method on examples that signify its applicability, such as the simulation of microtextures, soft robots, or soft skeletal bodies, as shown in Fig. 4.1 and throughout the chapter.

4.2 Corrected Subspace Deformations

In this section, we introduce our subspace deformation model. We start by formulating the combination of a linear subspace model, nonlinear local corrections, and the mapping of these corrections to the global setting. All these components are parameterized by the same reduced handle-based degrees of freedom (DoFs). To allow the computation of forces and velocities, we also derive the Jacobian of our aggregate subspace model, and we analyze computationally efficient approximations. We conclude by discussing the application of variational solvers for dynamic and static deformations.

4.2.1 Formulation of the Subspace

As outlined in the introduction, we construct the subspace model as the addition of a reduced-order linear deformation and a nonlinear local correction. This separation is shown in Fig. 4.2. For the linear deformation, we choose the biharmonic generalized barycentric coordinates (BGBC) [Wan*15]. BGBC allow an intuitive definition of the linear subspace basis, formed by the transformations of points and rigid frames (referred to as *handles*), and we leverage this intuitive basis to construct compact parameterizations of the corrections in Section 4.3. Other choices of frame-based models [BEH18; Gil*11] would also be suited for the definition of the subspace. With handle DoFs $\mathbf{q}$ and BGBC basis $\mathbf{U}$, the linear portion of our subspace model is $\mathbf{U}\,\mathbf{q}$. To extend the accuracy of the linear handle-based subspace, we construct a nonlinear correction. In Section 4.3 we will discuss the details of this correction; for now we consider a general correction with nonlinear dependency on the reduced DoFs, $\mathbf{r}(\mathbf{q})$. Similar to pose-space deformation [LCF00], we express the correction in a local setting. This choice simplifies learning and hence maximizes the accuracy of the correction field, as shown in Fig. 4.3.

We map the local corrections to the full space using the deformation gradient $\mathbf{F}(\mathbf{q})$ of the linear subspace deformation. Notice that this mapping corresponds to a first-order

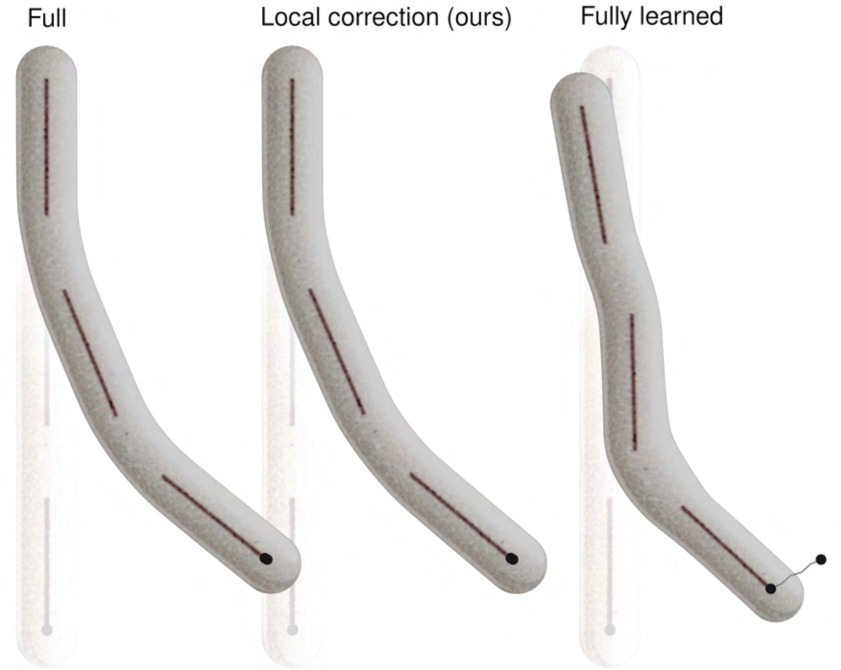


Figure 4.2: Our subspace model (center) disentangles the deformations due to three different sources (global linear, local nonlinear internal, local nonlinear external), enabling an efficient learning of nonlinear corrections, and accurate matching of full simulations (left). Directly learning the full deformation, on the other hand, leads to poor generalization capability (right). In the example, the subspace model is made of three bones, and deformations are produced by pulling with a spring from the circle at the bottom. Both our model and the fully learned approach use neural networks of the same complexity.

approximation of a correction applied to the undeformed setting [Mal*15]. Adding the linear and nonlinear components together, we can express our nonlinear subspace deformation model as:

$$\mathbf{x}(\mathbf{q}) = \mathbf{U}\,\mathbf{q} + \mathbf{F}(\mathbf{q})\,\mathbf{r}(\mathbf{q}). \tag{4.1}$$

In practice, we compute the deformation gradient on tetrahedral elements [ITF04] and then perform a moving least-squares approximation on nodes [Mül*04]. In the remainder of the section, we drop the explicit dependency of $\mathbf{q}$ from the various terms in (4.1).

Interestingly, the nonlinear correction to the linear subspace deformation can also be interpreted as a modulation of the linear basis. To this end, we rewrite (4.1) by reversing the order of $\mathbf{F}\,\mathbf{r}$ as mat$(\mathbf{r})$ vec$(\mathbf{F})$. Furthermore, the deformation gradient can be expressed through a linear operation mat$(\boldsymbol{\nabla})$ on the subspace deformation $\mathbf{U}\,\mathbf{q}$. Using the matrices mat$(\mathbf{r})$ and mat$(\boldsymbol{\nabla})$, we rewrite our corrected subspace model (4.1) as

$$\mathbf{x} = (\mathbf{I} + \mathbf{W})\ \mathbf{U}\,\mathbf{q}, \tag{4.2}$$

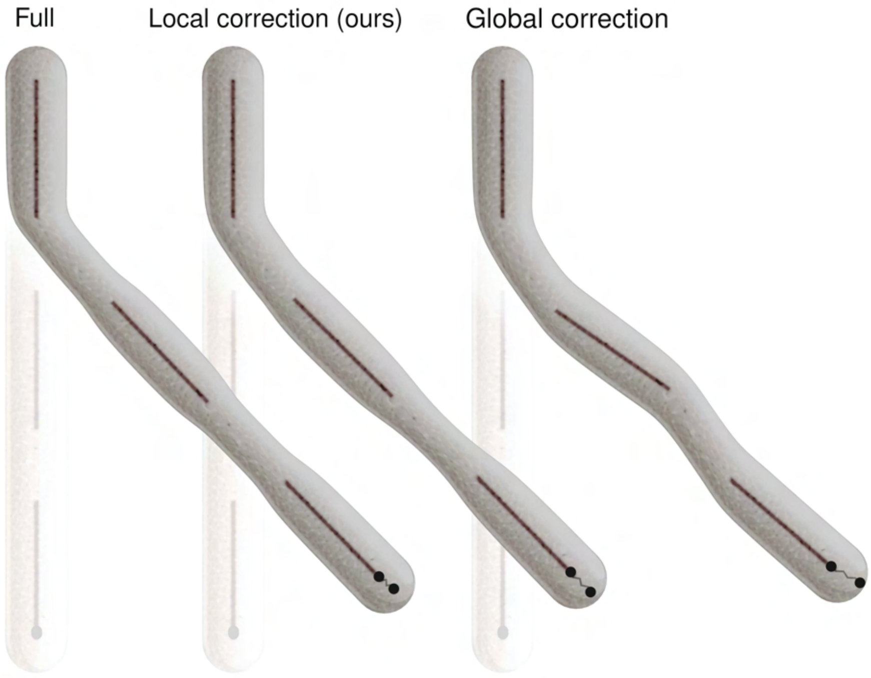


Figure 4.3: The deformation behavior of a full simulation (left) is accurately modeled when nonlinear corrections are learned on a local setting (center). Global corrections are more difficult to learn, and suffer artifacts (right). In the example, both local and global corrections use the same training data and neural-network architecture.

with $\mathbf{W} = \text{mat}(\mathbf{r})\,\text{mat}(\boldsymbol{\nabla})$. As evidenced in this expression, the nonlinear correction can be interpreted as an incremental modulation $\mathbf{W}\,\mathbf{U}$ to the linear subspace basis $\mathbf{U}$. This modulation weights the gradient of the subspace basis by the corrections. In practice, for the evaluation of full-space positions $\mathbf{x}$, we use (4.1), after computing $\mathbf{F}$ explicitly. For the transformation of forces to the subspace, however, it is convenient to analyze the basis modulation (4.2), as we see next.

4.2.2 Jacobian of Subspace Kinematics

A key ingredient of the subspace model is the Jacobian $\mathbf{J}$ that linearizes the mapping between the subspace DoFs $\mathbf{q}$ and the full-space deformation $\mathbf{x}$. Differentiating (4.1) and (4.2), we obtain:

$$\mathbf{J} = \frac{\partial \mathbf{x}}{\partial \mathbf{q}} = (\mathbf{I} + \mathbf{W})\ \mathbf{U} + \mathbf{F}\,\frac{\partial \mathbf{r}}{\partial \mathbf{q}}. \tag{4.3}$$

Using this Jacobian, one can transform subspace velocities $\dot{\mathbf{q}}$ to the full-space as $\dot{\mathbf{x}} = \mathbf{J}\,\dot{\mathbf{q}}$, and full-space forces $\mathbf{f}_{\mathbf{x}}$ to the subspace as $\mathbf{f}_{\mathbf{q}} = \mathbf{J}^T\,\mathbf{f}_{\mathbf{x}}$.

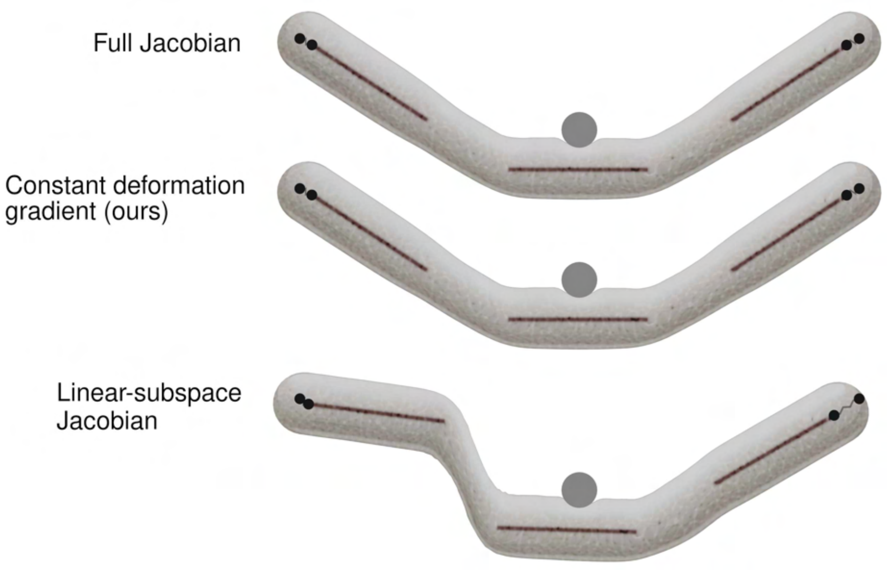


Figure 4.4: To maximize runtime efficiency, we have evaluated different approximations to the Jacobian of our deformation model (4.3). The behavior with the full Jacobian (top) is accurately matched when we ignore the change in the deformation gradient (middle), as in (4.4). However, deformation errors are evident (bottom) if we use the Jacobian of the linear subspace and ignore the change in the corrections $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}$; hence we retain this term.

Most of the computational overhead of our subspace corrections lies in the evaluation of the Jacobian. Therefore, we pay attention to the relevance of the terms $\mathbf{W}$ and $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}$ in (4.3). Fig. 4.4 shows a representative example where we evaluate different approximations of $\mathbf{J}$. We have observed that the term $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}$ bears an important role in the computation of forces and resulting deformations, hence it should not be ignored.

On the other hand, the term $\mathbf{W}$, which carries the Jacobian of the deformation gradient, can be safely discarded in the computation of forces. This is no surprise; as subspace deformations are smooth, the Jacobian of their deformation gradient is comparatively small. Dropping this term can be paralleled to ignoring the derivative of rotations in corotational elasticity models [MG04; Xu*15], but the effect is even milder for subspace deformations.

Based on our experiments, we conclude to approximate the Jacobian (4.3) as

$$\mathbf{J} \approx \mathbf{U} + \mathbf{F}\,\frac{\partial \mathbf{r}}{\partial \mathbf{q}}. \tag{4.4}$$

We have also experimented with using this Jacobian for force computations and the approximation $\mathbf{J} \approx \mathbf{U}$ to build the Hessian. However, this approximation results in excessive damping and slows down the convergence of Newton solves.

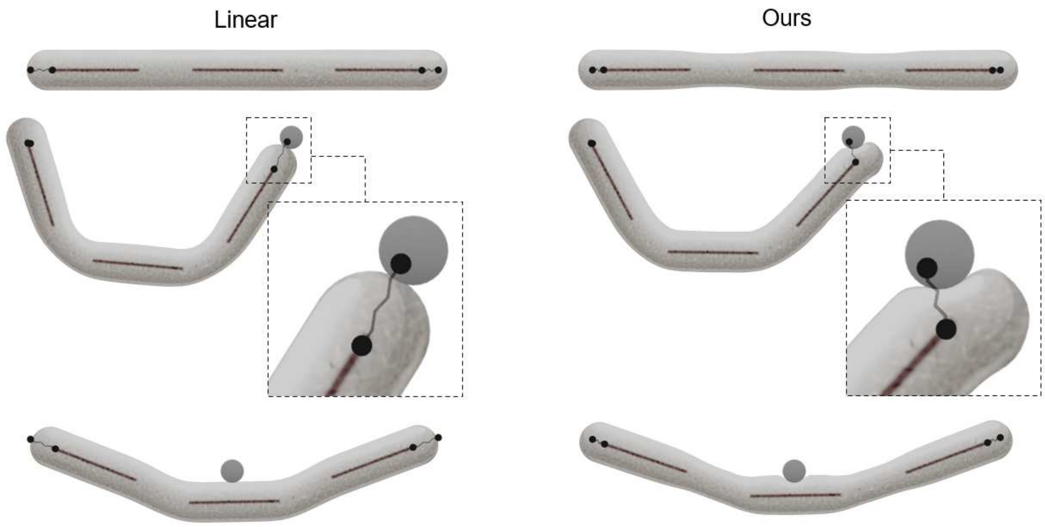


Figure 4.5: This example highlights the aggregation of deformations in our model. The left column shows the linear deformation $\mathbf{U}\,\mathbf{q}$. The right column shows the addition of nonlinear corrections. The top-right image includes only internal corrections $\mathbf{r}_{\text{int}}$, which restore nonlinear deformations. The middle-right and bottom-right images include both internal and external corrections, with the middle-right example highlighting external corrections $\mathbf{r}_{\text{ext}}$, which introduce accurate contact-driven details.

4.2.3 Dynamics and Integration

In our examples, we show both dynamic and (quasi-)static deformations. For a unified solution to both types of simulations, we use a variational formulation of backward Euler integration [Mar*11; Gas*15]. As done by Pan et al. [PBH15], the variational form of the subspace integration is easily formulated by expressing the objective function in the full space, with the subspace DoFs $\mathbf{q}$ as search variables. With an explicit update $\mathbf{x}^* = \mathbf{x}_{\text{old}} + h\,\dot{\mathbf{x}}_{\text{old}}$ of the full-space positions and time step h, time integration results in

$$\mathbf{q} = \arg\min \frac{1}{2\,h^2}\,(\mathbf{x}-\mathbf{x}^*)^T\,\mathbf{M}\,(\mathbf{x}-\mathbf{x}^*) + V(\mathbf{x}). \tag{4.5}$$

To time-step the rigid frames in the BGBC reduced DoFs $\mathbf{q}$, we parameterize the rotations in their tangent-space [TK94]. $\mathbf{M}$ denotes the full-space mass matrix and V the potential energy. Full-space forces are defined as $\mathbf{f}_{\mathbf{x}} = -\nabla V$. Our work admits general elasticity models and discretizations for the definition of full-space forces. In our examples, we have used a Neo-Hookean material [SGK18] with tetrahedral FEM discretization.

The optimality of (4.5) yields the following nonlinear equations:

$$\mathbf{J}^T\,\frac{1}{h^2}\,\mathbf{M}\,(\mathbf{x}-\mathbf{x}^*) - \mathbf{J}^T\,\mathbf{f}_{\mathbf{x}} = 0. \tag{4.6}$$

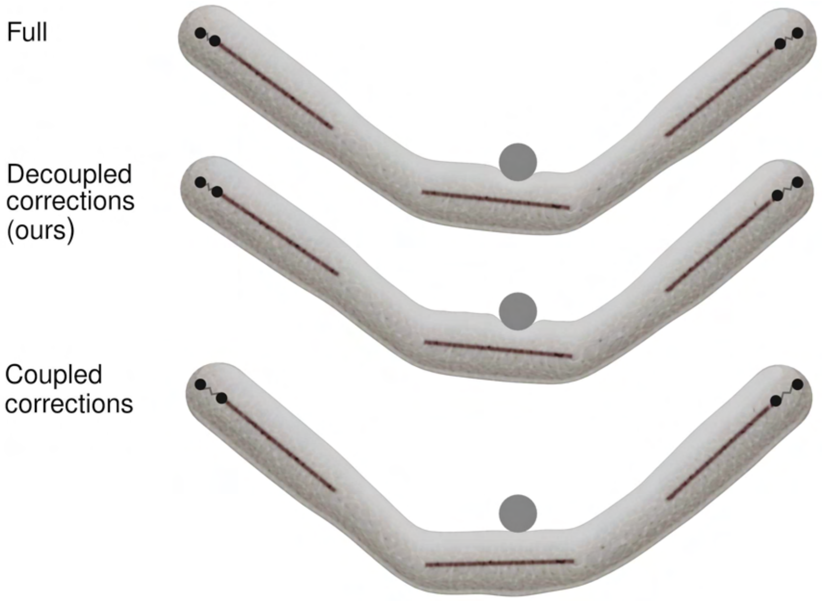


Figure 4.6: The nonlinear deformation of a full simulation (top) is accurately matched when internal and external corrections are learned separately (center). Trying to learn both types of corrections together complicates data generation and learning, and fails to reproduce external contact-driven corrections (bottom). In the example, the complexity of the neural-network architecture for coupled learning is equal to the added complexity of the decoupled architectures.

We solve these equations using a quasi-Newton method, where we approximate the Hessian of (4.5) as $\mathbf{J}^T \left(\frac{1}{h^2}\,\mathbf{M} - \frac{\partial \mathbf{f_x}}{\partial \mathbf{x}} \right) \mathbf{J}$.

As done often for subspace methods, we use a cubature approximation of forces and Hessians [AKJ08]. After training cubature points $\{x_k\}$ and weights $\{w_k\}$ [Tyc*13], one can approximate subspace forces (and similarly their Jacobian) as $\mathbf{f_q} \approx \sum_k w_k\, \mathbf{J}_k^T\, \mathbf{f}_{\mathbf{x},k}$, where $\mathbf{f}_{\mathbf{x},k}$ and $\mathbf{J}_k$ are, respectively, the force and the Jacobian at the cubature point. In our implementation, we use the same cubature approximation to project the mass matrix $\mathbf{M}$ to the subspace.

4.3 Learning Corrections

In a fully dynamic setting, the deformation of an object depends on its velocity, acceleration, and external forces. We represent dynamics in the linear subspace, but we want to retain the accuracy of nonlinear (quasi-)static deformations. We consider two sources of error in the linear subspace, and therefore model two separate corrections: internal corrections $\mathbf{r}_{\text{int}}$, which correct the linear subspace deformation in the absence of contact, and external cor-

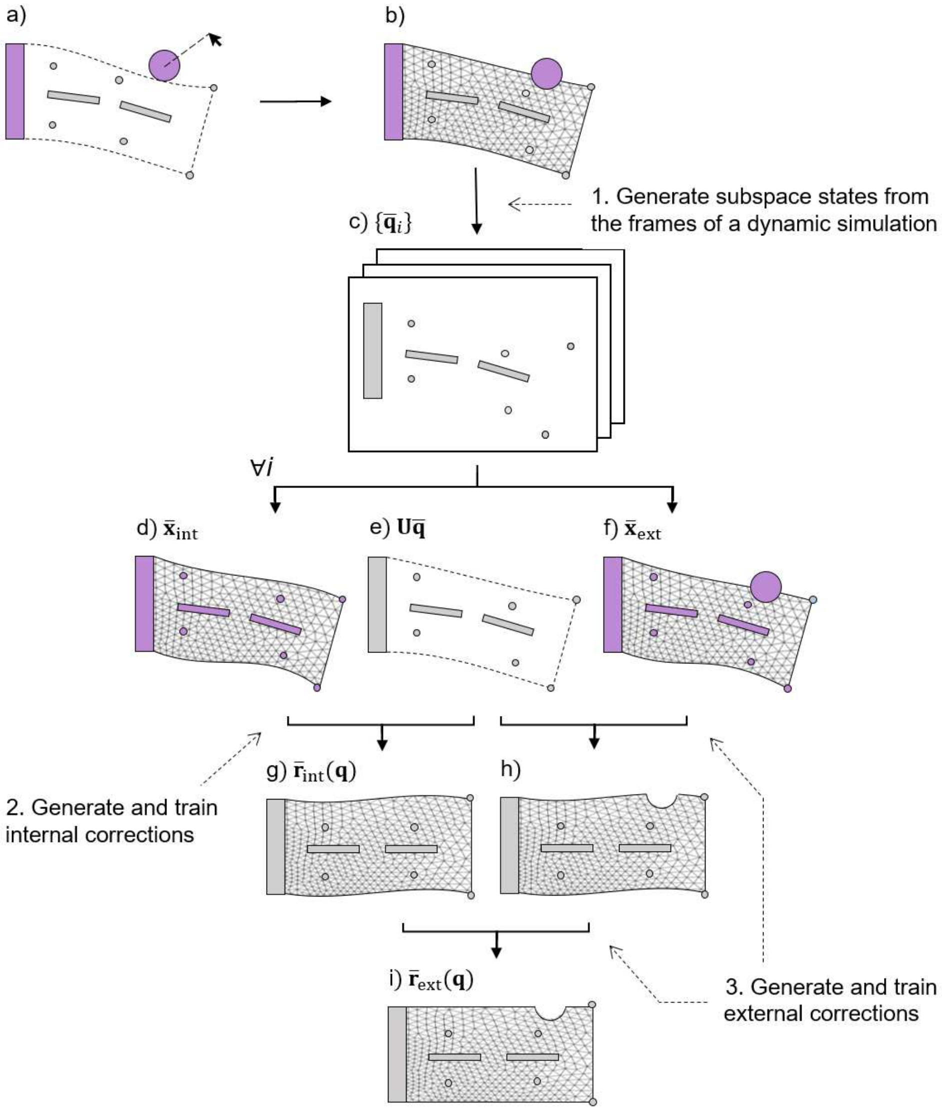

a)
b)
1. Generate subspace states from the frames of a dynamic simulation
c) $\{\bar{\mathbf{q}}_i\}$
$\forall i$
d) $\bar{\mathbf{x}}_{\text{int}}$
e) $\mathbf{U}\bar{\mathbf{q}}$
f) $\bar{\mathbf{x}}_{\text{ext}}$
2. Generate and train internal corrections
g) $\bar{\mathbf{r}}_{\text{int}}(\mathbf{q})$
h)
i) $\bar{\mathbf{r}}_{\text{ext}}(\mathbf{q})$
3. Generate and train external corrections

Figure 4.7: Data generation pipeline. First, a) we interactively record a linear-subspace dynamic simulation, and b) use the recorded interaction to generate an offline full dynamic simulation. c) For each frame, we extract a representative subspace state $\bar{\mathbf{q}}$. Then, we fix the DoFs corresponding to the subspace (in purple) and run two full static simulations, d) ignoring and f) including, external interactions. Nonlinear corrections are then computed by considering the difference between these full static deformations and the linear subspace solution $\mathbf{U}\bar{\mathbf{q}}$ in e). Internal corrections are generated by g) mapping the difference to the undeformed setting using $\mathbf{F}^{-1}$. Finally, external corrections are generated in two steps: first, h) the difference w.r.t. the linear subspace solution is again mapped to the undeformed setting; and second, i) internal corrections are substracted to account only for the effect of external interactions.

rections $\mathbf{r}_{\text{ext}}$, which correct the additional deviation introduced by contact. This separation into internal and external corrections, highlighted in Fig. 4.5, simplifies the generation of representative training data, and hence maximizes the accuracy of the aggregate correction, as shown in Fig. 4.6.

We start this section with a detailed definition and formulation of the internal and external corrections. Then, we describe the generation of training data for both types of corrections, following the pipeline outlined in Fig. 4.7. And we conclude with a discussion of implementation details of the learning architecture.

4.3.1 Internal and External Corrections

Given a subspace state $\mathbf{q}$ and constant external forces (i.e., gravity), but no other external interactions, the internal corrections $\mathbf{r}_{\text{int}}$ represent the deviation between the full-space equilibrium deformation and the full-space positions given by the linear subspace, $\mathbf{U}\,\mathbf{q}$. On the other hand, given a subspace state $\mathbf{q}$ and an external interaction state, the external corrections $\mathbf{r}_{\text{ext}}$ represent the deviation between the full-space equilibrium deformation and the full-space positions given by the internally corrected subspace, $\mathbf{U}\,\mathbf{q} + \mathbf{F}\,\mathbf{r}_{\text{int}}$. Fig. 4.5 demonstrates the aggregation of internal and external corrections.

We have considered external interactions due to kinematic colliders, but the formulation could be extended to other types of interactions. Note that interactions produced by prescribing some subspace DoFs $\mathbf{q}$ (e.g., moving handles of the subspace model) can be represented as part of internal corrections. We denote the interaction state as $\mathbf{z}$, which in our case may include the state and size of rigid colliders. For better learning ability, we parameterize the corrections expressing the interaction state relative to the subspace state $\mathbf{q}$. Here, handle-based reduced models such as BGBC come handy. We can define rigid transformations $\mathbf{A}(\mathbf{q})$ for the handles, and invert them to define relative external interactions $\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z}$.

Formally, our nonlinear correction is then split into internal and external corrections as:

$$\mathbf{r} = \mathbf{r}_{\text{int}}(\mathbf{q}) + \mathbf{r}_{\text{ext}}(\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z}). \tag{4.7}$$

By separating internal and external corrections, we avoid the combinatorial complexity of training for all possible internal and external interaction states. We can train internal corrections free of external interactions, and we can train external interactions only in the vicinity of the deformable object. Next, we describe our data generation pipeline.

Table 4.1: Model size and performance data of the examples shown in the chapter. For the worm, we show data with 1 and 6 colliders. Note that in both cases the corrections are trained with just 1 collider.

Example	Jelly + Circle	Jelly + Comb	Jelly + Star	Accordion	Auxetic	Worm	Bunny	Finger
Handles (points/frames)	8/1	8/1	8/1	16/2	16/2	0/3	24/1	0/4
Colliders	1	1 (rot.)	1 (rot.)	0	0	1 (×6)	1	1
Full mesh (tris or tets)	12,469	12,469	12,469	17,457	12,921	10,656	17,062	10,163
Cubature points	599	599	599	836	615	505	341	203
PCA corr. (int/ext)	-/50	-/100	-/100	15/-	15/-	15/120	-/120	-/120
Neurons (int/ext)	-/200	-/3,000	-/3,000	200/-	200/-	200/2,000	-/1,500	-/2,000
Train frames (int/ext)	-/9,747	-/187,039	-/270,000	5,880/-	6,291/-	3,730/138,379	-/17,425	-/25,128
Ours fps	201	135	145	85	99	13.6 (6.9)	48	10.9
Linear fps	444	410	432	119	122	403	87	143
Full fps	22.7	19.0	21.2	10.4	11.7	2.5	7.1	1.9

4.3.2 Data Generation

The generation of training data follows a strategy parallel to the decoupling of internal and external corrections. We visit separately (i) the configuration space of the deformable object, and (ii) the relative configuration space of the collider. For (i), as the space is very large and difficult to predict, we follow a user-guided sampling approach [BJ05]. For (ii), we follow an automated sampling approach, and traverse with the collider the surface of the deformable object on the configurations obtained in (i). Our decoupled sampling of (i) and (ii) is beneficial in two ways: it removes the need to explore (i) and (ii) together, which is hard even through user interaction, and it naturally produces training data to separately learn internal and external corrections. Based on this decoupling, the data generation pipeline proceeds in three steps: generation of representative states, generation and training of internal corrections, and generation and training of external corrections. A detailed representation of the data generation pipeline is shown in Fig. 4.7.

To generate representative states, we first execute fast dynamic simulations using the baseline linear subspace model (Fig. 4.7, a). These simulations are interactive in our examples, and one can move colliders and apply forces to quickly visit a large number of states. Next, we replay the same interactions, but we simulate deformations using the full-space model (Fig. 4.7, b). For each frame of these simulations, we project the full-space positions $\mathbf{x}$ to the subspace, using a least-squares mapping $\mathbf{q} = (\mathbf{U}^T\,\mathbf{U})^{-1}\,\mathbf{U}^T\,\mathbf{x}$. This projection yields a set of representative subspace states $\{\bar{\mathbf{q}}_i\}$ and the corresponding full-space positions $\{\mathbf{U}\bar{\mathbf{q}}_i\}$ (Fig. 4.7, c and e).

To generate internal correction targets, we must remove the effect of dynamics and external interactions from the full-space deformations described above, but leaving the subspace state unchanged. To this end, we compute constrained static deformations (Fig. 4.7, d). For each representative subspace state $\bar{\mathbf{q}}_i$, we compute the static full-space deformation $\bar{\mathbf{x}}_{\text{int},i}$, such that it is constrained to the given subspace state. With our choice of handle-based subspace model, enforcing the constraints is as simple as fixing the full-space DoFs corresponding to the handles. From the subspace and full states, we obtain target internal corrections simply by undoing our nonlinear subspace formulation (4.1):

$$\bar{\mathbf{r}}_{\text{int},i}(\bar{\mathbf{q}}_i) = \mathbf{F}(\bar{\mathbf{q}}_i)^{-1}\,(\bar{\mathbf{x}}_{\text{int},i} - \mathbf{U}\,\bar{\mathbf{q}}_i)\,. \tag{4.8}$$

At this point, we use these internal correction targets to train the internal correction $\mathbf{r}_{\text{int}}(\mathbf{q})$ (Fig. 4.7, g).

To generate external correction targets, we need to reintroduce the effect of external interactions on the representative subspace states. For each representative subspace state, we generate multiple interaction states, traversing with the collider the surface of the deformable object at varying depths. Without loss of generality, in the remainder we refer to one pair of subspace and interaction states. Given an interaction $\mathbf{z}_i$, we compute the static full-space deformation $\bar{\mathbf{x}}_{\text{ext},i}$ that is constrained to a given subspace state $\bar{\mathbf{q}}_i$ (Fig. 4.7, f). From the subspace and full states, we obtain target external corrections simply by undoing the subspace formulation (4.1) (Fig. 4.7, h). However, this time we also subtract the internal corrections:

$$\bar{\mathbf{r}}_{\text{ext},i}(\mathbf{A}(\bar{\mathbf{q}}_i)^{-1}\,\mathbf{z}_i) = \mathbf{F}(\bar{\mathbf{q}}_i)^{-1}\,(\bar{\mathbf{x}}_{\text{ext},i} - \mathbf{U}\,\bar{\mathbf{q}}_i) - \bar{\mathbf{r}}_{\text{int},i}(\bar{\mathbf{q}}_i). \tag{4.9}$$

At this point, we use these external correction targets to train the external correction $\mathbf{r}_{\text{ext}}(\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z})$ (Fig. 4.7, i).

4.3.3 Learning Architecture and Training

We learn separate models for internal and external corrections, but we follow the same methodology for both. Therefore, in this section we refer to arbitrary corrections $\mathbf{r}$. We have observed that corrections exhibit high coherence, hence we use principal component analysis (PCA) to reduce their dimensionality.

We use a fully connected, 2-layer neural network to model each type of nonlinear correction. For internal corrections, the input is the subspace state $\mathbf{q}$, and for external corrections, the input is the relative interaction state $\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z}$, as shown in (4.7). In both cases, the output of the network is the PCA representation of the corrections. We use `tanh` as activation function, and we have implemented the neural networks using PyTorch.

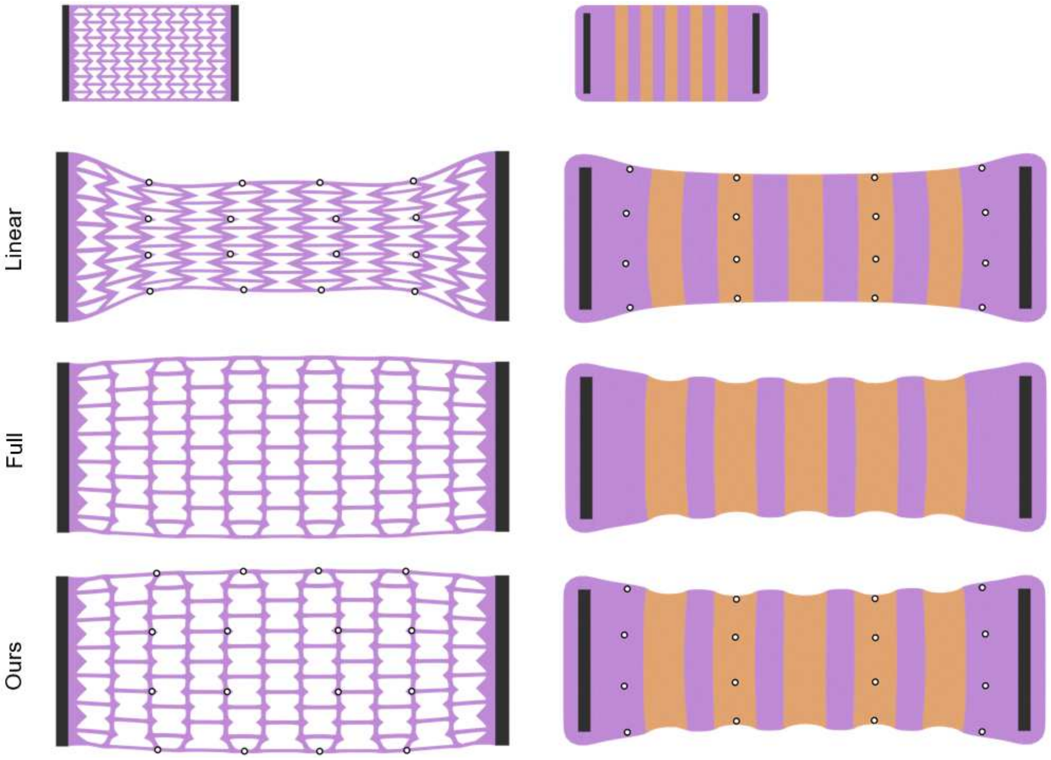


Figure 4.8: We simulate two types of microstructures, an auxetic structure (left) and accordion-like heterogeneous stripes (right), with subspace models defined by just 2 frames and 16 points. A purely linear model is incapable of showing nonlinear effects produced by material heterogeneity, such as the negative Poisson's ratio of the auxetic structure and the ripples of the striped structure. Our method practically matches the full solution, yet $9\times$ faster.

We use as training data the target corrections discussed in the previous section, together with their corresponding subspace and interaction states. We use as loss function the L^2 norm of the difference between target and estimated corrections, and we optimize the networks using Adam, 1000 to 2000 epochs, a batch size of 512, and learning rate of 1e-3. As done typically in machine learning methods, we separate a random subset of the training data and we use it as test data to monitor the convergence of the optimization of the neural network. This test data is different from that shown in the examples, which is made of completely new interactions, not used during training at all.

At runtime, the neural network is needed for the evaluation of displacements, but also for the transformation of full-space forces and the system Hessian to the subspace, as discussed in Section 4.2.3. Recall that the approximation of the Jacobian of our subspace model (4.3), requires the Jacobian of nonlinear corrections $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}$, as shown in (4.4). We use a matrix-free implementation of a conjugate-gradient solver, which in turn uses products $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}^T \mathbf{v}$ and $\frac{\partial \mathbf{r}}{\partial \mathbf{q}} \mathbf{v}$ with vectors $\mathbf{v}$. Our implementation of the neural network on PyTorch includes gradient back-propagation capabilities, which address the evaluation of $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}^{\mathrm{T}} \mathbf{v}$. For $\frac{\partial \mathbf{r}}{\partial \mathbf{q}} \mathbf{v}$, we do the following. We implement a function $\mathbf{y} = \frac{\partial \mathbf{r}}{\partial \mathbf{q}}^{T} \mathbf{w}$ once per system solve, by back-propagation of an arbitrary vector $\mathbf{w}$ through the network. Then, on each conjugate gradient iteration, we back-propagate the vector $\mathbf{v}$ through the function $\mathbf{y}(\mathbf{w})$ to obtain $\frac{\partial \mathbf{r}}{\partial \mathbf{q}} \mathbf{v} = \frac{\partial \mathbf{y}}{\partial \mathbf{w}}^{T} \mathbf{v}$.

4.4 Experiments

Table 4.1 summarizes the settings, model size, and performance of the examples shown in the chapter. All the examples were executed on an Intel Core i7-7700K 4-core 4.20 GHz PC with 32 GB of RAM. Next, we discuss in detail the different experiments.

Microstructures. The combination of different materials at a microscopic level can produce interesting macroscopic mechanical behaviors. However, the simulation of such microstructures at full resolution yields a very high computational cost. One approach to avoid this cost is to use numerical coarsening methods [Sch*15]. Nevertheless, numerical coarsening methods assume a linear response of the microstructure with respect to coarse DoFs [Kha*09; Tor*16; Che*18].

We have explored the use of our model for the simulation of microstructures, by augmenting the linear subspace model with nonlinear internal corrections. In Fig. 4.8 we show the application of our model to two different microstructures: an auxetic microstructure (top), and heterogeneous accordion-like stripes (bottom). In both cases, the difference with respect to the full-resolution simulation is almost imperceptible. With just 2 rigid handles (controlled by the user) and 16 point handles, both dynamics and detailed static deformations are reproduced very accurately. Notice how the purely linear model misses the fundamental behavior of the auxetic material (negative Poisson's ratio) and the ripples of the accordion-like stripes. Both effects are matched with our learning-based nonlinear corrections.

Jelly. This example (see Fig. 4.1) showcases soft 2D dynamics augmented with data-driven contact. The object is modeled on a subspace defined by just 1 rigid handle (controlled by the user) and 8 point handles, and we learn separately, as two disjoint models, external corrections produced by a comb-like collider which combines both a large contact area and small protrusions, and a star with pointy features. The purely linear subspace model suffers notable distortions and misses detailed contact deformations. Previous methods for local enrichment of subspace models [HZ13; Ten*15] assume a moderate contact area to be efficient, and would not scale well on the comb example. Our model, on the other hand, matches accurately the deformations of the full model, with a performance that comes close to the linear subspace model. Note also that dynamics are well captured in the subspace, i.e., high-resolution dynamics of the full model are quickly damped.

The proposed subspace model succeeds to capture detailed contact-driven deformations, but the challenge to accurately learn these deformations grows with the complexity and configuration space of the collider. In Table 4.2, we compare quantitatively the accuracy of the subspace jelly model for three different colliders: (i) the comb-like collider of Fig. 4.1, which produces a large and complex contact area and has a 3D configuration space (translation and rotation in 2D); (ii) the same comb-like collider but restricted to a 2D configuration space (with no rotation); (iii) and a small circle-like collider, which produces a small contact area and has a 2D configuration space. As summarized in the table, our model learns well the interaction with the small circle even with a small neural network and a small training set. However, as the complexity and configuration space of the collider grow, both the complexity of the neural network and the training set must grow. With small network and training set, the model captures well the global correction to the linear deformation, but fails to learn high-frequency details of the interaction with the complex comb. A qualitative comparison of results is shown in Fig. 4.9. The star collider of Fig. 4.1 also requires a complex network and a large training set, like the comb, due to the size of its configuration space and its pointy features, as indicated in Table 4.1.

Soft-Robot Worm. We have modeled a worm-like soft robot, with three bones surrounded by soft material (See Fig. 4.10). We simulate the worm using just 3 rigid handles, colocated with the bones, and no point handles. Even under such a compact subspace, we show that

Table 4.2: Evaluation of model accuracy as a function of the complexity of the collider and its configuration space, the size of the training data set, and the complexity of the neural network architecture. The benchmark for the comparisons is the jelly object in Fig. 4.1, using as colliders a small circle and a large comb-like object. Accuracy is measured as the RMSE of vertex displacements w.r.t. the linear model across all vertices in the object and all frames of the test data set, normalized by the RMS of the same vertex displacements. See also Fig. 4.9 for a visual comparison of some cases.

Training	Neurons			Neurons			Neurons		
frames	200	1000	3000	200	1000	3000	200	1000	3000
9,748	11%	-	-	24%	22%	-	25%	23%	12%
30,486	-	-	-	15%	14%	-	21%	17%	12%
187,039	-	-	-	-	-	-	15%	13%	10%
	Circle			Comb (no rotation)			Comb		

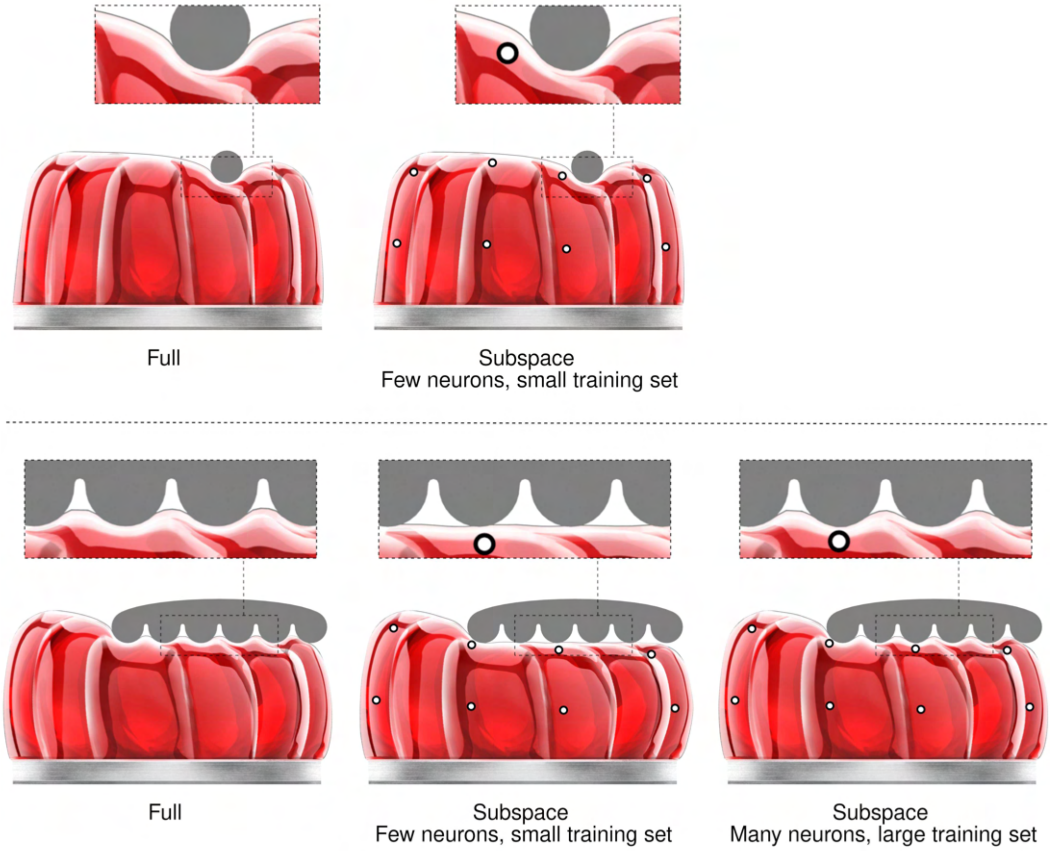

Full
Subspace
Few neurons, small training set
Full
Subspace
Few neurons, small training set
Subspace
Many neurons, large training set

Figure 4.9: Our subspace model successfully represents contact deformations due to both small and large colliders with high-resolution features. Nevertheless, large colliders with larger configuration space (e.g., the comb-like object on the bottom) require a larger training set and larger network architecture. A quantitative analysis of the error is summarized in Table 4.2.

our corrected model succeeds to match the dynamic and contact-driven deformations of a full simulation.

We produce training data by pulling with springs from the bones, and interacting with just one circular pin. Note that we use up to six pins in one example at runtime, as discussed below. We train internal and external corrections, following the procedure described in Section 4.3. Fig. 4.5 showcases both the internal and external corrections during spontaneous interactions outside the training data. Internal corrections are most evident in the soft regions between the bones. Conversely, the purely linear model suffers evident locking, and as a result it cannot stretch as it should. External corrections are most evident at the head of the worm. Conversely, the head of the purely linear model remains locally rigid.

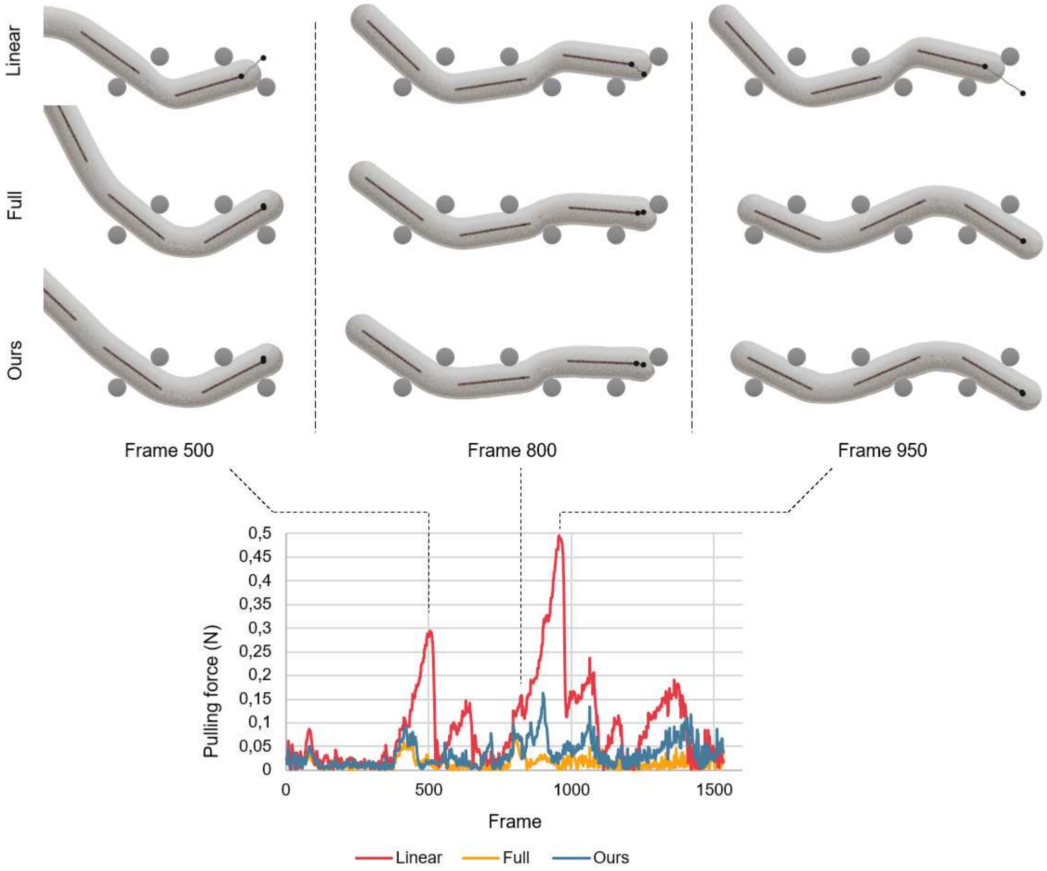


Figure 4.10: In this example, we pull a worm-like soft robot through a narrow passage. A purely linear subspace model (top) suffers strong distortions (see the soft regions between bones), and cannot deform locally to conform to the shape of the pins. Our model (bottom), even though it is built from a subspace of just 3 bones, follows closely the motion and deformations of a full model (middle). The plot shows the pulling force as each worm traverses the passage. The purely linear model suffers locking and reaches a peak force $5.6\times$ larger than the full model. With our model, the peak force is just $1.8\times$ larger. For this benchmark, we trained our external corrections for just one pin. At runtime, we evaluated the same function of external corrections six times, for each pin in the passage. Thanks to the separation of internal and external corrections in our model, external corrections are local in practice, and we can apply superposition of multiple external corrections as long as the colliders are sufficiently far from each other.

We also test the worm model on a more complex setting, well outside the training settings. Fig. 4.10 shows the worm being pulled through a narrow passage, where it collides against six pins. By modeling external corrections separately from internal corrections, their effect is mostly local. Then, if multiple colliders act sufficiently far from each other, we can safely assume superposition of their effects. Therefore, in this example, we train with just one pin, but we run the simulation with six pins, reusing six times the same neural network of external corrections. As we pull the worm through the passage, we monitor the necessary

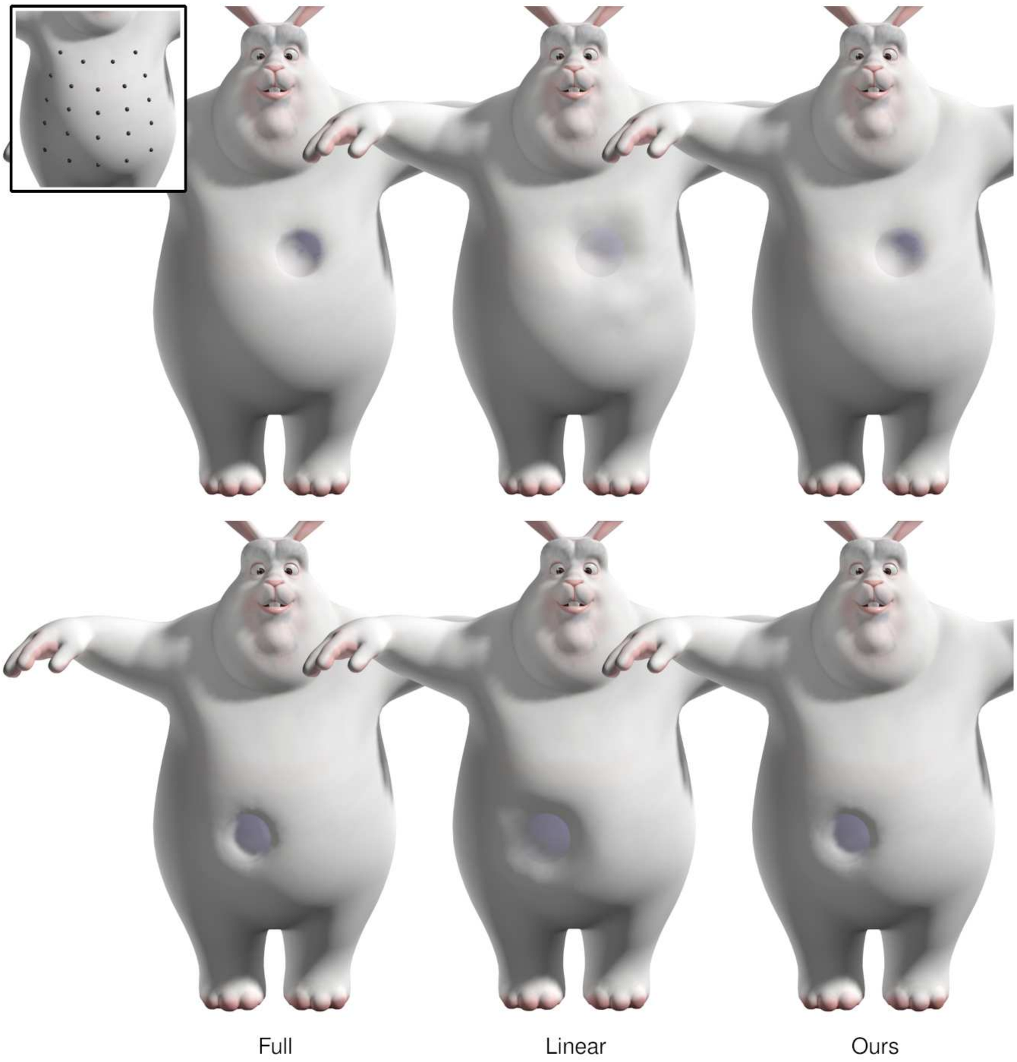


Figure 4.11: This model of Big Buck Bunny contains a soft-tissue layer on top of a rigid core. We learn contact-driven corrections to augment a linear subspace model (point frames highlighted in the inset). As shown in the examples, with our method contact-driven deformations do not suffer the resolution limitations of the linear model, and match closely the deformations of a full simulation model.

pulling force. With our model, the peak force is $1.8\times$ larger than with the full model. With the purely linear model, however, the peak force is $5.6\times$ larger. The linear model suffers strong distortions in the regions between bones, and cannot deform locally to conform to the shape of the pins. Our model does not suffer any of these limitations, and hence the force and overall motion are closer to the full model.

In our model, we decouple different sources of deformation, as we hypothesize that this explicit disentanglement simplifies learning and provides higher accuracy. To validate this hypothesis, we try to learn a fully nonlinear subspace model for the worm, on a simple example with no contact, as shown in Fig. 4.2. We use the same DoFs as in our model, i.e., the rigid transformations of the bones, and we use a neural network with the same complexity. To define the output of the model, we run PCA on the full positions of the training data. With our subspace model, the RMSE of vertex displacements w.r.t. the linear model across all vertices in the worm object and all frames of the test data set, normalized by the RMS of those same vertex displacements, is just 15%. With the fully learned model this error grows beyond $4,000\%$. Learning corrections on a global frame is not sufficient, and the error remains high at 75%, as depicted in Fig. 4.3. We cannot claim that it is not possible to learn full deformations directly; in fact Holden et al. [Hol*19] managed to learn full deformations, albeit with a more complex neural network, and with poor generalization and overdamped dynamics. Nevertheless, we confirm that our explicit disentanglement simplifies the problem. The decoupling of internal and external corrections is also critical for the accuracy of our model. Fig. 4.6 shows a different comparison, this time including contact, of our model vs. a model of the same total network complexity with coupled learning of internal and external corrections. In this comparison, the normalized RMSE with our model is 19%, and grows to 51% with coupled corrections.

Bunny. In Fig. 4.11 we show how we use our model to augment a linear subspace model of Big Buck Bunny with data-driven contact deformations.

The model contains a rigid core surrounded by a soft layer, and the linear subspace model is built using the inner rigid core as a frame handle, together with 24 point handles on the outer surface. We train external corrections due to contact with a spherical collider, following the pipeline described in Section 4.3.2. In the test simulation, it becomes apparent that the linear subspace model fails to produce correct contact deformations, as the point handles are too sparse. Our model, on the other hand, succeeds at producing contact deformations very close to those of the full model.

Finger. To conclude, we have also used our model to simulate deformations of a soft skeletal finger model, shown in Fig. 4.12. The finger is modeled with rigid anthropomorphic phalanges, surrounded by homogeneous soft tissue. We have built the linear model using just the 3 moving phalanges and the fixed palm as rigid handles, with no point handles. We fix the pose of the finger with springs, and as a result the change of finger pose in the example is small. Therefore, we have opted not to model internal corrections, and we have trained external corrections on a fixed pose of the phalanges. The results demonstrate that our approach produces an extremely compact subspace model, with contact-driven deformations that are far from the resolution that can be achieved with the purely linear

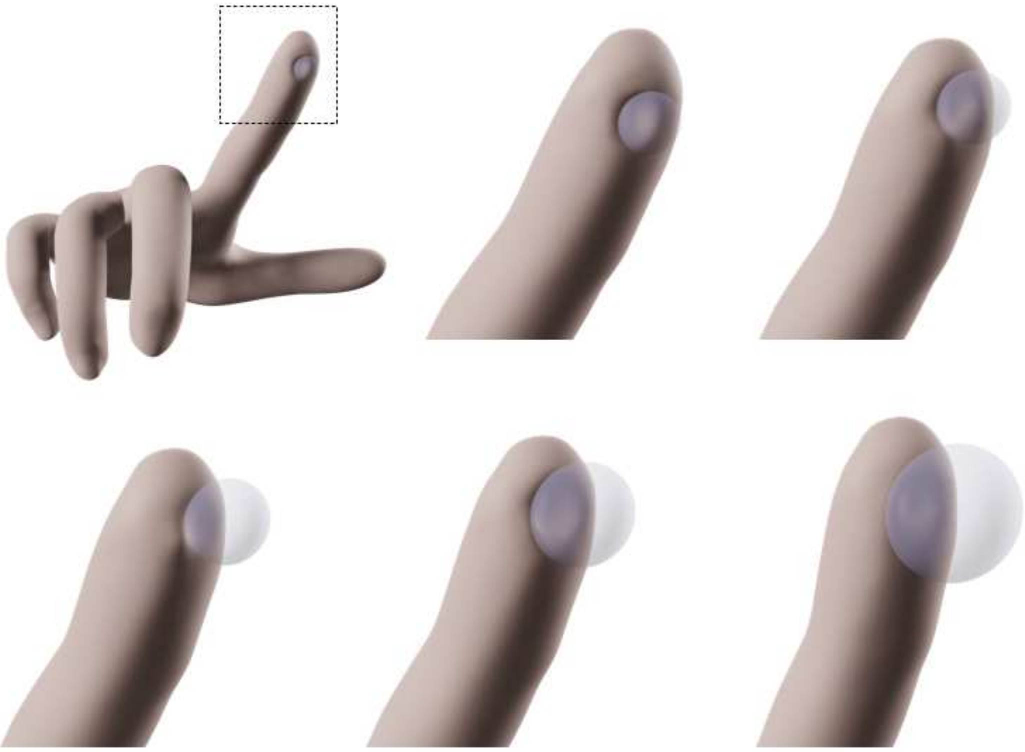


Figure 4.12: We model a finger with just 3 frame handles located at the phalanges. The full nonlinear deformation of the surrounding tissue is captured by our learning-based corrections. Moreover, in this example we learn external corrections as a function of the size of the spherical collider, opening the possibility of using parametric shape models.

model. At this point, the limiting factor was the resolution of the mesh, not the size of the collider. We would need to increase the resolution of the mesh to ensure smooth contact as a smaller collider traverses the surface.

On this example, we also explored the ability to learn corrections as a function of other interaction parameters, such as the size of the collider. We generated training data with 4 different sphere radii, and at runtime we tested arbitrary in-between values. Even though the example explores a very limited shape parameterization, it opens up the possibility of training interactions with parametric and generative shape models [Lop*15; Wu*16].

4.5 Conclusions

We have presented an approach to design compact yet accurate subspace simulation models, by aggregating global linear handle-based subspace deformations and local nonlinear corrections. We have shown that local corrections can be effectively learned from deformation examples, through a separation into internal and external (contact-driven) corrections. The model enables fast simulations with a combination of interaction dynamics and deformation detail that is unprecedented to the best of our knowledge. Nevertheless, there are interesting avenues for future work, which could address current limitations and extend the applicability of the approach.

The method is heavy on preprocessing, as it requires extensive precomputation of high-resolution contact simulations in order to accurately learn contact-driven deformations. This could be alleviated through more sparse sampling of contact simulations, perhaps thanks to changes to the neural network architecture and/or optimization method to achieve better generalization under sparse data, through self-supervised training methods, or by designing more atomic correction strategies not at the whole object level as we do.

The model cannot handle arbitrary contact, and it is currently limited to rigid kinematic colliders. The formulation is general and it admits deformable colliders, by inputting their state to the external correction model. However, scalability is unclear, both in terms of training complexity and generalization ability. At a theoretical level, the formulation can also be extended to support simulated colliders, and in that case the elastic energy of the object under study would depend on the state of the collider through the corrections. At a practical level, this dependency could complicate runtime efficiency though, potentially introducing dense coupling in the Hessian of the full simulation.

Our implementation of the subspace model uses a frame-based approach for the linear basis, but the formulation admits more general linear subspace models, such as modal bases [PW89] or modal derivatives [BJ05]. The parameterization of nonlinear corrections and the data generation process exploit the handle-based basis, and would need to be reformulated for more general linear models. As described in Section 4.3.1, we learn external corrections as a function of relative transformations of the collider, $\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z}$. $\mathbf{A}(\mathbf{q})^{-1}$ works only for handle-based models, but for general linear models relative transformations could be encoded using more general feature vectors, such as pairwise distances between sets of points in the object and the collider. As described in Section 4.3.2, in a couple steps of the data generation process we must constrain the full-space deformation to the subspace. For general linear subspace models, this can be executed using the least-squares mapping $\mathbf{q} = (\mathbf{U}^T\,\mathbf{U})^{-1}\,\mathbf{U}^T\,\mathbf{x}$ from full space to subspace, and setting a constraint on the resulting subspace configuration.

Currently, the model admits only quasi-static corrections, but no dynamic corrections. Learning-based dynamic corrections could be approached in two ways, as done by other methods: by explicitly inputting previous states to the learning architecture [CO18; Hol*19], or by building a recurrent learning architecture [SOC19]. Friction is another source of trajectory-dependent deformations. Friction could be handled in a way similar to dynamics, e.g., by introducing previous states of the collider to the learning architecture, or also by modeling the friction state as an input explicitly [VCO20].

5 Learning Contact Produced Deformations

We propose a novel method to machine-learn highly detailed, nonlinear contact deformations for real-time dynamic simulation. We depart from previous deformation-learning strategies, and also from the approach we followed in Chapter 4, and we model contact deformations in a contact-centric manner. This strategy shows excellent generalization with respect to the object's configuration space, and it allows for simple and accurate learning. We complement the contact-centric learning strategy with two additional key ingredients: learning a continuous vector field of contact deformations, instead of a discrete approximation; and sparsifying the mapping between the contact configuration and contact deformations. These two ingredients further contribute to the accuracy, efficiency, and generalization of the method. We integrate our learning-based contact deformation model with subspace dynamics, showing real-time dynamic simulations with fine contact deformation detail. The contributions presented in this chapter have led to the following publication:

- Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy.
"Contact-Centric Deformation Learning".
ACM Transactions on Graphics (SIGGRAPH) (2022)

5.1 Introduction

The simulation of contact and deformations has drawn great interest in computer graphics, as it serves to bring to life computer-generated models of humans and their surrounding objects [Ter*87; McA*11; SGK18]. However, one of the remaining challenges in the field is to simulate high-resolution contact at interactive rates, e.g., for virtual reality applications.

In our work, we look at leveraging machine-learning methodologies to model contact-driven deformations, inspired by their success in modeling *self-driven deformations* [Pon*15; SOC19; SSR20], i.e., deformations that emerge as a function of the object's own motion.

These methods employ a subspace representation of the deformable object, and then learn rich nonlinear deformations as a function of the subspace state. Some works have already attempted to model contact deformations using machine-learning approaches, but they model only smooth global contact response [Hol*19]. The method presented in Chapter 4 allows detailed deformations to be learned, but requires extensive precomputations and shows very limited 3D interactions.

We hypothesize that there is a fundamental limitation in previous deformation learning strategies, including our method presented in Chapter 4. Deformations are modeled in an object-centric manner, which is an excellent choice for self-driven deformations, as they are smooth with respect to the object's subspace state, and then machine learning achieves good generalization even from sparse data. However, contact-driven deformations are not smooth with respect to the object's state; therefore, machine-learning these deformations would require dense sampling of the object's subspace state. This is hard, as the configuration space may be large and difficult to cover.

We depart from previous deformation-learning strategies, and propose a contact-centric strategy to learn contact-driven deformations. This is also the intuition behind sculpting brushes in digital sculpting [FCG99], and similar to the learning of skeletal deformations in local body-part coordinates [Den*20]. We demonstrate that our contact-centric approach shows excellent generalization with respect to the object's subspace state (down to just 8 training poses for a challenging 'duck' example with an 87-dimensional subspace state, or 1 pose for the 'hand' example in the teaser figure).

Our novel method, presented in Section 5.2, gathers three main components:

1. As outlined above, we model contact deformations in a contact-centric manner, i.e., on a local reference of the collider. We observe that contact deformations are smoother when modeled in a contact-centric manner, and this contributes to better generalization, and easier and more accurate learning.

2. We regard contact deformations as a continuous vector field. Instead of learning a discrete approximation, we learn the continuous field directly, inspired by recent work on implicit surface modeling [Xie*21]. Learning the contact deformation field generalizes continuity and differentiability to unseen configurations.

3. We sparsify the mapping between the contact configuration and the resulting contact deformations. In this way, we leverage the locality of contact deformations, and we learn them effectively from sparse data.

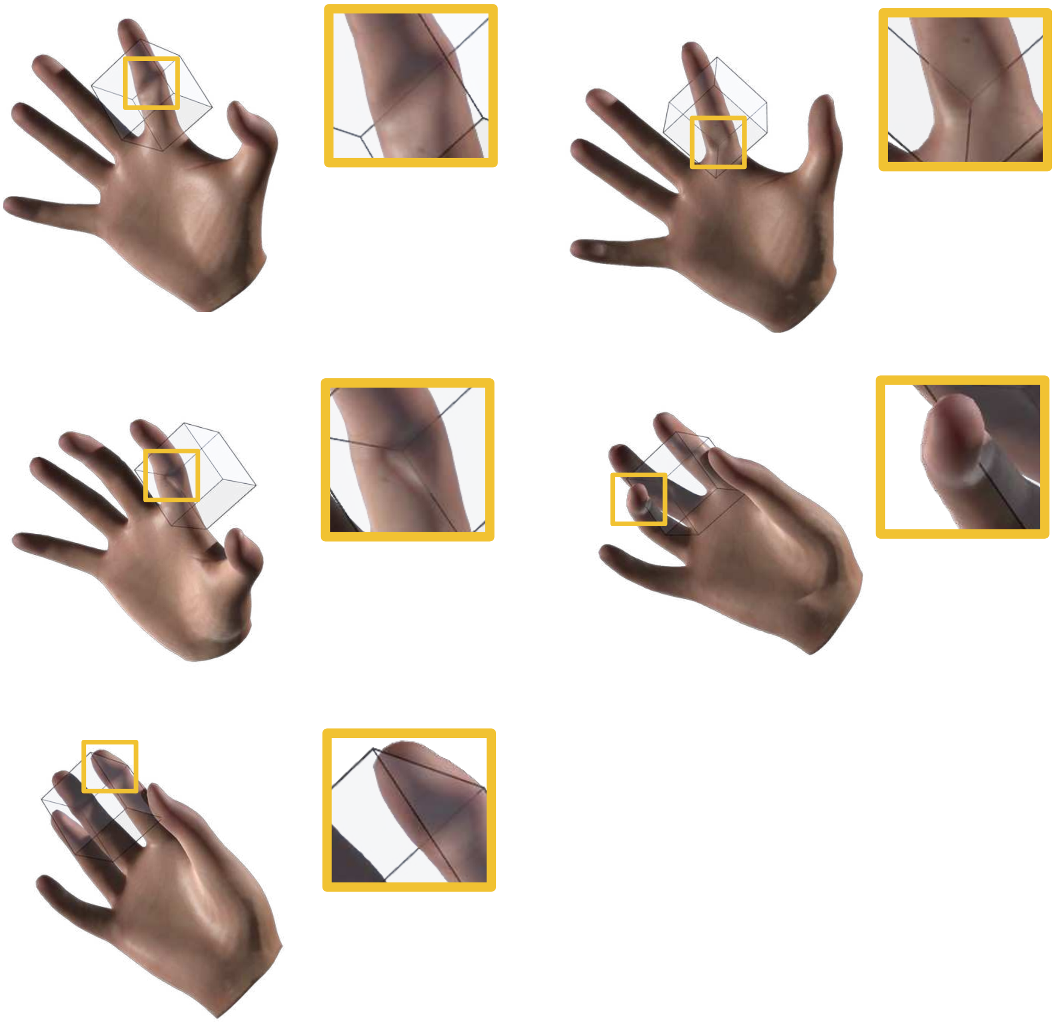


Figure 5.1: We present a learning-based method to augment a subspace deformable simulation with contact-driven deformation detail. We learn contact deformations in a contact-centric manner, which allows us to significantly reduce the sampling of configurations of the deformable object, and subsequently learn highly complex deformations. For this real-time simulation of the MANO model [RTB17] with dynamics, we used just one pose of the hand for training. Notice the accurate high-resolution deformations due to contact with a rigid object, highlighted in the zoom-ins.

In Section 5.3, we describe the neural-network approximation of our contact deformation model. We also discuss the efficient generation of training data. In Section 5.4, we discuss the simulation of dynamic deformations using our learning-based contact deformation model. We augment a dynamic subspace deformation with quasi-static contact-driven detail that is expressed in the same subspace, allowing simulations that are both fast and highly detailed.

We have applied our method to real-time dynamic simulations of different deformable objects. We show 2D and 3D subspace simulations generated with the bounded generalized biharmonic coordinates [Wan*15], and 3D simulations of the MANO hand model [RTB17]. We have augmented these dynamic subspace deformations with rich and highly-detailed contact deformations, all in real time.

5.2 Contact-Centric Deformations

In this section, we describe how we model contact-driven deformations. Our modeling approach, i.e., the selection of input and output representations of contact-driven deformations, is key for designing an effective learning-based approximation.

We start the section with a definition of the notation, as well as a description of our subspace deformable objects. Then, we define collider-space displacement fields, as a smooth representation of the deformation fields produced by contact. We continue with a discussion of continuous vs. discrete representations of the displacement field, and the impact on the design of a learning-based approximation. To conclude, we propose a sparse approximation of the displacement field to further improve the learning ability.

5.2.1 Definitions

We learn contact-centric deformations on a subspace deformable object $\mathcal{X}$. In the absence of contact, a point in (undeformed) object space $\bar{x} \in \mathbb{R}^3$ is mapped to a deformed position $\tilde{x}$ in world space through a subspace deformation. In our work, we show different subspace deformation models that combine a dynamic subspace deformation with quasi-static learning-based corrections (also parameterized in the same subspace). More specifically, one example we show is the use of combined point and frame handles, with a smooth deformation field defined by bounded generalized biharmonic coordinates (BGBC) [Wan*15], and further augmented with learning-based internal corrections as presented in Chapter 4.

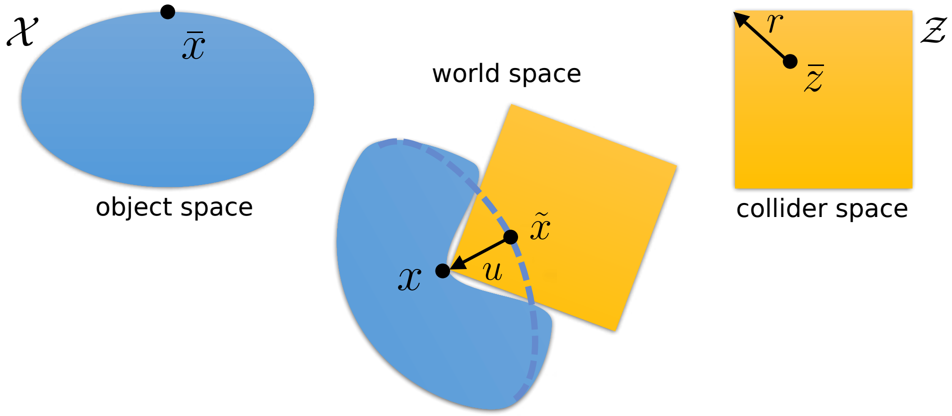


Figure 5.2: When a collider $\mathcal{Z}$ touches a deformable object $\mathcal{X}$, it produces a displacement field $u(\bar{x})$. We model the full deformation field $x(\bar{x})$ as the sum of a dynamic subspace deformation $\tilde{x}(\bar{x})$ and a learning-based approximation of the contact displacement field $u(\bar{x})$. A key insight of our method is to learn this field as a displacement $r(\bar{z})$ parameterized in collider space.

Another example that we show is the use of dynamic articulated skeletons, with linear blend skinning, and parametric pose-based corrections [RTB17]. We denote as $\mathbf{q}$ the subspace kinematic configuration of the deformable object $\mathcal{X}$.

Let us also consider a collider object $\mathcal{Z}$, and $\bar{z} \in \mathbb{R}^3$ a point in the collider space. In our work, we limit ourselves to rigid colliders. Then, we denote as $\mathbf{z}$ the rigid configuration of the collider $\mathcal{Z}$.

When the deformable object $\mathcal{X}$ touches a collider $\mathcal{Z}$, we augment the subspace deformation field $\tilde{x}(\bar{x})$ with a contact displacement field $u(\bar{x})$, which yields a total deformation field

$$x(\bar{x}) = \tilde{x}(\bar{x}) + u(\bar{x}), \tag{5.1}$$

as shown in Fig. 5.2. In our work, we model the subspace deformation $\tilde{x}(\bar{x})$ using dynamics, and the contact displacement $u(\bar{x})$ as a quasi-static deformation. In this way, deformable objects exhibit rich global dynamics combined with contact-driven detail.

5.2.2 Collider-Space Displacement

We wish to find a suitable parameterization of the contact displacement field $u(\bar{x})$ that allows efficient and accurate approximation with a learning-based architecture. In the limit case of a translation of a collider along a flat, infinite, homogeneous object, the displacement field induced by contact is constant when expressed in *collider space*. In more general cases, the collider may produce a global deformation on the deformable object, but far from the collider this deformation is well captured by the subspace deformation $\tilde{x}(\bar{x})$; it is close to the collider where the additional displacement $u(\bar{x})$ is relevant. We observe that, when the collider moves, this local contact-driven displacement varies more smoothly in collider space than in object space, as shown in Fig. 5.3.

Based on this intuition, we choose to parameterize the contact displacement in collider space, $r(\bar{z})$, as depicted in Fig. 5.2. Then, to evaluate the world-space displacement, we first transform the subspace deformation $\tilde{x}(\bar{x})$ to collider space, and then transform the collider-space displacement again to world space. With $T(\mathbf{z})$ a rigid transformation based on the collider's configuration, the displacement is formally obtained as

$$u(\bar{x}) = T(\mathbf{z}) \cdot r(\bar{z}), \quad \text{with } \bar{z} = T(\mathbf{z})^{-1} \cdot \tilde{x}(\bar{x}). \tag{5.2}$$

The contact displacement field depends on the relative configuration between the deformable object and the collider, which we express as $T(\mathbf{z})^{-1} \cdot \mathbf{q}$. In practice, we implement this by transforming all point and frame handles of the deformable object to the local reference frame of the collider. Based on this relative configuration, the contact displacement field can be defined by the following function:

$$r(\bar{z}) \equiv f \left(\ \bar{z}, \quad T(\mathbf{z})^{-1} \cdot \mathbf{q} \ \right). \tag{5.3}$$

We approximate the function f using machine learning. As discussed above, f is in practice a smooth function of the relative configuration between the collider and the deformable object. Typical data-driven deformation methods [Pon*15; SOC19; SSR20] learn instead object-space deformations $T(\mathbf{q})^{-1}\, u(\bar{x})$. However, we have found that the contact displacement parameterized in object space is far less smooth. As a direct consequence, the collider-space displacement function f can be learned using far fewer training data and with a smaller network than an object-space displacement function.

5.2.3 Learning of a Contact Displacement Field

For dynamic simulation of contact mechanics, the deformation field $x(\bar{x})$, and therefore the contact displacement $u(\bar{x})$, must be evaluated at two types of points. One type is points on the surface of the deformable object $\mathcal{X}$, for the computation of contact potentials or contact constraints. The other type is points within $\mathcal{X}$, for the computation of the internal energy

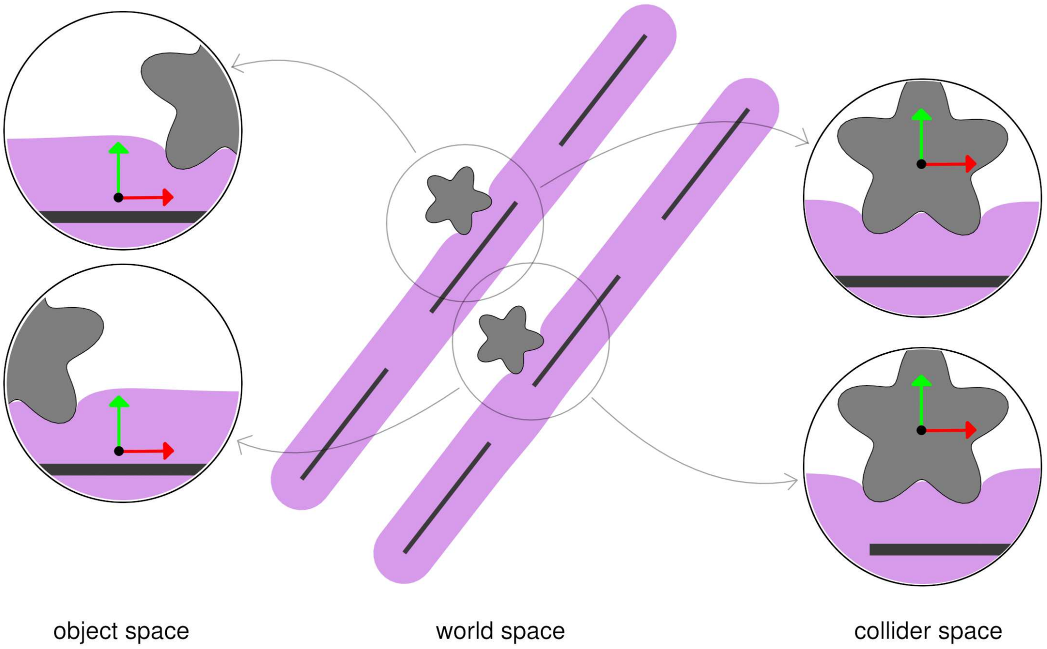


Figure 5.3: The close-ups compare the representation of contact displacements in object space $\bar{x}$ (left) vs. collider-space $\bar{z}$ (right) for these two examples. As the collider sweeps through the surface of the deformable object, collider-space contact displacements are notably smoother, and this drastically impacts the learning ability of our method.

(and its derivatives). Due to the subspace deformation, in practice we use cubature points in the second case [AKJ08].

Both types of evaluation points are fixed in object space $\bar{x}$. Therefore, under an object-space parameterization of contact displacements, it turns out convenient to learn directly the discrete representation of this function. Furthermore, a common approach in machine learning is to project such high-dimensional representations to a compact linear subspace using PCA, and learn only a small number of PCA coefficients.

However, the evaluation points are not fixed in collider space. While it might be possible to sample the collider space, and apply PCA-based learning, the resulting collider-space displacements should be interpolated to the evaluation points. Instead, motivated by recent methods that learn continuous fields [Xie*21], we opt to learn the contact displacement function f directly as a vector field.

Furthermore, by learning the field f using a multilayer perceptron (MLP) network, the result is memory-efficient, continuous, and fully differentiable, which are key properties for successful dynamic simulation. The displacement $u(\bar{x})$ is defined only inside the deformable object $\mathcal{X}$, and this leads to a discontinuity in the sampling of the collider space

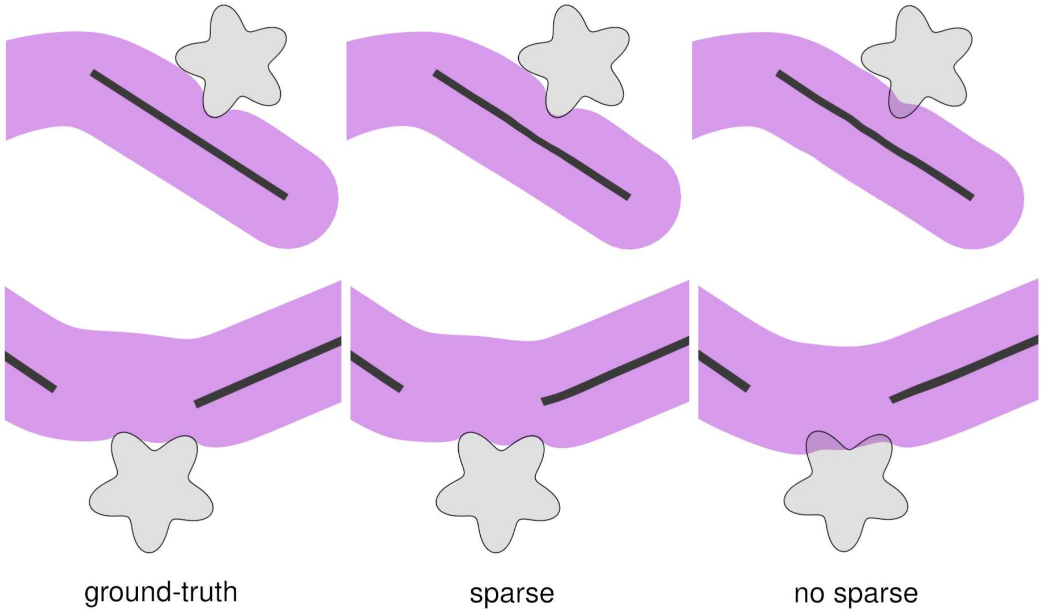


Figure 5.4: Two examples (top, bottom) to depict that contact displacements are dominated by the configuration of nearby handles/bones of the deformable object. We leverage this observation designing a sparse approximation of the contact displacement function. Here, we compare ground-truth displacements (left), learned displacements with sparsifying weights, i.e., Eq. (5.4) (middle), and without sparsifying weights, i.e., Eq. (5.3) (right). With the same training data, the sparse function achieves superior results, as it succeeds to disambiguate the subspace state that contributes to the contact displacements.

$\bar{z}$ when learning the function f. However, the inductive bias of the MLP network smoothly generalizes to unseen points in the collider space $\bar{z}$, which may be queried at runtime. The computation of forces and their derivatives requires the evaluation of gradients with respect to the collider space. However, the differentiability of the network provides gradient evaluation by construction.

5.2.4 Sparsification of the Learning Function

The major challenge in learning the contact displacement function f is the dimensionality of the configuration $\mathbf{q}$ of the deformable object. At first sight, applying our contact-centric deformations to objects with a rich underlying subspace $\mathbf{q}$ (i.e., with many subspace degrees of freedom) requires a combinatorial explosion of the deformed configurations that must be fed as training data to learn f, and a function that is more complex and more challenging to learn.

However, as discussed earlier, we can safely assume that contact displacements have local support, as deformations far from the collider are coarser and well represented by the underlying subspace deformation. Then, the contact displacement at an object-space location $\bar{x}$ is only influenced by the configuration $\mathbf{q}$ of nearby handles of the subspace deformation. Note that, even though the contact displacement function f is parameterized in collider space $\bar{z}$, it implicitly depends on the object space $\bar{x}$ through $\bar{z} = T(\mathbf{z})^{-1} \cdot \tilde{x}(\bar{x})$.

Based on the observations above, we approximate the contact displacement field (5.3) through a sparse function:

$$r(\bar{z}) \approx f\left(\; \bar{z}, \quad \mathbf{W}(\bar{x}) \cdot T(\mathbf{z})^{-1} \cdot \mathbf{q} \; \right). \tag{5.4}$$

where $\mathbf{W}(\bar{x})$ is a matrix of spatially varying sparsifying weights, i.e., many of its rows are zero.

We leverage the sparsity of our subspace deformation models to define the sparsifying weights. Specifically, with $\mathbf{U}(\bar{x})$ the subspace basis (e.g., BGBC basis or skinning weights) at a material point $\bar{x}$, we build the weights as $\mathbf{W}(\bar{x}) = \mathrm{diag}(\mathbf{U}(\bar{x}))$. Similar ideas of spatially varying sparse weights have been used in other contexts to obtain local pose definitions, e.g., weighted pose-space deformation [KM04] or pose attention weights [Sai*21]. Fig. 5.4 shows an example comparing the learning accuracy with a sparse vs. a dense learning function, and we provide a quantitative analysis in Section 5.5.

5.3 Data and Learning

We pose the problem of designing a learning-based approximation of the contact displacement function f in (5.4). Solving this problem requires addressing several tasks, which define the structure of this section.

First, we address the design and training of a neural network architecture $\mathbb{N}$ to compute the contact displacement function f. Second, we describe our strategy for sampling the arguments of f, i.e., the collider space, $\bar{z}$, and the relative configuration between object and collider, $T(\mathbf{z})^{-1} \cdot \mathbf{q}$. To conclude, we describe the generation of ground-truth contact displacements $r(\bar{z})$, which requires solving contact configurations with and without contact displacements.

5.3.1 Neural Network Architecture

We use a fully-connected, 2-layer MLP to model the function f (5.4), with `tanh` as activation function. This results in our neural model of contact deformation $\mathbb{N}$. The actual size of each layer depends on the specific example; see Section 5.5 for details. Our research focus was on the design of the function to be learned, not on the learning architecture. However, similar to other works that learn fields, we investigated the use of Fourier features [Sit*20; Ben*22] to improve the learning ability of the neural network. Nevertheless, our initial attempts were not successful, as the generalization outside the sampled region became worse. We leave optimizations of the neural network architecture as future work.

To learn the parameters of the network, we define a loss function that combines two terms. One is the L2 error of estimated contact displacements vs. ground-truth training displacements, summed over points in the volume of the deformable object $\mathcal{X}$. The other term is the L2 error of differences of estimated contact displacements vs. differences of ground-truth training displacements, summer over surface edges of $\mathcal{X}$. We observed best preservation of contact detail when combining both loss terms.

5.3.2 Sampling the Contact Displacement Function

The generation of training data requires sampling the arguments of f, i.e., the collider space, $\bar{z}$, and the relative configuration between object and collider, $T(\mathbf{z})^{-1} \cdot \mathbf{q}$. Sampling the collider space is easy. Motivated by the loss function defined above, we use the nodes $\{\bar{x}_i\}$ of the volumetric mesh of the deformable object, and we transform them to the collider space through $T(\mathbf{z})^{-1} \cdot \tilde{x}(\bar{x}_i)$. Note that the collider-space samples vary depending on the relative configuration of the deformable object and the collider.

Sampling the relative configuration between the object and the collider is more challenging. The dimensionality of this space can be seen as the number of degrees of freedom of the deformable object, leaving the collider fixed. However, under this view, the space would be very difficult to sample, as we care only about colliding configurations. Alternatively, we look at the space as the Cartesian product of four subspaces: the configuration $\mathcal{Q}$ of the deformable object after removing rigid transformations, the surface of the deformable object $\partial\mathcal{X}$ which defines contact locations, the rotation SO(3) of the collider, and the penetration depth $\mathcal{D} \subset \mathbb{R}$ between collider and deformable object. The full space can be represented as $\mathcal{Q} \times \partial\mathcal{X} \times \text{SO(3)} \times \mathcal{D}$. We sample each of these four subspaces independently. For the

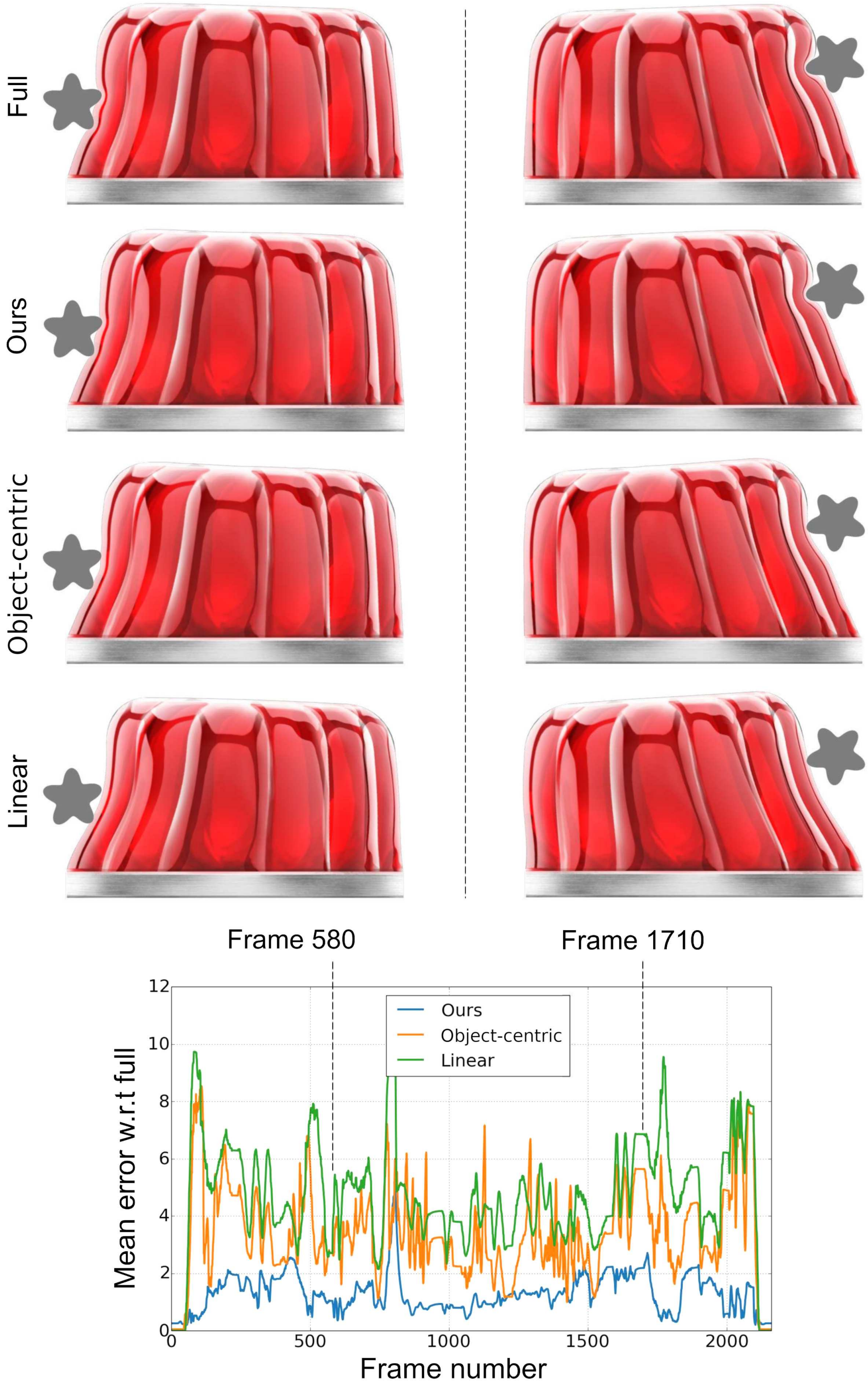


Figure 5.5: Our approach significantly improves the generalization capabilities of the object-centric method presented in Chapter 4, and closely matches the realism of full simulation. Our method is able to learn the complex interaction between the star-shape collider and the deformable jelly using one order of magnitude less neurons and training data than the original settings in Chapter 4. In contrast, when trained with such reduced dataset, the object-centric approach from Chapter 4 is unable to learn deformations due to contact.

penetration depth, we simply use evenly distributed samples up to a maximum depth, with a bit of random noise. For the other three subspaces, we generate a large potential set of samples and we pick a representative subset using a greedy furthest-point strategy (based on surface geodesic distance for $\partial\mathcal{X}$ and norm of axis angle for $SO(3)$).

We pay special attention to sampling the rigid-free configuration $\mathcal{Q}$ of the deformable object. We start by executing interactive contact simulations between the deformable object and the collider, leveraging the speed of the subspace simulation model of the deformable object. To represent the rigid-free configuration space $\mathcal{Q}$, we build a graph of handle connectivity of the subspace model, and for each state $\mathbf{q}$ of the deformable object we compute relative handle transformations for all edges in the graph. Given a data set of relative transformations, we normalize separately the entries corresponding to each edge. Based on this definition of rigid-free configurations, for furthest-point selection we use the Euclidean distance between normalized edge transformations.

Thanks to the smoothness of collider-space contact displacements, together with our decomposition of the relative configuration between the object and the collider, and the furthest-point sample selection discussed above, we manage to drastically reduce the number of samples needed in $\mathcal{Q}$, the configuration of the deformable object. This is arguably the hardest subspace to sample, and a naïve learning strategy would require exhaustive exploration of the configuration space. Instead, as shown in our results in Section 5.5, we sample complex high-dimensional configuration spaces (29 point handles in the ‘duck’) with fewer than 10 configuration samples, yet the learned model generalizes well to unseen states.

5.3.3 Ground-Truth Contact Displacements

Using the procedure described above, we can sample representative contact configurations of the collider and the deformable object in an efficient manner. Next, for each of these configurations we must compute ground-truth contact displacements $r(\bar{z})$, as the difference between full-space deformations $x(\bar{x})$ and subspace deformations $\tilde{x}(\bar{x})$. However, it is important that the subspace states $\mathbf{q}$ of these deformation fields match.

The subspace deformation is directly given by the interactive generation of contact configurations. Therefore, we are left to compute a full-space deformation constrained to the same subspace state. Even though this task is part of preprocessing, a typical constrained dynamics solve based on Lagrange multipliers could be very time-consuming, due to the large number of simulations and the size of the full-space representation. Instead, we restrict

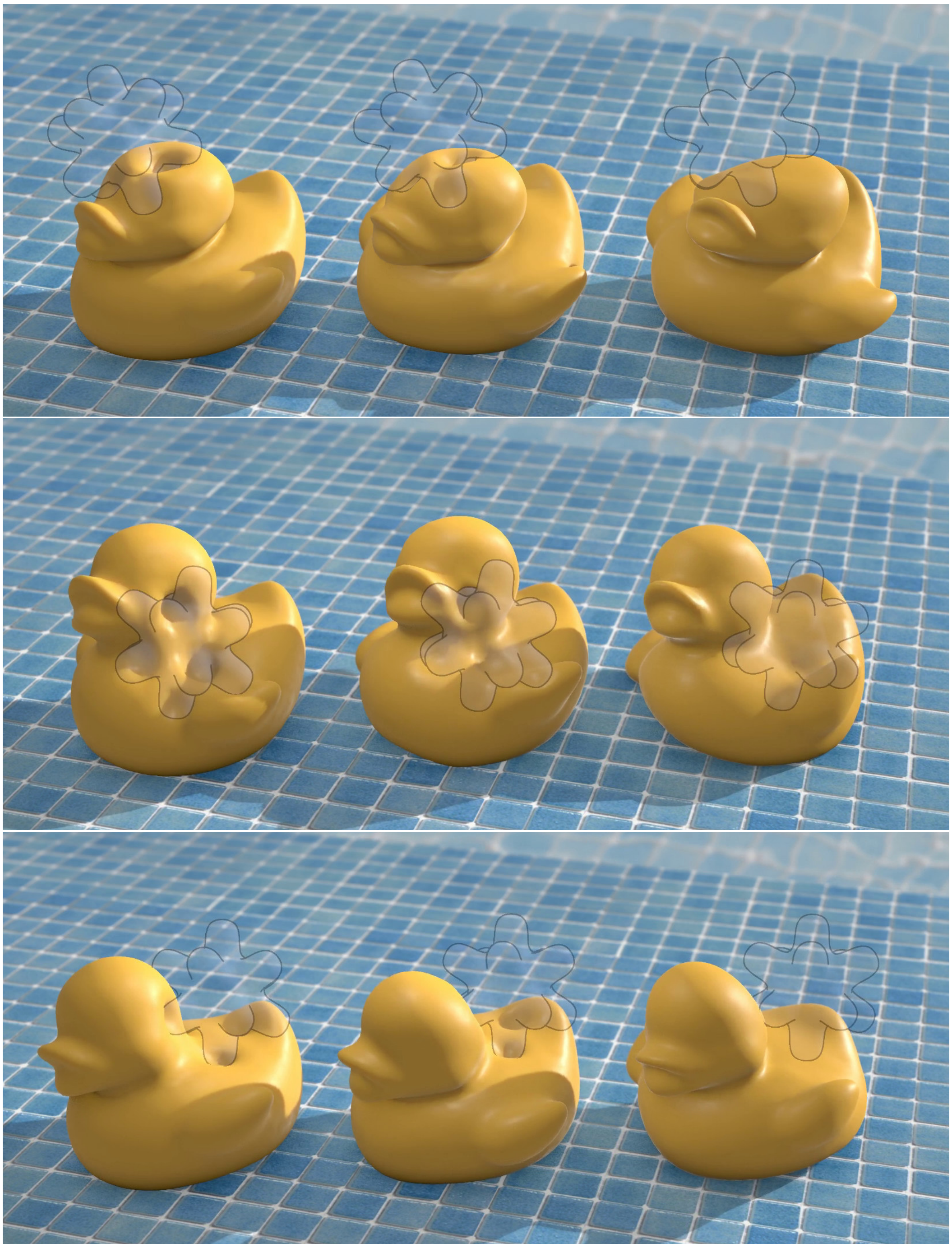


Figure 5.6: Qualitative evaluation. We show 3 frames of a sequence where a collider (semitransparent, for better visualization) interacts with a rubber duck. Our method (center), closely matches the natural deformations due to contact that emerge using a full simulation model (left). In contrast, a linear model [Wan*15] (right) is unable to deform correctly.

the full-space simulation to the null-space of the subspace using a projection method. Given the basis $\mathbf{U}$ of the subspace, the matrix $\mathbf{P} = \mathbf{I} - \mathbf{U}\left(\mathbf{U}^T\,\mathbf{U}\right)^{-1}\mathbf{U}^T$ represents a projection to the null-space of the subspace. To run the full-space simulation, we use the modified conjugate gradient method [AB03], with $\mathbf{P}$ as projection matrix. Note that we do not explicitly compute $\mathbf{P}$, we only compute the Cholesky factorization of $\mathbf{U}^T\,\mathbf{U}$, which is small and fast, and we apply the various matrix multiplications on each iteration of conjugate gradient.

5.4 Simulation of Dynamic Deformations

Our novel contact-centric learned deformations can be added to a dynamic subspace simulation model, retaining the fast subspace formulation in the combined simulation. In this section, we formulate the full dynamics problem, paying special attention to the inclusion of the learning-based contact displacement function $\mathbb{N}$ described in 5.3.1.

We find it convenient to formulate dynamics as an optimization problem, using the optimization-formulation of backward Euler [Mar*11; Gas*15]. In this way, we can seamlessly use our deformation field definitions (Section 5.2.1), integrate quantities on the full-space, and optimize only subspace degrees of freedom. Given a collider configuration $\mathbf{z}$, and an explicit-Euler update of full-space positions $x^*(\bar{x})$, the subspace configuration of the deformable object is computed as:

$$\begin{aligned}
\mathbf{q} &= \arg\min W_{\text{inertial}} + W_{\text{elastic}} + W_{\text{contact}} \\
W_{\text{inertial}} &= \int_X \frac{\rho}{2\,h^2}\,\|x(\bar{x},\mathbf{q},\mathbf{z}) - x^*(\bar{x})\|^2\,d\bar{x}, \\
W_{\text{elastic}} &= \int_X \Psi(x(\bar{x},\mathbf{q},\mathbf{z}))\,d\bar{x}, \\
W_{\text{contact}} &= \int_{\partial X} \Phi(T(\mathbf{z})^{-1}\cdot x(\bar{x},\mathbf{q},\mathbf{z}))\,d\bar{x}.
\end{aligned} \tag{5.5}$$

Here, ρ is the mass density of the object, h is the time step, and the full-space deformation field $x(\bar{x})$ is defined by combining (5.1), (5.2) and (5.4):

$$\begin{aligned}
x(\bar{x},\mathbf{q},\mathbf{z}) = {} & \tilde{x}(\bar{x},\mathbf{q}) \\
& + T(\mathbf{z})\cdot\mathbb{N}\Big(\ \bar{z} = T(\mathbf{z})^{-1}\cdot\tilde{x}(\bar{x},\mathbf{q}),\quad \mathbf{W}(\bar{x})\cdot T(\mathbf{z})^{-1}\cdot\mathbf{q}\ \Big).
\end{aligned} \tag{5.6}$$

In this expression, we show the explicit dependencies of the deformable object configuration $\mathbf{q}$, as these are important for the evaluation of gradients.

Table 5.1: Details about dataset size and runtime performance for the different objects used to showcase our method. For descriptions about sample types (e.g., $\mathcal{Q}$, $\partial\mathcal{X}$, SO(3), $\mathcal{D}$), see Section 5.3.2.

Example	handles	mesh size	# samples				neurons	fps		
	point/bone	tris/tets	$\mathcal{Q}$	$\partial\mathcal{X}$	SO(3)	$\mathcal{D}$		linear	full	ours
Duck	29/0	37,049	8	64	30	5	500	277	2	25
Floater	24/0	39,450	8	64	30	5	500	211	1	23
Hand	0/16	82,395	1	300	48	5	500	81	1	36
Jelly 2D	8/1	13,720	5	21	32	5	300	548	5	53
Jelly 3D	18/1	60,830	5	64	30	5	500	191	1	19
Worm	0/3	20,213	7	64	32	5	300	428	3	50

In (5.5) above, Ψ is an elastic energy model. In our case, we have used the stable Neo-Hookean formulation. Φ is a contact potential based on a signed distance field precomputed for the collider. It is zero for negative distances and cubic for positive distances. We integrate the inertial and elastic terms using cubature [AKJ08], with cubature points and weights estimated using the data-oblivious approach presented in Section 3.7.2. We integrate the contact term using all surface mesh points instead.

To solve the optimization (5.5), we use a Newton-CG solver. To evaluate gradients of the learning function $\frac{\partial r}{\partial \mathbf{q}}$, we perform back propagation on the neural network. We do not store the Hessian explicitly, but instead execute Hessian-vector products. In this regard, we ignore the Hessian of the learning function, and we implement gradient-vector products through an auxiliary back propagation step as explained in Section 4.3.3. Despite the small size of the subspace Hessian, we found more convenient to use Newton-CG than a direct solver, as each Newton step required very few CG iterations in practice, and hence minimized the number of network evaluations.

5.5 Experiments

In this section we quantitatively and qualitatively evaluate our method in a variety of objects, scenarios, and interactions. Additionally, we compare it to the method presented in Chapter 4, where we also model contact deformations using a data-driven approach. As baseline, we also show results using a linear subspace model based on Wang et al. [Wan*15]. All the examples were executed on an Intel Core i7-7700K 4-core 4.20 GHz PC with 32 GB of RAM.

In Table 5.1 we provide details of the objects and datasets used to generate our results, including the mesh discretization of the full-space simulations.

Jelly 2D. Figure 5.5 shows frames of a star-shape collider interacting with a 2D jelly, using 4 different methods: full simulation, our contact-centric approach, object-centric method from Chapter 4, and the linear method of Wang et al. [Wan*15]. To stress the generalization capabilities of our approach, we used a reduced version of the dataset from Chapter 4, consisting of 16,800 ground truth samples ($\approx 15\times$ less than the original dataset). To quantitatively evaluate our results, we also plot the mean per-vertex error of each method through a test sequence of more than 2,000 frames. Notice, moreover, that our model requires only 300 neurons while the object-centric method in Chapter 4 requires 3,000 neurons.

Jelly 3D. Figure 5.7 shows a similar comparison in 3D, with a pointy collider interacting with a 3D jelly. We demonstrate that our collider-centric learning approach is accurate when trained with just 5 configurations of the jelly. The training cost is dominated by the simulation of all training samples ($48,000$ in total, accounting for all sampled configurations of the collider), which took 37 hours. With the same training data, object-centric learning (Chapter 4) fails to produce accurate results. Object-centric learning suffers the curse of dimensionality, and multiplying the training data to 25 configurations of the jelly (184 hours) barely improved the results. The errors for all configurations are also compared numerically in Table 5.2. The test examples have been generated by projecting static full-space deformations to the handle-based subspace, and then adding learning-based contact deformations. Errors are normalized with respect to the difference between full-space and linear subspace deformation.

These results show that, when using the same amount of training data, our method generalizes much better than the object-centric method presented in Chapter 4, and it closely reproduces the realism of a full simulation. Importantly, our method is not only able to train with less data, but also to model more complex and highly deformable contact interactions at real-time framerates.

Table 5.2: Relative error for the different methods and training settings of the 3D Jelly example shown in Fig. 5.7.

# samples Q	5	25
Ours	57%	56%
Object-centric (Chapter 4)	99%	86%

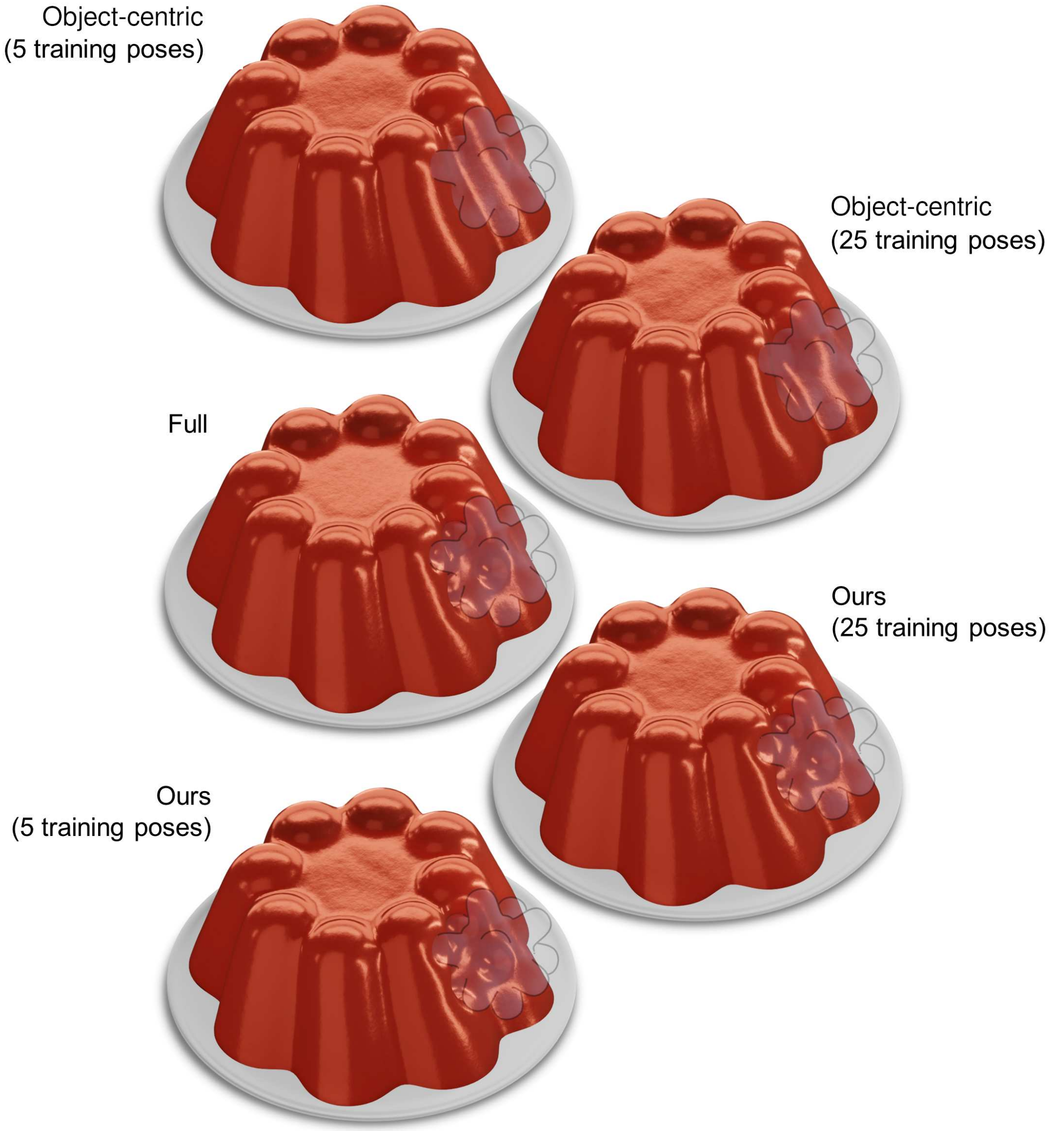


Figure 5.7: The generalization capabilities of our collider-centric method are also evident in this 3D jelly example. Our method is accurate when trained with just 5 poses of the jelly, and increasing the number of poses to 25 provides little gain. In contrast, object-centric learning as done in Chapter 4 fails to learn contact deformations with 5 poses, and only slightly improves with 25 poses. In Table 5.2 we provide numerical comparisons. Object-centric learning suffers the curse of dimensionality, and would require an intractable number of training poses.

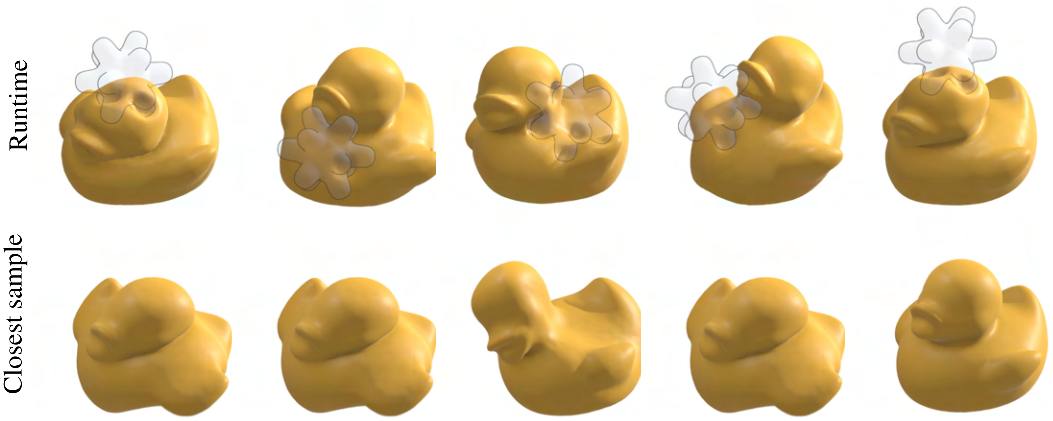


Figure 5.8: To depict the good generalization capabilities of our method, here we visualize the *closest* training sample (bottom) to a wide range of different states of the deformable duck (top). Frames were randomly picked from a sequence where the collider interacts in real time with the duck. For this particular demo, we use only 8 samples of the duck state $\mathcal{Q}$ to train. Since our approach is collider-centric, it generalizes well to unseen states of the deformable duck.

Worm 2D. We have used the worm in Figures 5.3 and 5.4 to evaluate the qualitative and quantitative effect of the sparsification of the learning function. We have compared the error with respect to a full-space simulation, with and without sparsification, for different amounts of training data (varying the samples of the worm's configuration space). The errors reported in Table 5.3 confirm that sparsification allows a drastic reduction in the amount of training data required.

Hand. Figure 5.1 showcase an interactive sequence where a 3D hand manipulates a rigid cube. The subspace deformation is built with the MANO model [RTB17], and the skeleton is dynamically simulated. The runtime interaction was produced interactively with a LeapMotion device for hand tracking, and commanding the hand's skeleton through spring forces. Notice how the skin surface of the hand naturally deforms when it touches the cube, even on sharp edges and corners, all in real time. This example was trained on a single flat pose of the hand. Despite such extremely simple sampling of the configuration space of the hand, our contact-centric formulation, together with the sparsifying weights, achieve excellent generalization to unseen hand poses. We demonstrate that the accuracy of the

Table 5.3: Relative error in the Worm example in Figures 5.3 and 5.4, with and without sparsification, for different amounts of training data.

# samples $\mathcal{Q}$	1	3	7
sparse	44%	43%	39%
no sparse	96%	76%	47%

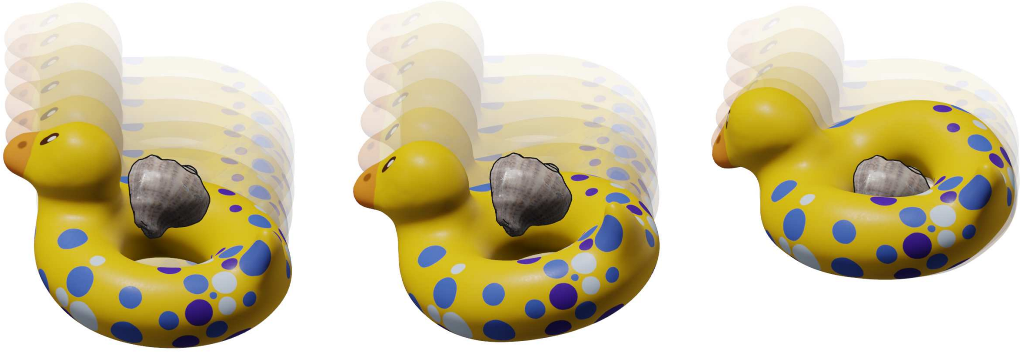


Figure 5.9: A full-space floater (left) falls on top of a rigid shell and it deforms, to let the shell pass through. This motion is well represented with our method (middle), while a linear subspace (right) fails to represent the necessary deformations, and the shell gets stuck.

contact deformation is well-kept across all hand regions, including palm and fingertips, and for any hand pose.

Ducks. Figure 5.6 shows frames of a sequence where a collider closely interacts with a rubber duck, and qualitatively compares our results to the deformations obtained with a linear subspace model [Wan*15] and a full-space simulation. The linear model, bounded by the limited expressivity of the subspace, is unable to reproduce the deformations due to contact, producing an unnatural behavior. In contrast, our approach is capable of accurately modeling deformations due to contact, closely matching the realism of the full-space simulation, even in situations with strong interactions and heavily deformed states.

In Figure 5.8, to qualitatively evaluate the generalization capabilities of our approach, we show frames of a sequence where we interactively manipulate the duck with a collider. For each frame, we show the closest duck deformation in the training set. Notice that, even when the closest sample is far from the current state of the duck, our method is able to accurately reproduce the deformations due to contact. In the particular case of the Duck scenario, our contact-centric representation is able to learn accurate contact with a deformable object using as few as 8 deformed examples. Importantly, even if we learn deformations without dynamics or friction, at runtime our method generalizes well to those settings.

Floater. Our method does not explicitly rely on geometric features of the deformable object or the collider, such as genus or symmetries. Figure 5.9 shows a scene with a deformable floater of genus 1 and a shell collider. The floater falls on top of the shell and it deforms, to let the shell pass through. With a linear subspace method, the shell fails to pass through the floater, as the necessary deformations are not well represented. Our method, on the other hand, represents them correctly.

5.6 Conclusions

In this work, we have presented a contact-centric method to learn contact-driven deformations. These deformations are added to a subspace dynamic simulation model, to produce real-time dynamic simulations of deformable objects with rich contact detail. We have demonstrated that contact-centric parameterization of the learning function drastically simplifies its complexity: the space of configurations can be sparsely sampled, and the resulting learning models are smaller, more efficient, and easier to learn. We further complement contact-centric modeling with a continuous field representation and sparsification of the learning function, which contribute to excellent generalization capabilities.

Our work is not free of limitations, and some of these suggest directions for non-trivial future work. We learn a deformation model per collider object. This could find applicability in interactive applications where colliders are known in advance, but it fails to address applications where multiple colliders interact in complex ways or where colliders are defined dynamically. We also assume that the collider is rigid, which again covers a large set of use cases, but it does not account for deformable-deformable contact.

We have introduced our contact-centric modeling approach in the context of deformable object simulation. However, this approach might find applicability in other problems of interaction with objects, such as joint tracking of hands and objects, or grasp synthesis.

6 Learning Generalized Contact Deformations

This chapter presents a learning-based method for the simulation of rich contact deformations on reduced deformation models. Previous works, including our models in Chapters 4 and 5, learn deformation models for specific pairs of objects; we lift this limitation by designing a neural model that supports general rigid collider shapes. We do this by formulating a novel collider descriptor that characterizes local geometry in a region of interest. The chapter shows that the learning-based deformation model can be trained on a library of colliders, but it accurately supports unseen collider shapes at runtime. We showcase our method on interactive dynamic simulations with animation of rich deformation detail, manipulation and exploration of untrained objects, and augmentation of contact information suitable for high-fidelity haptics. The method presented in this chapter has led to the following publication:

- Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy.
“Learning Contact Deformations with General Collider Descriptors”.
SIGGRAPH ASIA Conference Proceedings (2023)

6.1 Introduction

The simulation of contact deformations remains an open challenge in computer graphics and beyond. Despite the continuous development of algorithms for faster, more accurate, and/or more robust contact deformations [BEH18; HZ13; SGK18; Li*20], the targets of resolution and performance grow unstopably, and call for even faster and more accurate methods. This is particularly relevant as we envision real-time applications where we interact with objects that appear and feel real.

In recent years, machine learning has been explored as an approach to accelerate the runtime computation of rich deformations [Pfa*21; Hol*19; Ful*19], via exhaustive precomputation of the parameters of neural models. Few of these learning-based methods address the computation of deformations induced by contact, and those that do make many limiting assumptions about the nature of the colliding objects, as we do in Chapters 4 and 5. For instance, to the best of our knowledge, learning-based models for rich contact deformation are trained on specific pairs of objects, and therefore do not scale to the combinatorial complexity of object-object interactions.

In this work, we lift a crucial limitation of learning-based contact deformation models, and we present the first model that generalizes the collider geometry, i.e., it is not trained on a specific collider. Our learning-based deformation model augments a reduced deformable model (which is fast but lacks detail) with rich contact deformations induced by any rigid collider shape. Our key observation is that contact deformations not captured by the reduced model are local, hence we train the learning-based model on local shape descriptors of colliders, not on the full collider shape. In Section 6.2 we describe the *collider descriptor* in detail, as well as the formulation and training of the neural contact deformation model. Our learning-based deformation model is trained on a library of colliders, but supports general colliders at runtime. In Section 6.3 we discuss efficient runtime evaluation of the collider descriptor.

Our proposed collider descriptor can be connected to shape descriptors. However, while shape descriptors are typically used for labeling or classification problems [Qi*17a; Wan*19], we use our collider descriptor as a conditional signal for a neural deformation field. We find that, for our application of contact deformation, it is convenient to design a descriptor that encodes together local geometry and the relative configuration with respect to this local geometry. Moreover, in the design of the collider descriptor, we have addressed challenges concerning spatial smoothness and rotational invariance.

In our examples, we have integrated the learning-based contact deformation model in interactive dynamic simulations. We show that our model can reach the accuracy of previous works trained on specific colliders, while our model supports general collider shapes and therefore infinitely more diverse runtime contact scenarios, as shown in Fig. 6.1. Moreover, we showcase interesting applications beyond visual animation of deformation detail, such as exploration and manipulation of products for e.g. online retail, or augmentation of contact information suitable for high-fidelity haptics.

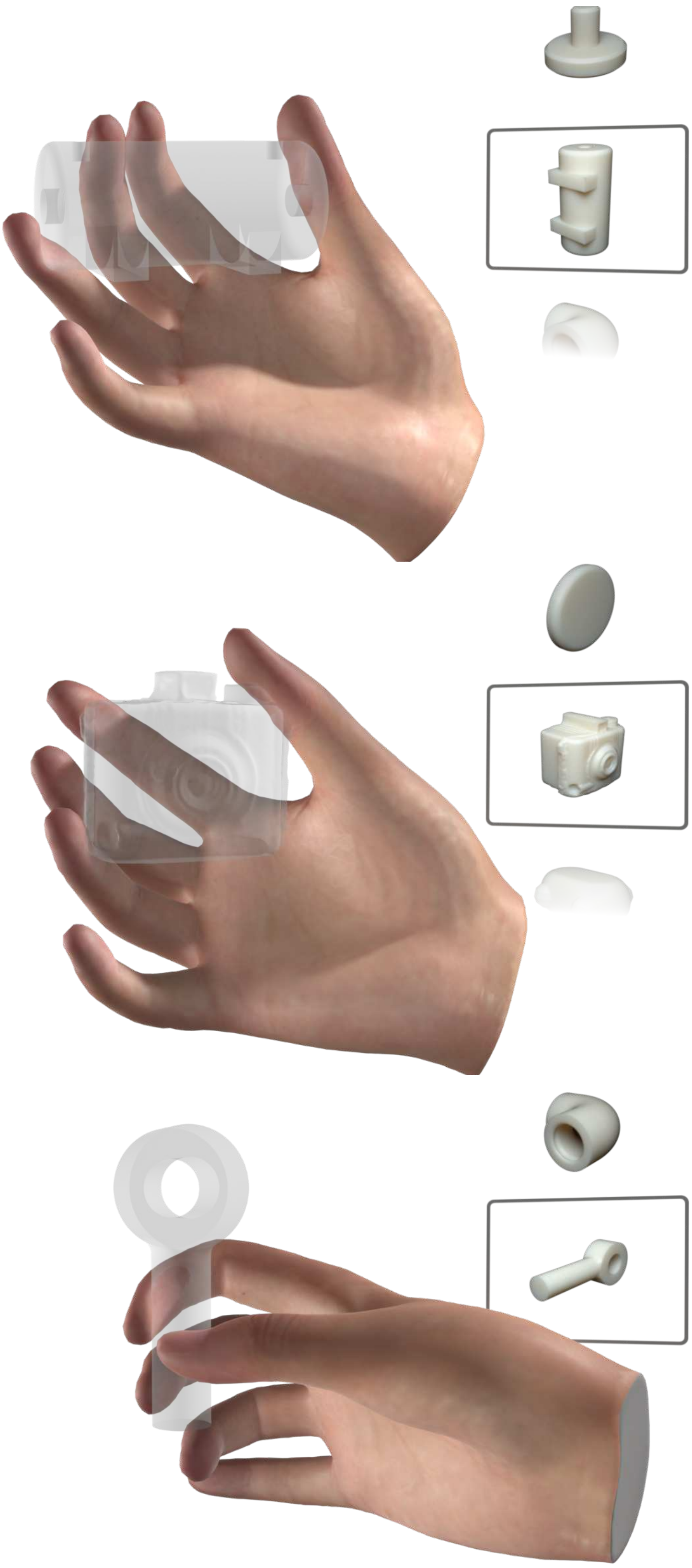


Figure 6.1: Interactive tactile exploration of diverse objects. We compute detailed contact deformations in real time using a learning-based model that generalizes to collider shapes. None of the colliders shown in the images were used for training.

6.2 Deformation Based on Collider Descriptors

Given an arbitrary rigid collider and a certain reduced deformable object, we seek to design a neural deformation model that augments the reduced model with rich and detailed deformations resulting from the interaction with the collider. We start this section by formalizing the definition of the contact deformation model and motivating the collider descriptor that serves as input to the model. We continue with a detailed discussion of the components of the collider descriptor. To conclude, we formalize the neural model and we discuss how it is trained on a library of rigid colliders. Fig. 6.2 outlines the computational pipeline of our contact deformation model.

6.2.1 Motivation of the Collider Descriptor

Let us formally define the kinematics and shape of both the rigid collider and the deformable object, and then the main components of our proposed model. We characterize the rigid collider by its rest-shape $\mathcal{Z} \subset \mathbb{R}^3$ and a rigid transformation $T \in SE(3)$. Conversely, we characterize the deformable object by its reduced degrees of freedom (DoFs) $\mathbf{q} \in \mathbb{R}^{n_r}$. Given a rest-shape parameterization $\bar{x} \in \mathbb{R}^3$, the baseline reduced deformed state is $\tilde{x}(\bar{x}, \mathbf{q})$. We have demonstrated our neural deformation model on handle-based reduced deformations [Wan*15], but our formulation is agnostic of the reduced model for the most part. We only leverage the handle-based representation to parameterize several heuristics.

We generally define a contact deformation field $u(\bar{x}) \in \mathbb{R}^3$ as a function of the point of interest in space $\tilde{x}$, the reduced deformation state $\mathbf{q}$, the shape of the collider $\mathcal{Z}$, and the transformation of the collider T. Formally, $u(\bar{x}) = f(\tilde{x}(\bar{x}, \mathbf{q}), \mathbf{q}, \mathcal{Z}, T)$. Adding this contact deformation field to the baseline reduced deformation, we obtain the full deformed state $\tilde{x}(\bar{x}) + u(\bar{x})$.

Our goal is to learn an accurate approximation of the contact deformation function f, which can be applied to generic colliders. To this end, we observe that contact deformation details not captured by the reduced model are local, and hence only need local information of the shape of the collider $\mathcal{Z}$ in the vicinity of the point of interest $\tilde{x}$. Generally, we wish to construct a collider descriptor $\hat{\mathcal{Z}}(\tilde{x}, \mathcal{Z}, T)$, which depends on the point of interest and the transformed shape of the collider. Then, our contact deformation can be generally expressed as $u(\bar{x}) = f(\hat{\mathcal{Z}}, \mathbf{q}, \tilde{x}, T)$, i.e., a function of the collider descriptor $\hat{\mathcal{Z}}$, the deformed state $\mathbf{q}$ of the object, the point of interest $\tilde{x}$, and the rigid transformation of the collider T.

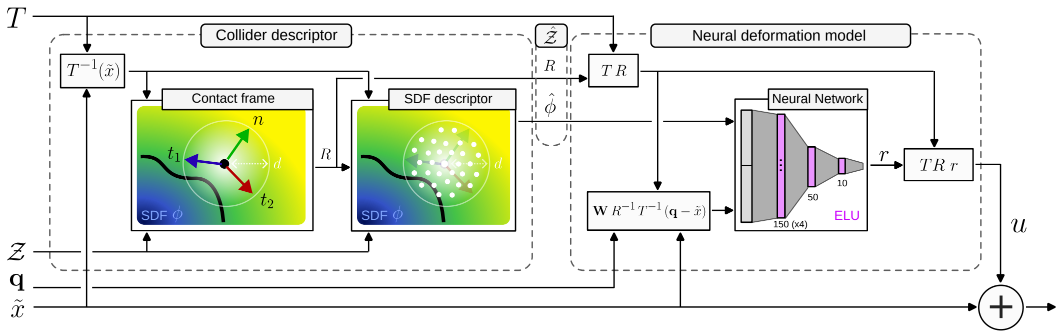


Figure 6.2: Summary of the computational pipeline of our contact deformation model. The pipeline has two parts: (left) the evaluation of a local collider descriptor, and (right) the evaluation of the neural deformation. The collider descriptor takes as runtime input the shape of the collider $\mathcal{Z}$ (characterized by its SDF ϕ), the rigid transformation of the collider T, the deformed state of the colliding object $\mathbf{q}$, and the deformed point of interest $\tilde{x}$. The evaluation of the collider descriptor includes two parts: a contact frame R and a local SDF descriptor $\hat{\phi}$. Then, the deformed point of interest $\tilde{x}$ is transformed to the local reference of the contact frame, it is weighted by sparsifying weights $\mathbf{W}$, and the result is input to the neural deformation model together with the local SDF descriptor. As a result, we obtain a local contact deformation r that is then transformed back to world space and added to $\tilde{x}$.

To build a local collider descriptor, we choose a radius of influence d around the point of interest $\tilde{x}$. Since large deformations are captured by the underlying reduced model, we choose d as the average rest-shape distance between handles of the reduced model. In the rest of the chapter, we refer as d-ball to a sphere of radius d.

Our choice of collider descriptor includes two components, $\hat{\mathcal{Z}} = \{R, \hat{\phi}\}$. $R \in SO(3)$ is a local contact frame, and $\hat{\phi}$ is a local descriptor of the signed-distance-field (SDF) of the collider. In the following subsections, we provide more detail about both components R and $\hat{\phi}$.

6.2.2 Contact Frame

Thanks to a local contact frame R, we can design a neural deformation model in local coordinates of the contact, and then transform this deformation to world space:

$$u(\tilde{x}(\bar{x})) = T\,R(\tilde{x})\,r(\tilde{x}). \tag{6.1}$$

As shown in Chapter 5, a contact-centric representation increases the learning ability of contact deformations. However, unlike this previous method, which used simply the

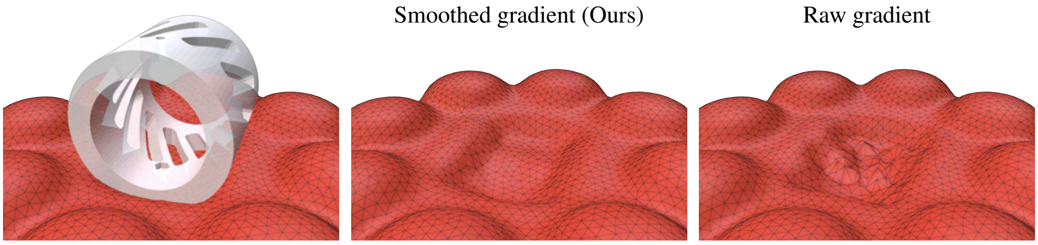


Figure 6.3: This collider with holes (left) produces many SDF gradient discontinuities. Using the raw gradient to define the contact frame (right) makes learning difficult, while our smoothed gradient (middle) helps learning.

transformation T of the collider and did not generalize to arbitrary colliders, we fit the frame R to the local shape of the collider, thus avoiding a collider-specific choice.

To fit the frame R, we separate the computation of a normal axis n from the computation of two tangent axes t_1 and t_2. For the tangent axes, we use different policies during training and during runtime inference, as discussed below.

Normal Axis. To fit the normal axis, we analyze the surface geometry of the collider in the vicinity of the point of interest $\tilde{x}$. The SDF ϕ of the collider serves as representation of this local geometry, hence we fit n based on the gradient of the SDF. However, as opposed to just evaluating the SDF gradient at $\tilde{x}$, we compute a smooth approximation of the gradient operation on the d-ball centered at $\tilde{x}$. This provides a continuous and smooth normal for the frame, which simplifies the learning task as shown in Fig. 6.3.

Given a triple of orthogonal $\mathbb{R}^3$ vectors $B = \{b_1, b_2, b_3\}$, we define a finite-difference approximation of the SDF gradient as:

$$\nabla_B \phi(\tilde{x}) = \sum_{i=1}^{3} \frac{\phi(T^{-1}(\tilde{x}) + b_i) - \phi(T^{-1}(\tilde{x}) - b_i)}{2\,\|b_i\|^2}\, b_i. \tag{6.2}$$

Note that the SDF ϕ is constant in local coordinates of the rigid collider, hence the point of interest $\tilde{x}$ is first transformed to the local reference system of the collider. Given the finite-difference approximation above, we define a smooth approximation $\tilde{\nabla}\phi(\tilde{x})$ of the SDF gradient by sampling vector triples B_j on the d-ball at $\tilde{x}$:

$$\tilde{\nabla}\phi(\tilde{x}) = \frac{1}{N}\sum_{j=1}^{N} \nabla_{B_j}\phi(\tilde{x}). \tag{6.3}$$

To generate the random vector triples B_j, we sample directions uniformly and we sample lengths from a normal distribution with standard deviation d.

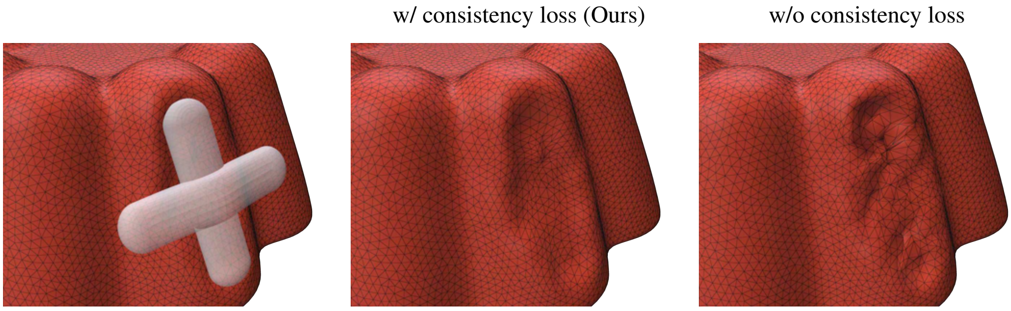


Figure 6.4: Even under smooth colliders (left), our consistency loss is necessary to ensure robustness with respect to tangent rotations of the contact frame (middle); deformations are not correctly learned without this loss (right).

Finally, we normalize the smoothed gradient to obtain the normal of the frame R, $n = \frac{\tilde{\nabla}}{\|\tilde{\nabla}\|}$.

Tangent Axes during Training. To make the deformation model r robust to relative transformations between the deformable object and the collider, we train it for random tangent rotations, but we enforce consistency of the output. To generate random rotations, we draw the tangent axis t_1 from a uniform distribution of orientations normal to n, and we define $t_2 = n \times t_1$. To enforce output consistency, we generate several random frames R for each point of interest $\tilde{x}$, and we add an explicit consistency loss term that penalizes the difference in the resulting contact deformation u for all these random frames. Fig. 6.4 compares the result with and without the consistency loss. Please see more details about model training in Section 6.2.4.

As an alternative to random rotations, we explored aligning the tangents t_1 and t_2 to principal directions of the SDF, similar to how we align the normal n to the gradient of the SDF. However, to deal with SDF discontinuities and locally flat regions, we found it was necessary to apply a similar randomization followed by smoothing. We did not see an improvement in quality despite the extra cost of computing principal directions through SVD, hence we opted for fully random tangent rotations.

Tangent Axes at Runtime. While random sampling of tangent axes makes the model robust to rotations during training, it adds some flickering during runtime inference. As an alternative, at runtime we opt to bias model inference, to achieve both temporal and spatial coherence, as demonstrated in Fig. 6.5. We pick arbitrary tangent directions as $t_1 = \frac{n\times(1,0,0)}{\|n\times(1,0,0)\|}$ and $t_2 = n \times t_1$.

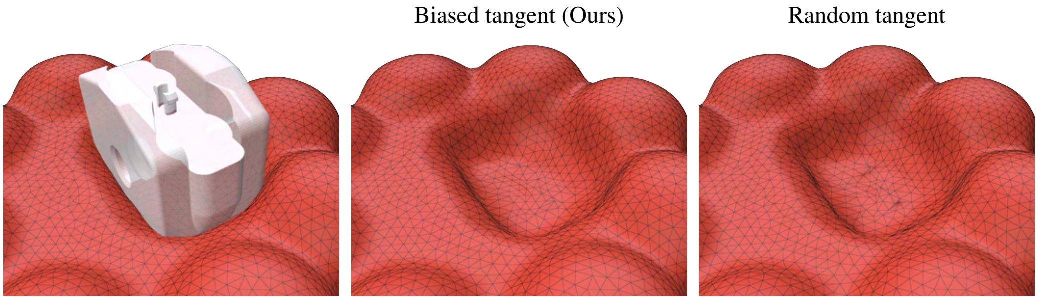


Figure 6.5: For runtime inference, we bias the tangent rotation of the contact frame to maximize spatial and temporal coherence (middle). Using random rotations at runtime leads to noise (right).

When n is exactly aligned with the tangent bias direction $(1, 0, 0)$, there is a singularity, and the result naturally defaults to a random choice of tangent frame. However, this situation is so rare, and the possible flickering due to the random frame so small, that the combined effect is imperceptible. There is a robust solution to the singularity, based on evaluating the model with two tangent bias directions (e.g. $(1, 0, 0)$ and $(0, 1, 0)$), and then interpolating the result (e.g. based on the angle between the normal and the bias directions). However, this approach doubles the inference cost, with almost identical results. Therefore, we discard this option in practice.

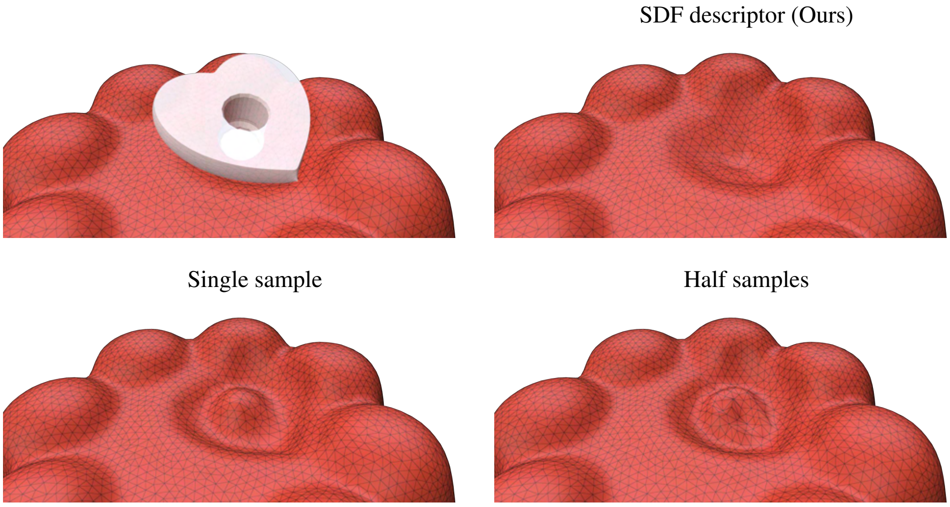
6.2.3 Local SDF Descriptor

We seek a descriptor of the local shape of the collider in the vicinity of the point of interest $\tilde{x}$. To this end, we resort again to the SDF ϕ of the collider, but this time we look at the complete SDF within the d-ball at $\tilde{x}$.

We construct the descriptor $\hat{\phi}$ by concatenating the values of the SDF ϕ on a finite number of samples $\{w_i \in \mathbb{R}^3\}$ around $\tilde{x}$, scaled by d and oriented according to the local contact frame R.

$$\hat{\phi} = \{\phi(T^{-1}(\tilde{x}) + R\, d\, w_i)\}. \tag{6.4}$$

We choose the (normalized) sampling pattern $\{w_i\}$ to be the cubature rule for degree-7 integration inside a sphere (which yields 64 samples) [Str71], plus a sample at the center. By orienting the sampling pattern according to the frame R, we achieve invariance of the descriptor with respect to rotations of the collider normal. Recall that R does not provide rotation invariance on the tangent plane, but the randomized training discussed in Section 6.2.2 makes the model robust to rotation.



SDF descriptor (Ours)

Single sample

Half samples

Figure 6.6: This collider has a cylindrical void, not a hole passing all the way through (top left). Our local SDF descriptor correctly captures the deformation produced by the flat collider face, unaffected by the void (top right). When using a single SDF sample (bottom left) or half the samples as descriptor (bottom right), local shape is not correctly represented, and the void erroneously affects the deformation.

Fig. 6.6 demonstrates how the local SDF descriptor provides relevant local shape information for the computation of contact deformations. We compare our descriptor to using (a) $\hat{\phi} = \phi(T^{-1}(\tilde{x}))$, i.e., just the SDF at the point of interest $\tilde{x}$, and (b) $\hat{\phi} = \{\phi(T^{-1}(\tilde{x}) + R\,d\,w_i),\ \ w_i^T\,(1,0,0) \leq 0\}$, i.e., only half of the sampling pattern along $-n$.

6.2.4 Neural Deformation Model

Given both components R and $\hat{\phi}$ of the collider descriptor $\hat{\mathcal{Z}}$, we can formulate a neural model to compute the local contact deformations r in (6.1). We input to the model the local SDF descriptor $\hat{\phi}$ and the deformation of the object $\mathbf{q}$. Again, we adapt from Chapter 5 the definition of the deformation input in local coordinates of the contact, but we use the local contact frame R, which generalizes to arbitrary colliders, as opposed to their global collider frame T, which is collider-specific. We also follow Chapter 5 in using sparsifying weights $\mathbf{W}(\bar{x}) = \text{diag}(\mathbf{U}(\bar{x}))$, where $\mathbf{U}$ denotes the basis of the deformation model. This sparsification is a benefit of the handle-based reduced model.

Altogether, the neural model of contact deformation $\mathbb{N}$ is formally defined as:

$$r(\tilde{x}(\bar{x})) = \mathbb{N}\left(\hat{\phi}(\tilde{x}), \mathbf{W}(\bar{x})\left(R(\tilde{x})^{-1}\,T^{-1}\left(\mathbf{q} - \tilde{x}\right)\right)\right). \tag{6.5}$$

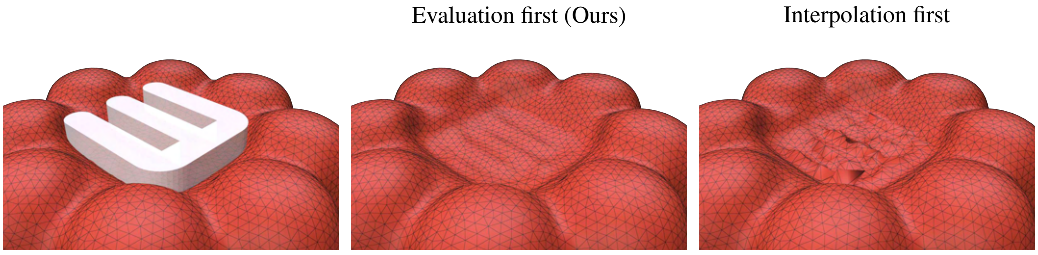


Figure 6.7: Given a precomputed grid of collider descriptor values, we fetch the descriptor values from grid points, evaluate the neural contact deformation model, and then interpolate the deformation result (middle). Interpolating the descriptor and then evaluating the model only once saves cost, but suffers strong noise (right).

To express the deformation input $\mathbf{q}$ in local coordinates of the contact frame R, again we leverage the handle-based discretization of our reduced deformation model. For point handles in $\mathbf{q}$, which define a translation, we use the transformation to R as expressed in (6.5) above. For frame handles, we use the same transformation for the translation part, and we simply omit the subtraction $-\tilde{x}$ for the rotation part.

Network Structure. In our experiments, we use a simple neural network structure to parameterize $\mathbb{N}$. Specifically, we use networks of 6 layers, with (150,150,150,150,50,10) neurons in each hidden layer, fully connected and with ELU as activation function.

Training of the Network. We train the model $\mathbb{N}$ for a particular deformable object in a supervised manner, providing ground-truth data of contact deformations for a variety of collider objects n_o, reduced object deformations n_d, and contact configurations n_c. The major loss term simply compares the error between predicted and ground-truth values of contact deformations. As already mentioned in Section 6.2.2, we add a self-supervised consistency loss for robust handling of the tangential orientation of the contact frame.

For each collider object, we start by executing an interactive simulation between the reduced deformable object and the collider, and we pick the n_d most distant deformations of the deformable object. For the rest of the ground-truth data generation, we freeze the reduced deformed state, and we execute full-space simulations constrained to this reduced state. We sample n_c contact configurations, which include a contact location on the surface of the deforming object, a relative transformation of the collider, and a collision depth. This procedure is similar to the one described in 5.3.2.

For each training deformation, we include in the supervision data a random set of points of interest $\tilde{x}$. To select these points, we consider the interior of the collider object together

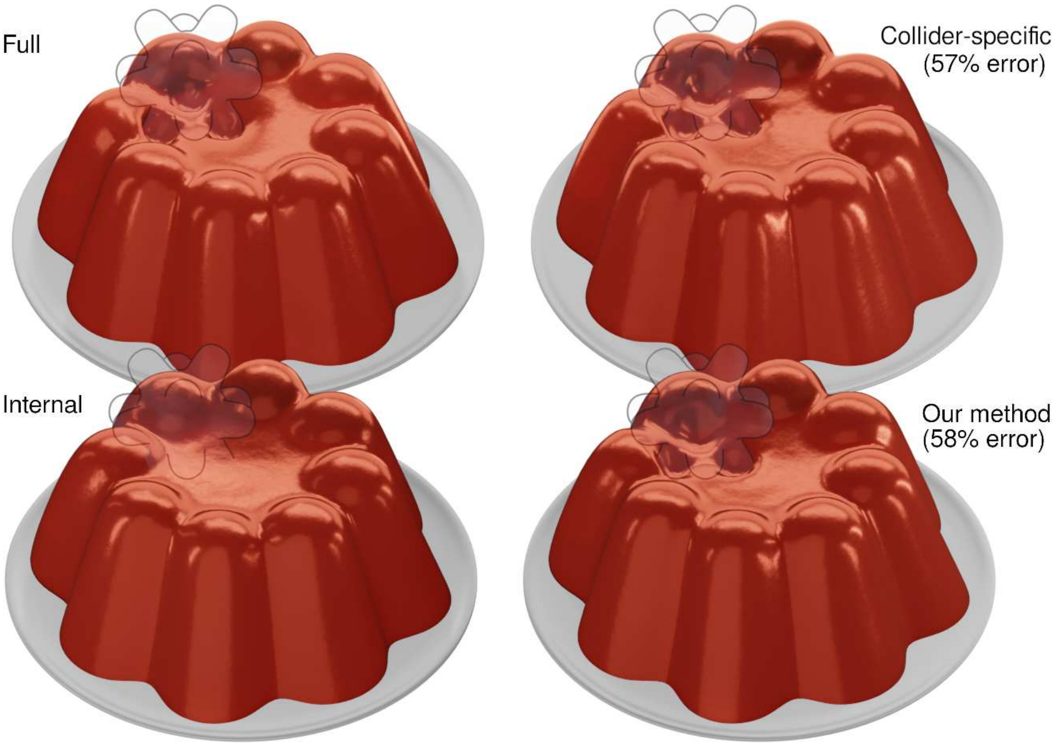


Figure 6.8: We compare our method to our collider-specific method presented in Chapter 5, and we achieve on par accuracy. We use as baseline for comparison the same reduced model with data-driven internal corrections.

with points within a d-ball from the collider surface, but we sample with lower probability points outside the collider, according to a normal distribution with standard deviation d. Furthermore, we force ground-truth deformations to attenuate toward 0 for points of interest at distance d from the surface of the collider, to ensure that the trained model does not suffer discontinuities.

6.3 Runtime Model Evaluation

To evaluate the contact deformation model at runtime, we propose an acceleration by precomputing the collider descriptor for each collider. In this section, we first discuss this precomputation on a grid, and then we describe the runtime interpolation from the grid to arbitrary points.

6.3.1 Precomputation of the Collider Descriptor

As described in the previous section, the central ingredient of our contact deformation model is a collider descriptor $\hat{\mathcal{Z}} = \{R, \hat{\phi}\}$. While the neural deformation model $\mathbb{N}$ is trained on a library of colliders for a specific deformable object, the collider descriptor $\hat{\mathcal{Z}}$ is specific to each collider and agnostic of the deformable object. Following this observation, we can leverage per-collider precomputation of the collider descriptor. This approach brings an interesting consequence for application development. To use an arbitrary new collider with a given deformable object, we just need to execute a quick precomputation of the collider descriptor.

Recall that the collider descriptor $\hat{\mathcal{Z}}$ is evaluated at points of interest $\tilde{x}$, which are transformed through T^{-1} to the local reference system of the collider. Therefore, we choose to precompute $\hat{\mathcal{Z}}$ (both the contact frame R and the local SDF descriptor $\hat{\phi}$) on a grid surrounding the collider. We set up the grid on the bounding box of the collider enlarged by the ball radius d. In our experiments, we have used regular grids, with size 50 along the largest dimension for small colliders, and 90 for large colliders (in the finger and soft-ball examples, see Section 6.4). Finally, we cull the precomputation of the descriptor on points that are further than d from the surface of the collider. In our experiments, we obtained speed-ups between $1.6\times$ and $3.3\times$ (larger as d grows) when precomputing the collider descriptor.

Note that we only precompute the collider descriptor for runtime inference. As discussed in Section 6.2.2, for training we randomize the contact frame R, and hence we cannot leverage precomputation.

6.3.2 Grid Interpolation

Given collider descriptors at grid points, we need to evaluate the contact deformation model at arbitrary points of interest $\tilde{x}$. To do this, we transform $\tilde{x}$ to the local reference system of the collider, locate its grid cell, fetch the collider descriptor at all 8 grid corners, evaluate the neural contact deformation model $\mathbb{N}$ at the grid corners, and trilinearly interpolate the deformations. First interpolating the collider descriptor and then evaluating the neural model just once is more efficient, but far less robust due to the interpolation of frames, as shown in Fig. 6.7.

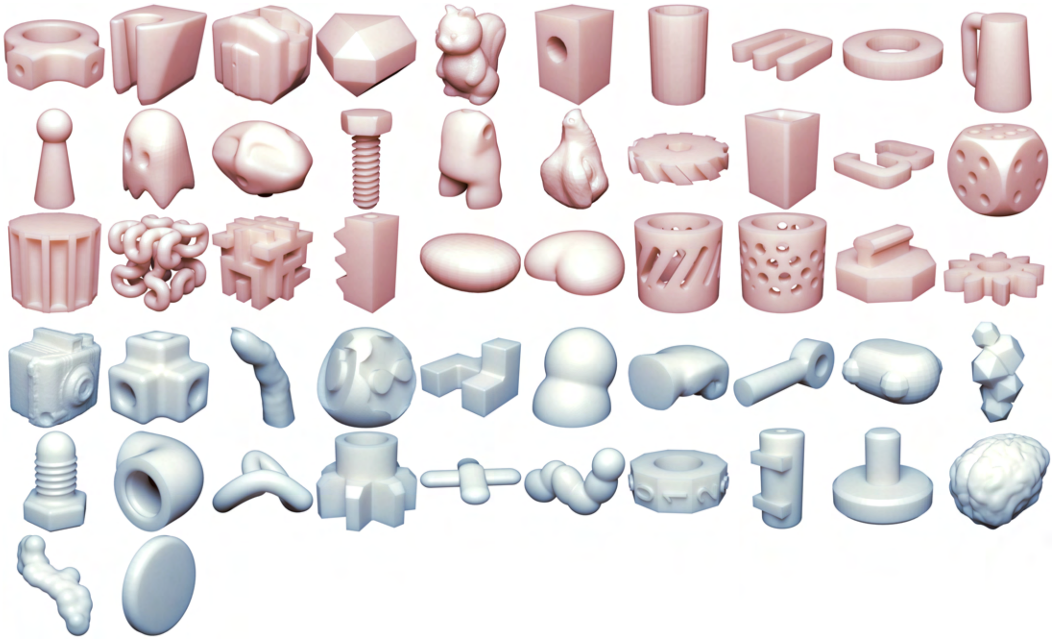


Figure 6.9: Collider objects (all from Thingi10K [ZJ16]) used for training (top, in red) and quantitative testing (bottom, in blue) our deformation model. Average error of our model is 17.0% on the train colliders and 17.8% on the test colliders, demonstrating its generalization capabilities.

Table 6.1: Complexity of the deformable objects used in the experiments, their training data set size (see Section 6.2.4), train and test error, and runtime performance.

Object	Handles point/bone	Tets	# Samples n_o	n_d	n_c	Error % train	test	fps linear	full	ours
Ball	15/0	29,244	30	1	150	20.8	21.0	129	1	9
Hand	0/16	82,395	30	1	525	17.1	16.8	81	1	26
Jelly	18/1	60,830	30	5	105	16.2	17.9	200	1	24
Finger	7/2	31,218	30	1	32	14.0	15.5	142	1	11

6.4 Experiments

6.4.1 Objects and Accuracy

Fig. 6.9 shows the 30 colliders used for training and the 22 colliders used for quantitative testing in our experiments. All collider objects are part of the Thingi10K library [ZJ16], and they were selected to represent a variety of smooth and sharp shapes, voids and holes, and organic and synthetic objects. Note that in some experiments we use scene-specific colliders too, but they were not used for quantitative testing.

Table 6.1 summarizes the deformable objects used in our experiments. For each object, we indicate its reduced-model and mesh complexity, the number of objects $\times$ deformations $\times$ contact samples used for training, the train and test error, and runtime performance. To quantify error, we define as 100% error the mean square of ground-truth $u(\tilde{x})$, i.e., the difference in deformation between the underlying linear reduced model and the full-space deformation. As shown in the table, the average error with train colliders is 17.0%, and it barely grows to 17.8% with test colliders, demonstrating the generalization capabilities of the proposed model.

We have also compared the accuracy of our model to the collider-specific method presented in Chapter 5. The Jelly object was trained in Chapter 5 with a pointy collider (see Fig. 6.8), reporting 57% error on a contact interaction sequence. With our model, even if the pointy collider is not used for training, error barely changes to 58%.

6.4.2 Dynamic Simulation and Performance

In our experiments, we show the contact deformation model fully embedded in dynamic simulations. We formulate and solve the simulations using the optimization version of backward Euler [Kan*00], which boils down to solving for the deformation DoFs that minimize an energy $\Psi(x(\mathbf{q}))$ subject to the total deformation given by $x(\mathbf{q}) = \tilde{x}(\mathbf{q}) + u(\mathbf{q})$. We use a Newton solver with just one iteration per time step and conjugate gradient for the linear solve. We compute gradients $\frac{\partial \Psi}{\partial x} \frac{\partial x}{\partial \mathbf{q}}$, which requires back propagation of the neural model, and we approximate Hessian-vector products in the linear solve as $\frac{\partial x}{\partial \mathbf{q}}^T \frac{\partial^2 \Psi}{\partial x^2} \frac{\partial x}{\partial \mathbf{q}} \cdot \mathbf{v}$, using an auxiliary back propagation as described in Section 4.3.3.

We have executed all our experiments on an Intel Core i7-7700K 4-core 4.20 GHz PC with 32 GB of RAM, and we have implemented the neural networks using LibTorch on C++/CUDA. Table 6.1 summarizes the performance of our experiments. Speed-ups with respect to full-space simulations range between $9\times$ and $26\times$.

6.4.3 Examples

In Fig. 6.10, we showcase an example where the dynamics of the rigid collider are also simulated. Deformations and contact are detailed and robust in this case too, and they are excessively smooth for the linear reduced model.

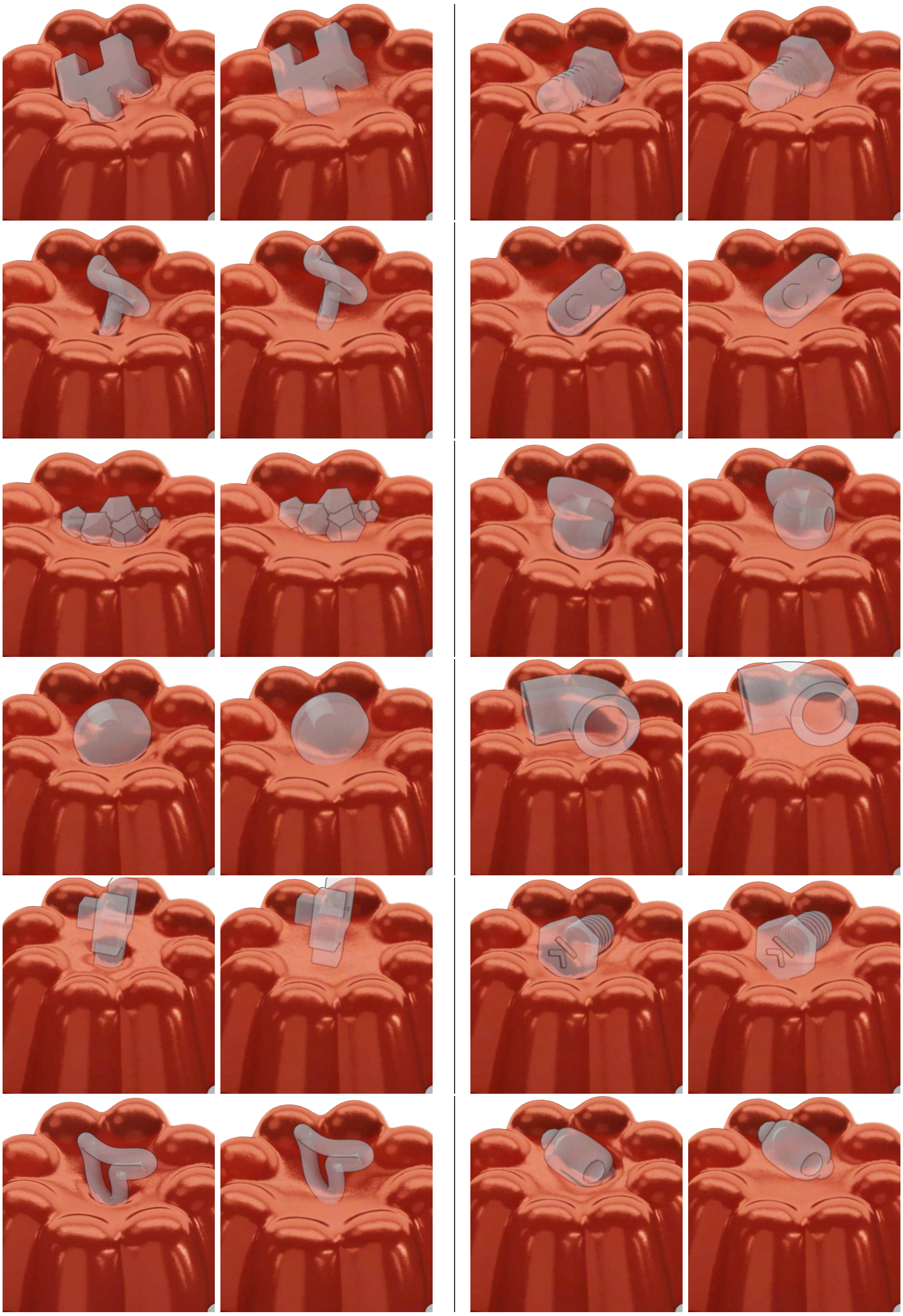


Figure 6.10: Dynamic rigid colliders falling on the jelly. With our model (left of the pairs), the colliders produce detailed contact deformations. With the linear reduced model (right of the pairs), deformations are too smooth.

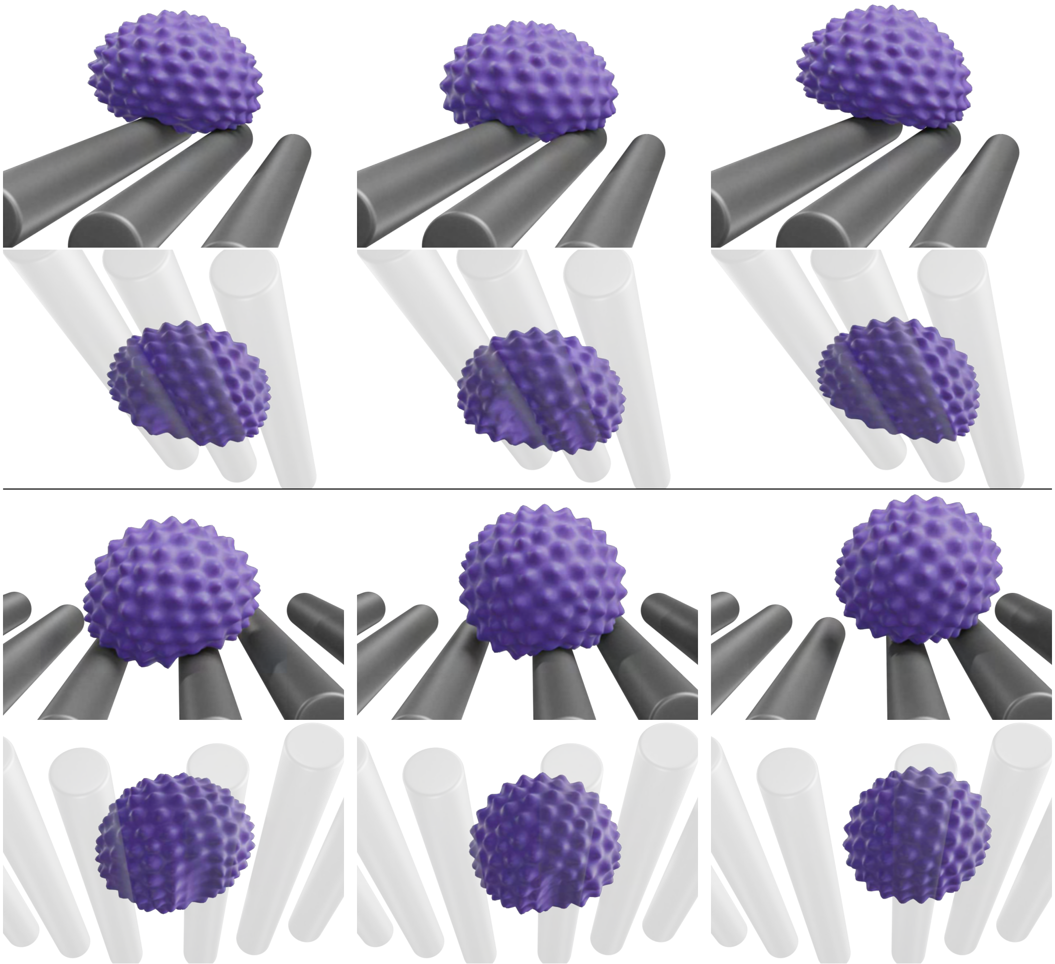


Figure 6.11: The images compare our method (left images), a full simulation (middle images), and the linear reduced model (right images), simulating contact of a soft spiky ball with rigid bars not seen at training. With the linear reduced model, deformations are too smooth and the spikes retain much of their shape. With our model, the bars produce clear indentations, and the spikes collapse under contact, much like with the full model.

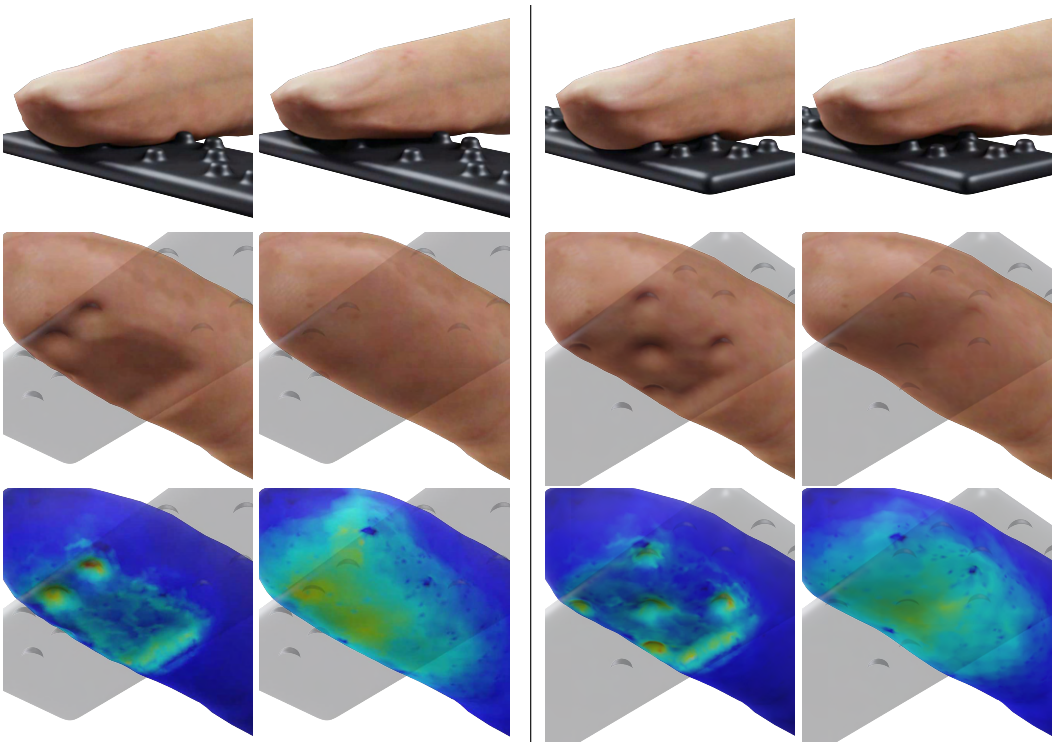


Figure 6.12: A soft finger reading braille. Our model (first and third column) resolves the detailed deformations produced by braille dots, while the linear reduced model (second and fourth column) fails. The bottom images show the norm of Green strain.

In Fig. 6.11, we showcase an example where the soft object, a spiky ball, has surface detail, showing that our method works well for objects whose surface is not smooth. With the linear reduced model, deformations are too smooth and the spikes retain much of their shape. With our model, on the other hand, the bars produce clear indentations, and the spikes collapse under contact, much like with the full model.

Fig. 6.1 shows detailed contact deformations of a hand model exploring interactively different objects. None of the colliders in this example were used for training. This example also demonstrates the applicability to fine tactile exploration of objects in e-commerce applications, where the objects can be readily used for simulation without object-specific precomputation.

Finally, Fig. 6.12 demonstrates the applicability of our method for virtual touch problems. A soft finger model reads the word “touch” written in braille. While the linear reduced model fails to resolve the deformations produced by braille dots, our model produces highly detailed deformations. Note that the braille collider is far from those used for training. The high-resolution strain field is also suitable for driving high-fidelity haptics [VCO20].

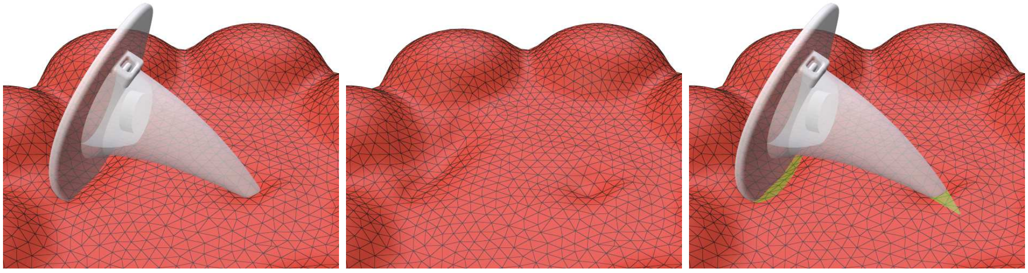


Figure 6.13: The quality of the deformations produced by our model is limited by the sampling resolution of the descriptor and the mesh resolution of the input deformation dataset. For this hat collider with a pointy end (left), our model fails to resolve a deformation with sufficient detail (middle), resulting in noticeable intersections (right).

6.5 Conclusions

We have introduced a learning-based method for the computation of highly detailed contact deformations. Our method overcomes a major scalability limitation of previous works, and generalizes to colliders of arbitrary shape, by learning deformations as a function of local collider shape. Key to this feature was the design of a local shape descriptor as a condition signal for a neural field model.

Despite the novelty of our approach and results, there are still limitations that could inspire future work. In terms of low-level technical limitations, one is that the accuracy of deformations may be limited by the sampling resolution of the SDF descriptor as well as the mesh resolution of the input deformation dataset. Deformations with pointy features are not always resolved correctly, as shown in the example in Fig. 6.13. Another low-level technical limitation is that our model needs to be trained with respect to rotations of the collider. This is for two reasons: the need to represent the result of the model (i.e., a vector quantity, the displacement field) in some reference system, and the definition of the SDF descriptor through sampling aligned with the local geometry. It would be interesting to bake the rotational invariance directly in the neural model, and avoid the construction of an explicit contact frame.

In terms of high-level limitations, our method works only for rigid colliders, and it must be trained independently for each deformable object. It would be interesting to explore models that are conditioned by local shape, material and deformation, thus enabling learning-based contact deformation for arbitrary object-object interactions. Our method also makes the underlying assumption that the deformable object is well described by a reduced number of degrees of freedom, and does not clearly extend to objects whose deformation is inherently high-dimensional (e.g., cloth that can be folded arbitrarily).

7 Conclusions

In this last chapter, we review the most relevant contributions of this work, also discussing open challenges to be explored and promising ideas for the future. Finally, a more general overview of the research conducted during this thesis is provided, along with some insights of the process.

7.1 Discussion

In this thesis we have presented important contributions for the simulation of deformable objects, by introducing new simulation models for soft avatars, reduced deformation and contact. In the process, we have solved some interesting challenges to achieve expressive, efficient and general deformation methods.

One of the most challenging aspects of modeling deformable objects is replicating their potentially complex behavior using simple and efficient models, and in this regard, the human body is a specially challenging system. Being of central importance for many applications, in this thesis we have dedicated special attention to efficiently modeling the deformation of human soft-tissues, successfully validating our methods. In Chapter 3 we have presented an accurate soft-avatar model with full body dynamic deformations. In Chapters 4 ,5 and 6, we apply our novel contact deformation methods to demonstrate the applicability in specifically relevant body parts such as hands (Fig. 5.1 and Fig. 6.1) or fingers (Fig. 4.12 and Fig. 6.12). Not just for visual purposes, our methods can be also applied for controlling haptic devices and applications where force feedback is required. As opposed to surface models, we can easily calculate stresses inside the soft-tissue, thanks to our volumetric description (colormaps in Fig. 6.12).

Nevertheless, there is still room for improvements when capturing the deformation of complex objects. On one hand, finding constitutive models with more flexible parameterizations will further increase the expressiveness and the ability to describe real world behaviors, e.g. using novel material models for homogenization or using multiple heterogeneous soft-tissue layers. On the other hand, to estimate a large number of model parameters from real data, we still need to support efficient differentiability and gradient computations. It would even be interesting to explore ways for an efficient second-order optimization of properties.

Another important challenge that we have addressed is the slow runtime performance of high-quality physical simulations. To overcome this limitation, we design fast reduced simulation methods with data-driven deformation corrections. Our choice of reduced model provides proper dynamic behaviors by construction, something challenging to achieve in pure data-driven methods, even if they are designed for time dynamics [Hol*19]. In addition, our methods use data-driven models and neural networks to substitute complex deformation relationships, building from captured [Lop*15] or simulated example data. This avoids the need for a comprehensive physics-based model and gives us much more room for efficient approximations.

Our reduced models are efficient, but still have some limitations. We use handle-based subspaces, which are particularly useful thanks to their affine transformation properties and geometric nature. However, currently there is some manual work involved when selecting the location of the handles on the object. In this context, designing an automatic handle placement and selection approach can be a useful improvement. Also, instead of using the simple geometric smoothness criteria for building our biharmonic handle subspaces [Wan*15], it might be a good idea to take into account the material and energy properties of the deformable object. Finally, our focus has been mostly on reducing the number of simulated DoFs by improving the kinematics of reduced deformation models, but force computation and cubature can also be improved with data-driven techniques, specially for contact, where we are not using any cubature approximation. In this line it would be interesting to explore also contact-centric alternatives to improve the efficiency of contact force computations.

The improvements presented by our methods are largely based on novel data-driven deformation techniques. We have devoted much of this thesis to improving their generalization capabilities, reduce the overfitting, and achieve more practical and less data-intensive models. To a large extent, this was made possible thanks to our efficient representations and generalized descriptors for learning arbitrary contact deformations. We must emphasize also the importance of our combined data-driven and reduced subspace simulations, with a significant contribution to the generalization. Same as reduced simulations, our models can solve general time dynamics, and can be easily coupled to other physical models for general interaction. e.g. bidirectional coupling with simulated cloth (Fig. 3.6), bidirectional coupling with simulated rigid body colliders (Fig. 6.10), or external force-based interactions (Fig. 4.10).

Our novel data-driven methods reduce drastically the amount of required data. Even so, in Chapters 4, 5, and 6, most of the computational resources are used for the generation of contact deformation examples. Motivated by the reduction of preprocessing times, it may worth exploring a self-supervised approach for learning contact deformations following recent works in this line [SOC22; BME22]. In a complementary way, the design of improved sampling strategies for collider configurations can help in reducing the required number of representative deformation examples. Also, the sampling of configurations and corresponding data generation are currently performed as separate processes. However, as a more exploratory path, it might be interesting to perform both actions simultaneously. This could generate a more physically informed sampling process reducing even more the number of required examples.

Finally, as an interesting option, it would also be possible to train our contact models with real contact deformation data, instead of using simulated examples as we currently do. However, to be useful at runtime, this would require an estimation of the material parameters as well, unlike our methods presented in Chapters 4, 5, and 6, where the material properties are explicitly assigned. With appropriate data capture systems and using dynamic and static deformation examples, the material estimation method presented in Chapter 3 can be combined with our contact deformation learning methods.

Overall, in this thesis we have introduced novel methods for the simulation of deformable objects, successfully combining data-driven techniques and physically-based simulations. As discussed in this chapter, there are still many aspects to be explored, but I hope that the presented work helps to bridge the gap between precise and interactive models, paving the way towards general methods that allow us to simulate efficiently and accurately the behavior of our own bodies and the deformable objects around us.

7.2 General Insights

The methods presented in this thesis are the result of five years of dedicated research. In this process, we have explored different ideas, discarding some of them, but also corroborating interesting insights as we moved forward with our discoveries. Looking back, there are some guiding concepts that I have been internalizing during this journey.

One of these concepts is the importance of exploiting the particularities of each problem to achieve the desired objectives, choosing the proper representation for each problem. It might sound trivial, but it has been critical in our decision-making process, moving us forward in our research. Specially in the context of our efficient simulations, this has been a key insight for shaping the models and achieving our objective of interactive methods. I think this also applies when taking inspiration from ideas originally designed for different engineering applications. They can be a productive starting point, but to be truly effective and useful, they require a careful rethinking process to adapt them to our problems in computer graphics. I believe that keeping this idea in mind during our research can helps as a compass.

Another relevant concept that has been gaining momentum in my mind during these years is the productive interplay between physics and data. Physical insights provide us efficient ways to build preliminary models for a given phenomenon. However, first principles are difficult to apply in the description of complex systems, and only go up to a certain level before resulting in very complex and convoluted models. One example is the complexity of different materials and soft tissues necessary to model the human avatar of Chapter 3 using a pure physics-based method. However, using data and data-driven representations we can complement simple and efficient physical models. When doing so, we tend to lose some interpretability, but this can be a good entry point to start playing with the problem. Building data-driven models can help understanding the problems, and I will risk to say that it can even help to design new physical models. In fact, this is very similar to what experimental scientists do.

This thesis is a clear example of the described interplay. Our goal of improving the generalization of data-driven methods has required us to understand and use the physics and geometry of the problem. For example, our initial attempt in Chapter 4 for a contact deformation model operates as a black box, mapping collider configurations to the expected deformations. Then, thanks to the lessons learned and after understanding the geometric aspects of contact deformations, in Chapter 5 we changed to a new contact-centric representation, augmenting the interpretablility of our model and reducing the data requirements. Similarly, in Chapter 6 we went further in this direction, this time realizing also about the importance of the local collider shape in contact deformations, obtaining an even more general method, and requiring even less data. Will this path lead perhaps to an effective contact model based just on physics and geometry? Whether this is achieved or not, it is an interesting direction to continue exploring.

Looking ahead to the future of computer graphics and the efficient modeling of deformable object, I think the ideas described in this section are going to be more relevant than ever. In the next years, new methods may adopt novel representations to describe deformations, and the journey will continue exploring the interplay between data-driven methods and physics. Of course, as in any good adventure, this one will also be full of obstacles, but if I have discovered something during these past five years, it is the extraordinary personal reward you get after experiencing such adventures.

Bibliography

[Agu*10] Edilson de Aguiar, Leonid Sigal, Adrien Treuille, and Jessica K. Hodgins. “Stable Spaces for Real-Time Clothing”. In: *ACM Trans. Graph.* 29.4 (2010).

[Aig*22] Noam Aigerman, Kunal Gupta, Vladimir G. Kim, et al. “Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes”. In: *ACM Trans. Graph.* 41.4 (2022).

[AHS03] Irene Albrecht, Jörg Haber, and Hans-Peter Seidel. “Construction and animation of anatomically based human hand models”. In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association. 2003, pp. 98–109.

[AXS21] Thiemo Alldieck, Hongyi Xu, and Cristian Sminchisescu. “imGHUM: Implicit Generative Models of 3D Human Shape and Articulated Pose”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 5461–5470.

[All*03] Brett Allen, Brian Curless, Brian Curless, and Zoran Popović. “The Space of Human Body Shapes: Reconstruction and Parameterization from Range Scans”. In: *ACM Transactions on Graphics (Proc. SIGGRAPH)* 22.3 (July 2003), pp. 587–594.

[ACP02] Brett Allen, Brian Curless, and Zoran Popović. “Articulated Body Deformation from Range Scan Data”. In: *ACM Transactions on Graphics (Proc. SIGGRAPH)* 21.3 (2002), pp. 612–619.

[AKJ08] Steven S. An, Theodore Kim, and Doug L. James. “Optimizing Cubature for Efficient Integration of Subspace Deformations”. In: *ACM Trans. Graph.* 27.5 (2008).

[Ang*05] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, et al. “SCAPE: Shape Completion and Animation of People”. In: *ACM Trans. Graph.* 24.3 (July 2005), pp. 408–416.

[AB03] Uri M. Ascher and Eddy Boxerman. “On the Modified Conjugate Gradient Method in Cloth Simulation”. In: *Vis. Comput.* 19.7-8 (2003).

[AL20] Matan Atzmon and Yaron Lipman. “SAL: Sign Agnostic Learning of Shapes from Raw Data”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2020.

[Aum*22] Tristan Aumentado-Armstrong, Stavros Tsogkas, Sven Dickinson, and Allan D. Jepson. “Representing 3D Shapes With Probabilistic Directed Distance Fields”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 19343–19354.

[Bai*18] Stephen W. Bailey, Dave Otte, Paul Dilorenzo, and James F. O’Brien. “Fast and Deep Deformation Approximations”. In: *ACM Trans. Graph.* 37.4 (2018).

[BJ05] Jernej Barbič and Doug L. James. “Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models”. In: *ACM Trans. Graph.* 24.3 (July 2005), pp. 982–990.

[BZ11] Jernej Barbič and Yili Zhao. “Real-Time Large-Deformation Substructuring”. In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH ’11. Vancouver, British Columbia, Canada: Association for Computing Machinery, 2011.

[BS19] Adam W. Bargteil and Tamar Shinar. “An Introduction to Physics-Based Animation”. In: *ACM SIGGRAPH 2019 Courses*. SIGGRAPH ’19. Los Angeles, California: Association for Computing Machinery, 2019.

[Bat*16] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray kavukcuoglu. “Interaction Networks for Learning about Objects, Relations and Physics”. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016, pp. 4509–4517.

[Ben*22] Nuri Benbarka, Timon Höfer, Hamd ul-Moqeet Riaz, and Andreas Zell. “Seeing Implicit Neural Representations As Fourier Series”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2022, pp. 2041–2050.

[BME22] Hugo Bertiche, Meysam Madadi, and Sergio Escalera. “Neural Cloth Simulation”. In: *ACM Trans. Graph.* 41.6 (2022).

[Bha*20] Bharat Lal Bhatnagar, Cristian Sminchisescu, Christian Theobalt, and Gerard Pons-Moll. “LoopReg: Self-supervised Learning of Implicit Surface Correspondences, Pose and Shape for 3D Human Mesh Registration”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Dec. 2020.

[Bic*08] Bernd Bickel, Manuel Lang, Mario Botsch, Miguel A. Otaduy, and Markus Gross. “Pose-Space Animation and Transfer of Facial Details”. In: *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2008, pp. 57–66.

[Bog*17] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. “Dynamic FAUST: Registering Human Bodies in Motion”. In: *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*. Piscataway, NJ, USA: IEEE, July 2017.

[BEH18] Christopher Brandt, Elmar Eisemann, and Klaus Hildebrandt. “Hyper-Reduced Projective Dynamics”. In: *ACM Trans. Graph.* 37.4 (2018).

[Cai*22] Xinhao Cai, Eulalie Coevoet, Alec Jacobson, and Paul Kry. “Active Learning Neural C-space Signed Distance Fields for Reduced Deformable Self-Collision”. In: *Graphics Interface 2022*. 2022.

[Cap*05] Steve Capell, Matthew Burkhart, Brian Curless, Tom Duchamp, and Zoran Popović. “Physically Based Rigging for Deformable Characters”. In: *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’05. Los Angeles, California: Association for Computing Machinery, 2005, pp. 301–310.

[Cap*02] Steve Capell, Seth Green, Brian Curless, Tom Duchamp, and Zoran Popović. “Interactive Skeleton-Driven Dynamic Deformations”. In: *ACM Trans. Graph.* 21.3 (July 2002), pp. 586–593.

[CO18] Dan Casas and Miguel A Otaduy. “Learning nonlinear soft-tissue dynamics for interactive avatars”. In: *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1.1 (2018), p. 10.

[Cha*20] Rohan Chabra, Jan E. Lenssen, Eddy Ilg, et al. “Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction”. In: *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX*. Glasgow, United Kingdom: Springer-Verlag, 2020, pp. 608–625.

[Cha*17] Michael B. Chang, Tomer Ullman, Antonio Torralba, and Joshua B. Tenenbaum. “A Compositional Object-Based Approach to Learning Physical Dynamics”. In: *Proceedings of the International Conference on Learning Representations*. 2017.

[Che*18] Jiong Chen, Hujun Bao, Tianyu Wang, Mathieu Desbrun, and Jin Huang. “Numerical Coarsening Using Discontinuous Shape Functions”. In: *ACM Trans. Graph.* 37.4 (2018).

[Che*21] Peter Yichen Chen, Maurizio Chiaramonte, Eitan Grinspun, and Kevin Carlberg. “Model reduction for the material point method via learning the deformation map and its spatial-temporal gradients”. In: *arXiv preprint arXiv:2109.12390* (2021).

[CZ19] Zhiqin Chen and Hao Zhang. “Learning implicit fields for generative shape modeling”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 5939–5948.

[Che*20] Nuttapong Chentanez, Miles Macklin, Matthias Müller, Stefan Jeschke, and Tae-Yong Kim. “Cloth and Skin Deformation with a Triangle Mesh Based Convolutional Neural Network”. In: *Computer Graphics Forum* 39.8 (2020), pp. 123–134.

[CMP20] Julian Chibane, Aymen Mir, and Gerard Pons-Moll. “Neural Unsigned Distance Fields for Implicit Function Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Dec. 2020.

[Chu*23] Ethan Chun, Yilun Du, Anthony Simeonov, Tomas Lozano-Perez, and Leslie Kaelbling. *Local Neural Descriptor Fields: Locally Conditioned Object Representations for Manipulation*. 2023. arXiv: `2302.03573 [cs.RO]`.

[Cor*21] Enric Corona, Albert Pumarola, Guillem Alenyà, Gerard Pons-Moll, and Francesc Moreno-Noguer. “SMPLicit: Topology-aware Generative Model for Clothed People”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2021.

[Den*20] Boyang Deng, JP Lewis, Timothy Jeruzalski, et al. “NASA: Neural Articulated Shape Approximation”. In: *Proc. of European Conference on Computer Vision (ECCV)*. 2020.

[Dou*15] Mingsong Dou, Jonathan Taylor, Henry Fuchs, Andrew Fitzgibbon, and Shahram Izadi. “3D scanning deformable objects with a single RGBD sensor”. In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 493–501.

[Fan*13] Ye Fan, Joshua Litven, David I. W. Levin, and Dinesh K. Pai. “Eulerian-on-lagrangian Simulation”. In: *ACM Trans. Graph.* 32.3 (2013).

[Fau*11] François Faure, Benjamin Gilles, Guillaume Bousquet, and Dinesh K. Pai. “Sparse Meshless Models of Complex Deformable Solids”. In: *ACM Trans. Graph.* 30.4 (July 2011).

[FCG99] Eric Ferley, Marie-Paule Cani, and Jean-Dominique Gascuel. “Practical Volumetric Sculpting”. In: *The Visual Computer* 16 (Sept. 1999).

[Ful*19] Lawson Fulton, Vismay Modi, David Duvenaud, David Levin, and Alec Jacobson. “Latent-space Dynamics for Reduced Deformable Simulation”. In: Jan. 2019.

[Gal*09] Nico Galoppo, Miguel A. Otaduy, William Moss, et al. “Controlling Deformable Material with Dynamic Morph Targets”. In: *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games*. I3D ’09. Boston, Massachusetts: Association for Computing Machinery, 2009, pp. 39–47.

[Gal*07] Nico Galoppo, Miguel A. Otaduy, Serhat Tekin, Markus Gross, and Ming C. Lin. “Soft Articulated Characters with Fast Contact Handling”. In: *Computer Graphics Forum* 26.3 (2007), pp. 243–253.

[Gas*15] Theodore F Gast, Craig Schroeder, Alexey Stomakhin, Chenfanfu Jiang, and Joseph M Teran. “Optimization integrator for large time steps”. In: *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 21.10 (2015), pp. 1103–1115.

[Gil*11] Benjamin Gilles, Guillaume Bousquet, Francois Faure, and Dinesh K. Pai. “Frame-Based Elastic Models”. In: *ACM Trans. Graph.* 30.2 (Apr. 2011).

[GDY19] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. “Hamiltonian Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019, pp. 15379–15389.

[Gro*20] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. “Implicit Geometric Regularization for Learning Shapes”. In: *Proceedings of Machine Learning and Systems 2020*. 2020, pp. 3569–3579.

[Gue*18] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J. Mitra. “PCPNet: Learning Local Shape Properties from Raw Point Clouds”. In: *Computer Graphics Forum* 37.2 (2018), pp. 75–85.

[Hah*12] Fabian Hahn, Sebastian Martin, Bernhard Thomaszewski, et al. “Rig-Space Physics”. In: *ACM Trans. Graph.* 31.4 (July 2012).

[Hah*13] Fabian Hahn, Bernhard Thomaszewski, Stelian Coros, Robert W. Sumner, and Markus Gross. “Efficient Simulation of Secondary Motion in Rig-Space”. In: *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’13. Anaheim, California: Association for Computing Machinery, 2013, pp. 165–171.

[Hah*14] Fabian Hahn, Bernhard Thomaszewski, Stelian Coros, et al. “Subspace Clothing Simulation Using Adaptive Bases”. In: *ACM Trans. Graph.* 33.4 (July 2014).

[HZ13] David Harmon and Denis Zorin. “Subspace Integration with Local Deformations”. In: *ACM Trans. Graph.* 32.4 (July 2013).

[Hol*19] Daniel Holden, Bang Chi Duong, Sayantan Datta, and Derek Nowrouzezahrai. “Subspace Neural Physics: Fast Data-Driven Interactive Simulation”. In: *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2019.

[Hou*22] Trevor Houchens, Cheng-You Lu, Shivam Duggal, Rao Fu, and Srinath Sridhar. *NeuralODF: Learning Omnidirectional Distance Fields for 3D Shape Representation*. 2022. arXiv: `2206.05837 [cs.CV]`.

[Hua*20] Zeng Huang, Yuanlu Xu, Christoph Lassner, Hao Li, and Tony Tung. "ARCH: Animatable Reconstruction of Clothed Humans". In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 3093–3102.

[ITF04] G. Irving, J. Teran, and R. Fedkiw. "Invertible Finite Elements for Robust Simulation of Large Deformation". In: *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2004, pp. 131–140.

[ITF06] G. Irving, J. Teran, and R. Fedkiw. "Tetrahedral and hexahedral invertible finite elements". In: *Graphical Models* 68.2 (2006). Special Issue on SCA 2004, pp. 66–89.

[Jac*11] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. "Bounded Biharmonic Weights for Real-time Deformation". In: *ACM Trans. Graph.* 30.4 (July 2011), 78:1–78:8.

[Jac*14] Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. "Skinning: Real-time Shape Deformation". In: *ACM SIGGRAPH 2014 Courses*. 2014.

[Kad*16] Petr Kadleček, Alexandru-Eugen Ichim, Tiantian Liu, Jaroslav Křivánek, and Ladislav Kavan. "Reconstructing Personalized Anatomical Models for Physics-based Body Animation". In: *ACM Trans. Graph.* 35.6 (2016), 213:1–213:13.

[KK19] Petr Kadleček and Ladislav Kavan. "Building Accurate Physics-based Face Models from Data". In: *Proc. ACM Comput. Graph. Interact. Tech.* 2.2 (2019), 15:1–15:16.

[Käh*02] Kolja Kähler, Jörg Haber, Hitoshi Yamauchi, and Hans-Peter Seidel. "Head shop: Generating animated head models with anatomical structure". In: *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM. 2002, pp. 55–63.

[Kan*00] C. Kane, J. E. Marsden, M. Ortiz, and M. West. "Variational integrators and the Newmark algorithm for conservative and dissipative mechanical systems". In: *International Journal for Numerical Methods in Engineering* 49.10 (2000), pp. 1295–1325.

[Kar*21] Korrawe Karunratanakul, Adrian Spurr, Zicong Fan, Otmar Hilliges, and Siyu Tang. "A Skeleton-Driven Neural Occupancy Representation for Articulated Hands". In: *International Conference on 3D Vision (3DV)*. 2021.

[Kar*20] Korrawe Karunratanakul, Jinlong Yang, Yan Zhang, et al. "Grasping Field: Learning Implicit Representations for Human Grasps". In: *International Conference on 3D Vision*. 2020.

[Kav*11] Ladislav Kavan, Dan Gerszewski, Adam W. Bargteil, and Peter-Pike Sloan. "Physics-Inspired Upsampling for Cloth Simulation in Games". In: *ACM Trans. Graph.* 30.4 (2011).

[Kha*09] Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. "Numerical Coarsening of Inhomogeneous Elastic Materials". In: *ACM Trans. on Graphics* 28.3 (2009), 51:1–51:8.

[Kim*13] Doyub Kim, Woojong Koh, Rahul Narain, et al. “Near-Exhaustive Precomputation of Secondary Cloth Effects”. In: *ACM Trans. Graph.* 32.4 (2013).

[Kim*17] Meekyoung Kim, Gerard Pons-Moll, Sergi Pujades, et al. “Data-driven physics for human soft tissue animation”. In: *ACM Transactions on Graphics (TOG)* 36.4 (2017), p. 54.

[KDI19] Theodore Kim, Fernando De Goes, and Hayley Iben. “Anisotropic Elasticity for Inversion-safety and Element Rehabilitation”. In: *ACM Trans. Graph.* 38.4 (July 2019), 69:1–69:15.

[KJ11] Theodore Kim and Doug L. James. “Physics-Based Character Skinning Using Multi-Domain Subspace Deformations”. In: *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’11. Vancouver, British Columbia, Canada: Association for Computing Machinery, 2011, pp. 63–72.

[KB18] Martin Komaritzan and Mario Botsch. “Projective Skinning”. In: *Proc. ACM Comput. Graph. Interact. Tech.* 1.1 (2018), 12:1–12:19.

[KJP02] Paul G. Kry, Doug L. James, and Dinesh K. Pai. “EigenSkin: Real Time Large Deformation Character Skinning in Hardware”. In: *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’02. San Antonio, Texas: Association for Computing Machinery, 2002, pp. 153–159.

[KLM01] P. Krysl, S. Lall, and J. E. Marsden. “Dimensional model reduction in non-linear finite element dynamics of solids and structures”. In: *International Journal for Numerical Methods in Engineering* 51.4 (2001), pp. 479–504.

[KM04] Tsuneya Kurihara and Natsuki Miyata. “Modeling Deformable Human Hands from Medical Images”. In: *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2004, pp. 355–363.

[Lan*20] Lei Lan, Ran Luo, Marco Fratarcangeli, et al. “Medial Elastics: Efficient and Collision-Ready Deformation via Medial Axis Transform”. In: *ACM Trans. Graph.* 39.3 (2020).

[LC21] Kookjin Lee and Kevin T. Carlberg. “Deep Conservation: A Latent-Dynamics Model for Exact Satisfaction of Physical Conservation Laws”. In: *AAAI*. 2021.

[LST09] Sung-Hee Lee, Eftychios Sifakis, and Demetri Terzopoulos. “Comprehensive Biomechanical Modeling and Simulation of the Upper Body”. In: *ACM Trans. Graph.* 28.4 (2009), 99:1–99:17.

[LT06] Sung-Hee Lee and Demetri Terzopoulos. “Heads Up!: Biomechanical Modeling and Neuromuscular Control of the Neck”. In: *ACM Trans. Graph.* 25.3 (July 2006), pp. 1188–1198.

[LCF00] J. P. Lewis, Matt Cordner, and Nickson Fong. “Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation”. In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’00. USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 165–172.

[LLK19] Jing Li, Tiantian Liu, and Ladislav Kavan. “Fast Simulation of Deformable Characters with Articulated Skeletons in Projective Dynamics”. In: *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’19. Los Angeles, California: Association for Computing Machinery, 2019.

[Li*20] Minchen Li, Zachary Ferguson, Teseo Schneider, et al. “Incremental Potential Contact: Intersection-and Inversion-Free, Large-Deformation Dynamics”. In: *ACM Trans. Graph.* 39.4 (2020).

[LB14] Yijing Li and Jernej Barbič. “Stable Orthotropic Materials”. In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’14. Copenhagen, Denmark: Eurographics Association, 2014, pp. 41–46.

[Li*19] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B. Tenenbaum, and Antonio Torralba. “Learning Particle Dynamics for Manipulating Rigid Bodies, Deformable Objects, and Fluids”. In: *Proceedings of the International Conference on Learning Representations*. 2019.

[Liu*13] Libin Liu, KangKang Yin, Bin Wang, and Baining Guo. “Simulation and Control of Skeleton-driven Soft Body Characters”. In: *ACM Trans. Graph.* 32.6 (Nov. 2013), 215:1–215:8.

[Liu*19] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. “Point2Sequence: Learning the Shape Representation of 3D Point Clouds with an Attention-based Sequence to Sequence Network”. In: *AAAI Conference on Artificial Intelligence*. 2019.

[Lop*15] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. “SMPL: A Skinned Multi-person Linear Model”. In: *ACM Trans. Graph.* 34.6 (Oct. 2015), 248:1–248:16.

[Luo*20] R. Luo, T. Shao, H. Wang, et al. “NNWarp: Neural Network-Based Nonlinear Deformation”. In: *IEEE Transactions on Visualization and Computer Graphics* 26.4 (2020), pp. 1745–1759.

[Lyu*22] Qing Lyu, Menglei Chai, Xiang Chen, and Kun Zhou. “Real-Time Hair Simulation With Neural Interpolation”. In: *IEEE Transactions on Visualization and Computer Graphics* 28.4 (2022), pp. 1894–1905.

[Ma*20] Qianli Ma, Jinlong Yang, Anurag Ranjan, et al. “Learning to Dress 3D People in Generative Clothing”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2020.

[MLT89] N. Magnenat-Thalmann, R. Laperrière, and D. Thalmann. “Joint-Dependent Local Deformations for Hand Animation and Object Grasping”. In: *Proceedings on Graphics Interface ’88*. Edmonton, Alberta, Canada: Canadian Information Processing Society, 1989, pp. 26–33.

[Mal*15] Richard Malgat, Benjamin Gilles, David I. W. Levin, Matthieu Nesme, and François Faure. “Multifarious Hierarchies of Mechanical Models for Artist Assigned Levels-of-detail”. In: *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. 2015, pp. 27–36.

[Mar*11] Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. “Example-Based Elastic Materials”. In: *ACM Trans. Graph.* 30.4 (2011).

[McA*11] Aleka McAdams, Yongning Zhu, Andrew Selle, et al. “Efficient Elasticity for Character Skinning with Contact and Collisions”. In: *ACM Trans. Graph.* 30.4 (July 2011).

[Mes*19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. “Occupancy Networks: Learning 3D Reconstruction in Function Space”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2019.

[Mih*21] Marko Mihajlovic, Yan Zhang, Michael J Black, and Siyu Tang. “LEAP: Learning Articulated Occupancy of People”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2021.

[Mil*20] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, et al. “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis”. In: *Proc. of European Conference on Computer Vision (ECCV)*. 2020.

[Mül*04] M. Müller, R. Keiser, A. Nealen, et al. “Point Based Animation of Elastic, Plastic and Melting Objects”. In: *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2004, pp. 141–151.

[MG04] Matthias Müller and Markus Gross. “Interactive Virtual Materials”. In: *Proceedings of Graphics Interface 2004*. GI ’04. London, Ontario, Canada: Canadian Human-Computer Communications Society, 2004, pp. 239–246.

[Nak*18] Masaki Nakada, Tao Zhou, Honglin Chen, Tomer Weiss, and Demetri Terzopoulos. “Deep Learning of Biomimetic Sensorimotor Control for Biomechanical Human Animation”. In: *ACM Trans. Graph.* 37.4 (2018), 56:1–56:15.

[NSO12] Rahul Narain, Armin Samii, and James F. O’Brien. “Adaptive Anisotropic Remeshing for Cloth Simulation”. In: *ACM Trans. Graph.* 31.6 (Nov. 2012).

[Nea*06] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. “Physically Based Deformable Models in Computer Graphics”. In: *Computer Graphics Forum* 25.4 (2006), pp. 809–836.

[Neu*13] Thomas Neumann, Kiran Varanasi, Stephan Wenger, et al. “Sparse Localized Deformation Components”. In: *ACM Trans. Graph.* 32.6 (Nov. 2013).

[Ni*23] Ning Ni, Qingyu Xu, Zhehao Li, Xiao-Ming Fu, and Ligang Liu. “Numerical Coarsening with Neural Shape Functions”. In: *Computer Graphics Forum* n/a.n/a (2023).

[OHS03] James O’Brien, Kris Hauser, and Chen Shen. “Interactive Deformation Using Modal Analysis with Constraints”. In: *Graphics Interface* 3 (May 2003).

[Pai*18] Dinesh K. Pai, Austin Rothwell, Pearson Wyder-Hodge, et al. “The Human Touch: Measuring Contact with Real Human Soft Tissues”. In: *ACM Trans. Graph.* 37.4 (2018), 58:1–58:12.

[PBH15] Zherong Pan, Hujun Bao, and Jin Huang. “Subspace Dynamic Simulation Using Rotation-Strain Coordinates”. In: *ACM Trans. Graph.* 34.6 (Oct. 2015).

[Par*19] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2019.

[PH06] Sang Il Park and Jessica K Hodgins. “Capturing and animating skin deformation in human motion”. In: *ACM Transactions on Graphics (TOG)* 25.3 (2006), pp. 881–889.

[PH08] Sang Il Park and Jessica K. Hodgins. “Data-driven Modeling of Skin and Muscle Deformation”. In: *ACM Trans. Graph.* 27.3 (2008), 96:1–96:6.

[PLP20] Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. “TailorNet: Predicting Clothing in 3D as a Function of Human Pose, Shape and Garment Style”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. June 2020.

[PW89] Alex Pentland and John Williams. “Good Vibrations: Modal Dynamics for Graphics and Animation”. In: *Computer Graphics* 23.3 (1989), pp. 215–222.

[Pfa*21] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. “Learning Mesh-Based Simulation with Graph Networks”. In: *International Conference on Learning Representations*. 2021.

[Pon*17] Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J. Black. “ClothCap: Seamless 4D Clothing Capture and Retargeting”. In: *ACM Transactions on Graphics (Proc. of SIGGRAPH)* 36.4 (2017), 73:1–73:15.

[Pon*15] Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J. Black. “Dyna: A Model of Dynamic Human Shape in Motion”. In: *ACM Trans. Graph.* 34.4 (July 2015), 120:1–120:14.

[PO18] Adrien Poulenard and Maks Ovsjanikov. “Multi-Directional Geodesic Neural Networks via Equivariant Convolution”. In: *ACM Trans. Graph.* 37.6 (2018).

[Qi*17a] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 652–660.

[Qi*17b] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[Rob*17] Nadia Robertini, Dan Casas, Edilson De Aguiar, and Christian Theobalt. “Multi-view Performance Capture of Surface Details”. In: *International Journal of Computer Vision (IJCV)* 124.1 (Aug. 2017), pp. 96–113.

[RTB17] Javier Romero, Dimitrios Tzionas, and Michael J. Black. “Embodied Hands: Modeling and Capturing Hands and Bodies Together”. In: *ACM Trans. Graph.* 36.6 (2017).

[Sai*19] Shunsuke Saito, Zeng Huang, Ryota Natsume, et al. “PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization”. In: *The IEEE International Conference on Computer Vision (ICCV)*. 2019.

[Sai*20] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. “PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2020.

[Sai*21] Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J. Black. “SCANimate: Weakly Supervised Learning of Skinned Clothed Avatar Networks”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. June 2021.

[San*18] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, et al. “Graph Networks as Learnable Physics Engines for Inference and Control”. In: *ICML*. 2018, pp. 4467–4476.

[San*20] Igor Santesteban, Elena Garces, Miguel A. Otaduy, and Dan Casas. “SoftSMPL: Data-driven Modeling of Nonlinear Soft-tissue Dynamics for Parametric Humans”. In: *Computer Graphics Forum* 39.2 (2020), pp. 65–75.

[San*22] Igor Santesteban, Miguel Otaduy, Nils Thuerey, and Dan Casas. “ULNeF: Untangled Layered Neural Fields for Mix-and-Match Virtual Try-On”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo, S. Mohamed, A. Agarwal, et al. Vol. 35. Curran Associates, Inc., 2022, pp. 12110–12125.

[SOC19] Igor Santesteban, Miguel A. Otaduy, and Dan Casas. “Learning-Based Animation of Clothing for Virtual Try-On”. In: *Computer Graphics Forum* 38.2 (2019), pp. 355–366.

[SOC22] Igor Santesteban, Miguel A. Otaduy, and Dan Casas. “SNUG: Self-Supervised Neural Dynamic Garments”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 8140–8150.

[San*21] Igor Santesteban, Nils Thuerey, Miguel A Otaduy, and Dan Casas. “Self-Supervised Collision Handling via Generative 3D Garment Models for Virtual Try-On”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

[Sch*15] Christian Schumacher, Bernd Bickel, Jan Rys, et al. “Microstructures to Control Elasticity in 3D Printing”. In: *ACM Trans. Graph.* 34.4 (2015), 136:1–136:13.

[Sha*23] Nicholas Sharp, Cristian Romero, Alec Jacobson, et al. *Data-Free Learning of Reduced-Order Kinematics*. 2023. arXiv: `2305.03846 [cs.GR]`.

[She*21] Siyuan Shen, Yin Yang, Tianjia Shao, et al. “High-Order Differentiable Autoencoder for Nonlinear Model Reduction”. In: *ACM Trans. Graph.* 40.4 (2021).

[Si15] Hang Si. “TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator”. In: *ACM Trans. Math. Softw.* 41.2 (Feb. 2015), 11:1–11:36.

[SB12] Eftychios Sifakis and Jernej Barbic. “FEM Simulation of 3D Deformable Solids: A Practitioner’s Guide to Theory, Discretization and Model Reduction”. In: *ACM SIGGRAPH 2012 Courses*. 2012, 20:1–20:50.

[SNF05] Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. “Automatic Determination of Facial Muscle Activations from Sparse Motion Capture Marker Data”. In: *ACM Transactions on Graphics (Proc. SIGGRAPH)* 24.3 (2005), pp. 417–425.

[Sim*22] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, et al. “Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 6394–6400.

[Sit*20] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. “Implicit Neural Representations with Periodic Activation Functions”. In: *Proc. NeurIPS*. 2020.

[SGK18] Breannan Smith, Fernando De Goes, and Theodore Kim. “Stable Neo-Hookean Flesh Simulation”. In: *ACM Trans. Graph.* 37.2 (Mar. 2018), 12:1–12:15.

[SSR20] Steven L. Song, Weiqi Shi, and Michael Reed. “Accurate Face Rig Approximation with Deep Differential Subspace Reconstruction”. In: *ACM Trans. Graph.* 39.4 (2020).

[Sor05] Olga Sorkine. “Laplacian Mesh Processing”. In: *Eurographics 2005 - State of the Art Reports*. Ed. by Yiorgos Chrysanthou and Marcus Magnor. The Eurographics Association, 2005.

[Str71] A. H. Stroud. *Approximate calculation of multiple integrals*. Prentice-Hall, 1971.

[Tan*22a] Qingyang Tan, Zherong Pan, Breannan Smith, Takaaki Shiratori, and Dinesh Manocha. “N-Penetrate: Active Learning of Neural Collision Handler for Complex 3D Mesh Deformations”. In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri, Stefanie Jegelka, Le Song, et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, July 2022, pp. 21037–21049.

[Tan*22b] Qingyang Tan, Yi Zhou, Tuanfeng Wang, et al. “A Repulsive Force Unit for Garment Collision Handling in Neural Networks”. In: *Computer Vision – ECCV 2022*. Ed. by Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner. Cham: Springer Nature Switzerland, 2022, pp. 451–467.

[TK94] Camillo J. Taylor and David J. Kriegman. *Minimization on the Lie Group SO(3) and Related Manifolds*. Tech. rep. Yale University, 1994.

[Ten*15] Yun Teng, Mark Meyer, Tony DeRose, and Theodore Kim. “Subspace Condensation: Full Space Adaptivity for Subspace Deformations”. In: *ACM Trans. Graph.* 34.4 (July 2015).

[TOK14] Yun Teng, Miguel A. Otaduy, and Theodore Kim. “Simulating Articulated Subspace Self-Contact”. In: *ACM Trans. Graph.* 33.4 (July 2014).

[Ter*05] Joseph Teran, Eftychios Sifakis, Silvia S. Blemker, et al. “Creating and Simulating Skeletal Muscle from the Visible Human Data Set”. In: *IEEE Transactions on Visualization and Computer Graphics* 11.3 (2005), pp. 317–328.

[TW88] D. Terzopoulos and A. Witkin. “Physically based models with rigid and deformable components”. In: *IEEE Computer Graphics and Applications* 8.6 (1988), pp. 41–51.

[Ter*87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. “Elastically Deformable Models”. In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’87. New York, NY, USA: Association for Computing Machinery, 1987, pp. 205–214.

[Tiw*21] Garvita Tiwari, Nikolaos Sarafianos, Tony Tung, and Gerard Pons-Moll. “Neural-GIF: Neural generalized implicit functions for animating people in clothing”. In: *Proc. of Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 11708–11718.

[Tor*16] Rosell Torres, Alejandro Rodrıguez, José M. Espadero, and Miguel A. Otaduy. "High-resolution Interaction with Corotational Coarsening Models". In: *ACM Trans. Graph.* 35.6 (2016), 211:1–211:11.

[Tyc*13] Christoph von Tycowicz, Christian Schulz, Hans-Peter Seidel, and Klaus Hildebrandt. "An Efficient Construction of Reduced Deformable Objects". In: *ACM Trans. Graph.* 32.6 (2013).

[VCO20] Mickeal Verschoor, Dan Casas, and Miguel A. Otaduy. "Tactile Rendering Based on Skin Stress Optimization". In: *ACM Trans. Graph.* 39.4 (2020).

[Wan*10] Huamin Wang, Florian Hecht, Ravi Ramamoorthi, and James F. O'Brien. "Example-Based Wrinkle Synthesis for Clothing Animation". In: *ACM Trans. Graph.* 29.4 (July 2010).

[WPP07] Robert Y. Wang, Kari Pulli, and Jovan Popović. "Real-Time Enveloping with Rotational Regression". In: *ACM SIGGRAPH 2007 Papers*. SIGGRAPH '07. San Diego, California: Association for Computing Machinery, 2007.

[Wan*15] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. "Linear Subspace Design for Real-Time Shape Deformation". In: *ACM Trans. Graph.* 34.4 (2015).

[Wan*19] Yue Wang, Yongbin Sun, Ziwei Liu, et al. "Dynamic Graph CNN for Learning on Point Clouds". In: *ACM Transactions on Graphics (TOG)* (2019).

[WEH20] Ruben Wiersma, Elmar Eisemann, and Klaus Hildebrandt. "CNNs on Surfaces Using Rotation-Equivariant Features". In: *ACM Trans. Graph.* 39.4 (2020).

[WBT19] S. Wiewel, M. Becher, and N. Thuerey. "Latent Space Physics: Towards Learning the Temporal Evolution of Fluid Flow". In: *Computer Graphics Forum* 38.2 (2019), pp. 71–82.

[Wu*16] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling". In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016, pp. 82–90.

[WMW15] Xiaofeng Wu, Rajaditya Mukherjee, and Huamin Wang. "A Unified Approach for Subspace Simulation of Deformable Bodies in Multiple Domains". In: *ACM Trans. Graph.* 34.6 (2015).

[Xie*21] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, et al. *Neural Fields in Visual Computing and Beyond*. 2021. eprint: `arXiv:2111.11426`.

[XB16] Hongyi Xu and Jernej Barbič. "Pose-Space Subspace Dynamics". In: *ACM Trans. Graph.* 35.4 (July 2016).

[Xu*15] Hongyi Xu, Funshing Sin, Yufeng Zhu, and Jernej Barbič. "Nonlinear Material Design Using Principal Stretches". In: *ACM Trans. Graph.* 34.4 (2015).

[Yan*22] Lingchen Yang, Byungsoo Kim, Gaspard Zoss, et al. "Implicit Neural Representation for Physics-Driven Actuated Soft Bodies". In: *ACM Trans. Graph.* 41.4 (2022).

[Zha*21] Meng Zhang, Tuanfeng Y. Wang, Duygu Ceylan, and Niloy J. Mitra. "Dynamic Neural Garments". In: *ACM Trans. Graph.* 40.6 (2021).

[Zhe*21] Mianlun Zheng, Yi Zhou, Duygu Ceylan, and Jernej Barbic. "A Deep Emulator for Secondary Motion of 3D Characters". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 5932–5940.

[ZJ16] Qingnan Zhou and Alec Jacobson. *Thingi10K: A Dataset of 10,000 3D-Printing Models*. 2016. arXiv: `1605.04797 [cs.GR]`.

[Zor*04] Victor Brian Zordan, Bhrigu Celly, Bill Chiu, and Paul C DiLorenzo. "Breathe easy: model and control of simulated respiration for animation". In: *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association. 2004, pp. 29–37.

[ZBO13] J. S. Zurdo, J. P. Brito, and M. A. Otaduy. "Animating Wrinkles by Example on Non-Skinned Cloth". In: *IEEE Transactions on Visualization & Computer Graphics* 19.01 (2013), pp. 149–158.

A
Resumen

Muchos de los objetos con los que interactuamos a diario son deformables. De hecho, los materiales deformables son un elemento fundamental en la fabricación de mobiliario, accesorios y herramientas, entre otras muchas cosas. Una de las principales razones de su uso tan extendido es su característico comportamiento mecánico. Estos materiales pueden cambiar de forma, se adaptan mejor a las interacciones, y tienden a ser más cómodos y ergonómicos. No sólo en la fabricación de objetos, los materiales deformables también son muy comunes en los seres vivos, como los humanos. A excepción de los huesos, nuestro cuerpo está formado principalmente por tejidos blandos, como músculos y grasa. De hecho, podríamos decir que nuestro cuerpo es uno de los objetos deformables más importantes en nuestras interacciones cotidianas.

Por lo tanto, modelar y comprender adecuadamente el comportamiento de los materiales deformables es esencial para el progreso de distintas aplicaciones en computación gráfica, ingeniería mecánica y biomecánica. Además de ayudar a comprender nuestro entorno, el modelado de objetos deformables también es importante en entornos virtuales como juegos, simuladores de entrenamiento o realidad virtual. En este contexto, los mundos virtuales realistas exigen interacciones deformables complejas, con avatares, objetos y ropa, todo ello usando modelos de deformación adecuados para aportar una experiencia de inmersión completa.

En el campo de la computación gráfica se han propuesto diversos métodos computacionales para el modelado de objetos deformables, con distinto alcance y dominio de aplicación, abarcando desde métodos aproximados e interactivos, a métodos más precisos. Por desgracia, aún no existe un modelo de deformación general que se adapte de forma práctica a todos los casos de uso. Motivados por esta limitación, en esta tesis utilizamos métodos basados en datos para acortar la brecha existente entre los modelos de simulación precisos y los interactivos. El objetivo de esta tesis es desarrollar métodos de simulación útiles para objetos deformables que sean rápidos y precisos, con especial atención a modelar eficientemente comportamientos dinámicos, interacciones externas y deformaciones producidas por contacto. Para ello, investigamos la combinación de modelos físicos y técnicas basadas en datos.

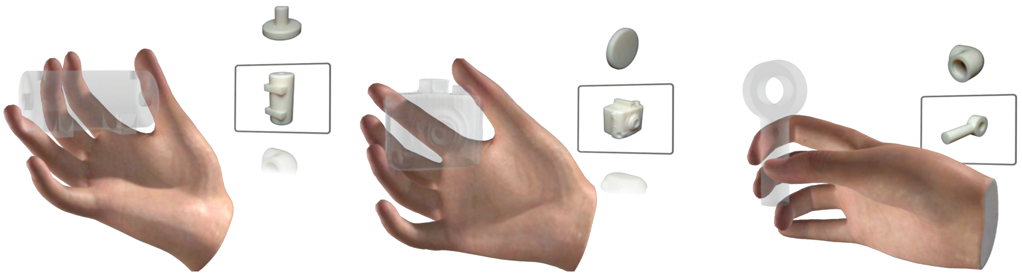


Figure A.1: Algunas deformaciones producidas por nuestros métodos en tiempo real. Las deformaciones de contacto aprendidas en esta mano generalizan ante interacciones con distintos objetos.

A.1 Antecedentes

Desde el trabajo pionero de Terzopoulos y Witkin [TW88], la comunidad de computación gráfica ha estado cada vez más interesada en la animación de objetos deformables basada en físicas. En las últimas décadas, se han presentado diferentes métodos para mejorar la eficiencia y expresividad de las deformaciones producidas, con muchos artículos y cursos que discuten la evolución y el desarrollo actual de la simulación de los objetos deformables, por ejemplo, [Nea*06; SB12; BS19].

Se han propuesto diferentes técnicas de rigging para la deformación controlable de objetos blandos, como es el lineal blend skinning o los métodos basados en puntos de control [Jac*14], mejorando las deformaciones con correcciones dependientes de la pose [LCF00]. Además, la simulación de cuerpos articulados deformables ha motivado muchos trabajos que enriquecen las técnicas de rigging con deformaciones dinámicas [Cap*02; Cap*05] acoplando bidireccionalmente el tejido blando y el cuerpo articulado [Liu*13]. Dejando a un lado los rigs articulados, otros métodos de animación utilizan definiciones más diversas de la pose global, agregando la deformación local de la superficie para modelar caras [Bic*08] o tela [Kav*11; ZBO13].

Se ha dedicado un esfuerzo importante a mejorar la eficiencia de las simulaciones físicas para objetos deformables [Gal*07; McA*11; LLK19]. De forma relevante, se han utilizado modelos reducidos de subespacio para aproximar las ecuaciones del movimiento, ignorando las deformaciones de alta frecuencia [SB12], utilizando análisis modal [PW89] o usando análisis de componentes principales a partir de ejemplos de deformación [KLM01]. Un problema común de las simulaciones reducidas es que las deformaciones de contacto no se resuelven con gran precisión, especialmente en el caso de las interacciones con objetos detallados. Algunos trabajos han abordado esta limitación, enriqueciendo el subespacio con bases de deformación local [HZ13] o simulando detalles en una región local [Ten*15].

Los subespacios tradicionales suelen tener soporte global, lo que significa que cada grado de libertad del deformable depende cinemáticamente de todos los grados de libertad del subespacio. Esto no es necesariamente realista, y puede provocar que nuevas interacciones no consideradas originalmente en el diseño produzcan artefactos poco plausibles. Algunos trabajos han aliviado este problema utilizando técnicas de descomposición de dominio [BZ11; KJ11; WMW15], mientras que otros abordan directamente el diseño de subespacios locales utilizando puntos de control [Wan*15] o transformaciones interpolados [Gil*11; Fau*11; BEH18]. También se han empleado subespacios en combinación con modelos de rigging. Los primeros trabajos capturan la deformación con una base de autovectores [KJP02], y trabajos posteriores mejoran este enfoque para la simulación dinámica de personajes [Gal*09] o telas [Hah*14].

En los últimos años, se han aplicado métodos basados en datos para aprender diferentes tipos de modelos de deformación. Entre otras aplicaciones, se han utilizado para sintetizar detalles de deformación de alta resolución condicionados por representaciones de baja dimensión [Lop*15; Bai*18], para inferir automáticamente subespacios no lineales compactos para deformaciones dinámicas [Ful*19], o para aprender las interacciones dinámicas entre un objeto deformable y algún(os) rigido(s) de colisión [Hol*19]. Otros trabajos han abordado la personalización de los modelos basados en físicas mediante la estimación de propiedades geométricas y de material a partir de datos observados [Kad*16; KK19] u optimizando los materiales a partir de mediciones locales [Pai*18].

Los métodos basados en datos y aprendizaje automático se caracterizan por usar novedosas representaciones en sus modelos. Entre estas representaciones, las más utilizadas son las denominadas redes neuronales artificiales, y los modelos que las utilizan se conocen como *representaciones neuronales* [Xie*21]. A la hora de modelar magnitudes de naturaleza espacial, estas representaciones neuronales han demostrado tener características interesantes, por sus propiedades de aproximación universales, su flexibilidad de parametrización, y la facilidad de realizar evaluaciones en puntos espaciales arbitrarios sin requerir de discretizaciones.

Estas representaciones se han extendido rápidamente para abordar muchos problemas diferentes en visión por computador y computación gráfica [Che*21]. Desde sintetizar nuevas vistas de escenas complejas [Mil*20], hasta reconstruir formas tridimensionales a partir de imágenes [Sai*19; Sai*20]. Los trabajos más recientes utilizan modelos neuronales implícitos para codificar formas tridimensionales, aprendiendo a aproximar las superficies mediante función de ocupación en puntos espaciales arbitrarios [Mes*19; CZ19], o aprendiendo distancias a la superficie [Par*19; CZ19; AL20].

A.2 Objetivos

Como ya se ha indicado, nuestro principal objetivo es diseñar modelos computacionales para la simulación eficiente de objetos y materiales deformables. A continuación enumeramos los objetivos más importantes que debe cumplir un modelo de deformación para ser útil en aplicaciones generales de computación gráfica:

- **Expresividad.** Nuestro objetivo son los modelos de deformación que generen resultados realistas y precisos desde el punto de vista perceptual. Una respuesta mecánica adecuada y un comportamiento dinámico también son propiedades relevantes a tener en cuenta. Es importante señalar que en las aplicaciones de computación gráfica, la evaluación final de la calidad suele realizarse mediante nuestros sentidos, lo que permite aproximaciones en los modelos que podrían no ser aceptables en otros campos de la ingeniería. Por lo tanto, buscamos modelos de deformación cuyos resultados sean lo suficientemente correctos como para parecer realistas.

- **Eficiencia.** Los modelos eficientes pueden utilizarse en distintos entornos, facilitan la exploración y permiten tiempos de simulación interactivos. Particularmente en computación gráfica, necesitamos modelos de deformación eficientes en términos de recursos computacionales. La razón es que en este contexto muchas de las aplicaciones requieren de una ejecución interactiva, ya sea para permitir experiencias inmersivas o para proporcionar previsualizaciones de los resultados finales. Además, estos modelos suelen ejecutarse en dispositivos con capacidades limitadas, como ordenadores personales, dispositivos móviles o cascos de realidad virtual, lo que motiva aún más la necesidad de eficiencia.

- **Generalización.** Para que un modelo de deformación sea útil, debe ser estable, permitir interacciones precisas y soportar deformaciones generales por contacto. Además, para ser aplicables en entornos poco controlados, los modelos de deformación deben responder con robustez a una amplia variedad de situaciones. Esto es especialmente relevante en las interacciones por contacto con otros objetos, en las que las deformaciones generadas pueden ser muy diversas. Por lo tanto, los modelos tienen que generalizar adecuadamente a esta diversidad.

Los métodos actuales para simular objetos deformables no cumplen todos los objetivos mencionados. Los modelos precisos basados en leyes físicas producen resultados detallados [SB12; Kim*17] y generalizan adecuadamente a deformaciones de contacto complejas [Li*20], sin embargo, a costa de perder eficiencia. Por otro lado, los modelos diseñados para aplicaciones interactivas son eficientes, pero muestran una expresividad y generalización limitadas a la hora de capturar respuestas realistas de los materiales [KB18; LLK19] y deformaciones por contacto de alta frecuencia espacial [PW89; BJ05; Hah*12; Hah*13; BEH18].

En esta situación, nosotros proponemos combinar modelos reducidos de simulación física con métodos basados en datos, para construir así modelos con todos los objetivos deseados. Apoyarnos en simulaciones físicas nos proporciona generalización, interacciones intuitivas basadas en fuerzas y dinámicas expresivas sin amortiguamiento. Además, utilizando técnicas novedosas basadas en datos, podemos mejorar la capacidad de generalización de los modelos aproximados de deformación, preservando el realismo de los modelos de alta resolución.

A.3 Metodología

Los modelos deformables útiles deben ser expresivos, eficientes y deben generalizar a diversas interacciones. A continuación analizamos la metodología seguida en esta tesis para conseguir métodos de simulación prácticos con estas propiedades:

- **Revisar la bibliografía.** Comenzamos nuestro proceso de investigación con la intuición de que combinar métodos de simulación física con técnicas basadas en datos podría ser una buena estrategia para mejorar el estado del arte en lo que respecta a los objetivos planteados en esta tesis. Para poner a prueba esta intuición, se realizó una revisión bibliográfica, recopilando distintos trabajos con ideas similares, y encontrando en el proceso una línea de trabajo interesante aplicada a la simulación de avatares deformables. Durante todo el proceso de esta tesis, se ha realizado una revisión periódica de las publicaciones más recientes, tanto para garantizar la novedad de nuestras contribuciones futuras, como para estar al día de las novedades más relevantes que pudieran aportar algo en nuestra línea de trabajo.

- **Capturar la complejidad de los objetos reales.** El cuerpo humano es un sistema especialmente complejo, con múltiples huesos, capas de tejidos blandos y mecanismos articulados acoplados. Por lo tanto, modelizar eficientemente el comportamiento de estos sistemas basándose únicamente en modelos físicas es casi imposible. Para resolver este reto, comenzamos nuestra investigación diseñando técnicas basadas en datos para reducir la complejidad de los modelos físicos, sustituyendo esta complejidad por datos. Entre otras cosas, optimizamos los parámetros de los materiales deformables y su grosor a partir de datos reales, diseñamos modelos de materiales expresivos, calculamos la deformación elástica en la pose de reposo y combinamos dinámicas simuladas con deformaciones cuasiestáticas capturadas del mundo real.

- **Mejorar el rendimiento de las simulaciones físicas.** Las simulaciones físicas siguen siendo la preferencia a la hora de calcular deformaciones de uso general, con un comportamiento preciso y dinámico. Sin embargo, para obtener resultados realistas se requieren simulaciones de alta resolución, y estas suelen ser lentas. Además, estos métodos pueden sufrir inestabilidades si no se toman medidas adicionales. Para evitar estas limitaciones, en esta tesis utilizamos modelos reducidos. Combinamos las simulaciones de subespacios con el alto nivel de detalle de las deformaciones basados en aprendizaje automático, presentando modelos eficientes, dinámicos y que responden de forma natural a las interacciones externas. Nuestros modelos utilizan subespacios basados en puntos de control, sobre los que añadimos eficientes correcciones no lineales generadas mediante datos, desacoplando las deformaciones internas y las externas generadas por contactos.

- **Aumentar la generalización de los métodos basados en datos.** Los modelos basados en datos tienden a no generalizar correctamente si no se proporcionan suficientes ejemplos de deformación. Es especialmente difícil soportar interacciones generales y deformaciones de contacto arbitrarias en modelos basados en datos, debido al gran número de tipos de interacción. Sin embargo, obtener conjuntos de datos exhaustivos puede ser muy caro o incluso imposible, y los modelos resultantes pueden ser ineficientes y de gran tamaño. Para abordar este problema, presentamos métodos para reducir la cantidad de datos necesarios, con resultados de buena calidad y usando pocos recursos computacionales. Nuestro enfoque se basa en generar ejemplos de deformación con una cobertura eficiente del espacio de interacciones, y en codificar los modelos de deformación con una representación adecuada para el problema en cuestión, por ejemplo, modelando las deformaciones de contacto desde el punto de vista del objeto que las genera, o utilizando descriptores locales para los contactos.

A.4 Resultados

Aquí recogemos las contribuciones de esta tesis, junto con los resultados clave de cada una de ellas, acompañado de las publicaciones resultantes:

- **Avatares con deformaciones de tejido blando dinámicas y precisas.** Presentamos un modelo de avatar animado con deformaciones híbridas de tejido blando, construido mediante la combinación de un modelo de cuerpo humano basado en datos y una simulación mecánica por elementos finitos. Como contribución clave, definimos la mecánica de la deformación en la pose de referencia del cuerpo, conservando en la medida de lo posible las deformaciones precisas del modelo basado en datos. También introducimos un material anisótropo no lineal para una dinámica precisa, optimizando automáticamente el grosor de la piel y las propiedades del material a partir de capturas de humanos reales. Por último, para lograr tiempos interactivos, aumentamos el modelo de avatar con una cuidadosa elección de subespacio para simulaciones reducidas. Estas contribuciones se analizan en el capítulo 3 y han dado lugar a las siguientes publicaciones:

- Cristian Romero, Miguel A. Otaduy, Dan Casas and Jesús Pérez. "Modeling and Estimation of Nonlinear Skin Mechanics for Animated Avatars". *Computer Graphics Forum (Proc. of Eurographics)* (2020)

- Javier Tapia, Cristian Romero, Jesús Pérez and Miguel A. Otaduy. "Parametric Skeletons with Reduced Soft-Tissue Deformations". *Computer Graphics Forum* (2021)

- **Modelo rápido de simulación reducida con correcciones de deformación aprendidas.** Hemos diseñado un nuevo método de subespacio para la simulación de deformaciones dinámicas, aumentando los subespacios lineales basados en puntos de control con correcciones no lineales aprendidas. De esta forma combinamos la dinámica de los rápidos métodos de subespacio con las deformaciones altamente detalladas de los métodos basados en aprendizaje automático. Aplicamos las correcciones no lineales en el estado no deformado, y para mejorar la generalización desacoplamos las correcciones internas y las externas generadas por contactos. Además, diseñamos un proceso de entrenamiento para generar ejemplos que cubran eficientemente el espacio de interacción. Estas contribuciones se discuten en el capítulo 4 y se han recogido en la siguiente publicación:

  - Cristian Romero, Dan Casas, Jesús Pérez, and Miguel A. Otaduy. "Learning Contact Corrections for Handle-Based Subspace Dynamics". *ACM Transactions on Graphics (SIGGRAPH)* (2021)

- **Representación eficiente para el aprendizaje de deformaciones por contacto.** Presentamos un método eficiente para el aprendizaje automático de las deformaciones no lineales producidas por contacto. Utilizando una nueva representación adaptada al problema, obtenemos simulaciones dinámicas detalladas en tiempo real. Para ello, nos apartamos de estrategias anteriores, modelando las deformaciones de contacto desde el punto de vista del objeto que genera la deformación. Además, aprendemos un campo vectorial continuo para las deformaciones, en lugar de emplear una aproximación discreta. Estas contribuciones se discuten en el capítulo 5 y han dado lugar a la siguiente publicación:

- Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. "Contact-Centric Deformation Learning". *ACM Transactions on Graphics (SIGGRAPH)* (2022)

- **Aprendizaje generalizado para deformaciones de contacto arbitrarias.** Formulamos un método basado en aprendizaje automático para la simulación de deformaciones producidas por contacto, generalizando el modelo de deformación introducido previamente a objetos de colisión arbitrarios. Gracias a un novedoso descriptor que caracteriza la geometría local en las regiones de interés, diseñamos un nuevo modelo neuronal que admite formas generales como objetos de colisión. Para validar nuestro enfoque, demostramos nuestro método con simulaciones dinámicas interactivas y deformaciones por contacto detalladas. Estas contribuciones se discuten en el capítulo 6 y han culminado en la siguiente publicación:

  - Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. "Learning Contact Deformations with General Collider Descriptors". *SIGGRAPH ASIA Conference Proceedings* (2023)

A.5 Conclusiones

En esta tesis hemos presentado importantes contribuciones para la simulación de objetos deformables, mediante la introducción de nuevos modelos de simulación para avatares con tejido blando, modelos reducidos y contacto. En el proceso, hemos resuelto distintos retos interesantes para conseguir métodos de deformación expresivos, eficientes y generales.

Siendo de vital importancia para muchas aplicaciones, hemos dedicado especial atención a modelar eficientemente la deformación de los tejidos blandos del cuerpo humano, validando exitosamente nuestros métodos. En el Capítulo 3 hemos presentado un modelo de avatar animado con deformaciones dinámicas de cuerpo completo. En los capítulos 4 ,5 y 6 aplicamos nuestros novedosos métodos de deformación por contacto para demostrar su utilidad en partes del cuerpo específicamente relevantes, como las manos (Fig. 5.1 y Fig. 6.1) o los dedos (Fig. 4.12 y Fig. 6.12).

No obstante, aún hay margen de mejora a la hora de capturar la deformación de objetos complejos. Por un lado, para aumentar aún más la expresividad y la capacidad de describir comportamientos del mundo real, es necesario encontrar modelos constitutivos de material con parametrizaciones más flexibles. Por otro lado, para poder estimar un gran número de parámetros a partir de datos reales, seguimos necesitando soportar métodos más eficientes para el cálculo de derivadas en nuestros modelos.

Otro reto importante que hemos abordado es la dificultad para obtener tiempos interactivos en las simulaciones físicas de alta calidad. Para superar esta limitación, hemos diseñado modelos reducidos de simulación rápida con correcciones de deformaciones aprendidas. Además, utilizamos modelos basados en datos y redes neuronales para sustituir complejas relaciones de deformación, construidas a partir de datos de ejemplos capturados [Lop*15] o simulados. Esto evita la necesidad de un modelo exhaustivo basado en físicas y nos brinda mucho más margen para realizar aproximaciones eficientes.

Aunque nuestros modelos reducidos sean eficientes, siguen teniendo algunas limitaciones. Utilizamos subespacios basados en puntos de control, los cuales resultan especialmente útiles gracias a su naturaleza geométrica. Sin embargo, el proceso de seleccionar la ubicación de los puntos de control requiere de cierto trabajo manual. En este contexto, el diseño de un enfoque automático para la colocación de los puntos supondría una mejora en la utilidad. Por otro lado, en esta tesis nos hemos centrado principalmente en reducir el número de grados de libertad simulados, mejorando la cinemática de los modelos reducidos de deformación. Sin embargo, el cálculo de fuerzas y su aproximación también se pueden mejorar con técnicas basadas en datos, especialmente para el contacto, donde no estamos utilizando ninguna aproximación.

Las mejoras que presentamos se basan en gran medida en nuevas técnicas de deformación basadas en datos. Hemos dedicado gran parte de esta tesis a mejorar sus capacidades de generalización para lograr modelos más prácticos y que requieran de menos datos. Esto ha sido posible gracias a nuestras representaciones eficientes y descriptores generalizados para el aprendizaje de deformaciones generales por contacto. Debemos destacar también la importancia de nuestras simulaciones combinadas utilizando aprendizaje automático y subespacios de reducción, contribuyendo significativamente a la generalización. Al igual que las simulaciones reducidas, nuestros modelos pueden generar comportamientos dinámicos temporales, además de poder acoplarse fácilmente a otros modelos físicos para una interacción natural. Por ejemplo, mostramos ejemplos de acoplamiento bidireccional

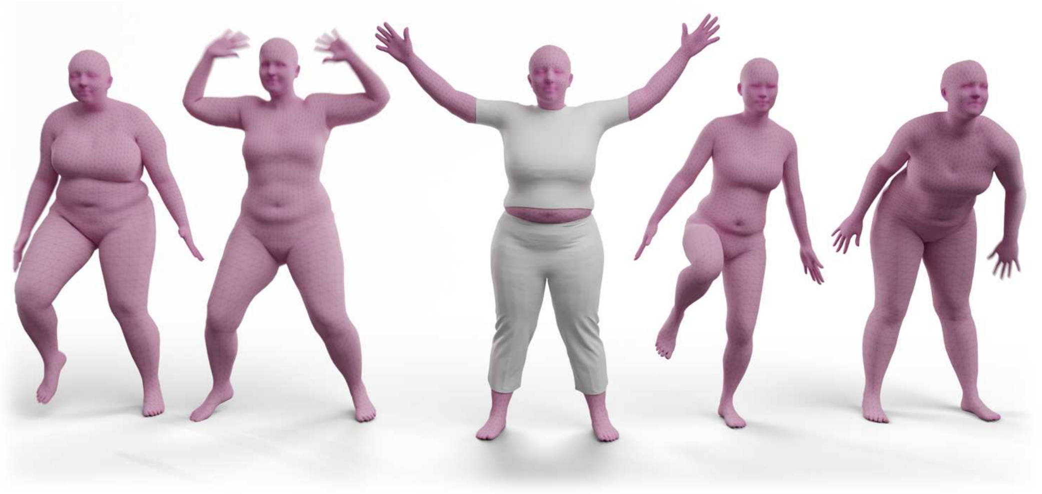


Figure A.2: Nuestros modelos permiten simular sistemas complejos como estos avatares deformables, con efectos dinámicos, interacciones realistas y acoplamientos con ropa.

con tela simulada (Fig. 3.6), acoplamiento bidireccional con objetos rígidos (Fig. 6.10), o interacciones basadas en fuerzas externas (Fig. 4.10).

Nuestras representaciones reducen drásticamente la cantidad de datos necesarios para el aprendizaje automático. Aún así, en los capítulos 4, 5, y 6, la mayor parte de los recursos computacionales son utilizados para la generación de ejemplos de deformación por contacto. Con la motivación de reducir los tiempos de este preprocesado, puede merecer la pena explorar un enfoque autosupervisado para el aprendizaje de las deformaciones, siguiendo los trabajos publicados recientemente en esta línea [SOC22; BME22]. Por último, como opción interesante, también sería posible entrenar nuestros modelos con datos reales de deformación de contacto, en lugar de utilizar ejemplos simulados como hacemos actualmente.

En conclusión, en esta tesis hemos introducido métodos novedosos para la simulación de objetos deformables, combinando con éxito técnicas basadas en datos y simulaciones físicas. Como se ha comentado en este capítulo, aún quedan muchos aspectos por explorar, pero espero que el trabajo presentado ayude a acortar la distancia entre los modelos de alta precisión y los interactivos, allanando el camino hacia métodos generales que nos permitan simular de forma eficiente y precisa el comportamiento de nuestros propios cuerpos y de los objetos deformables que nos rodean.