Learning Contact Corrections for Handle-Based Subspace Dynamics

CRISTIAN ROMERO, DAN CASAS, JESÚS PÉREZ, and MIGUEL OTADUY, Universidad Rey Juan Carlos, Spain

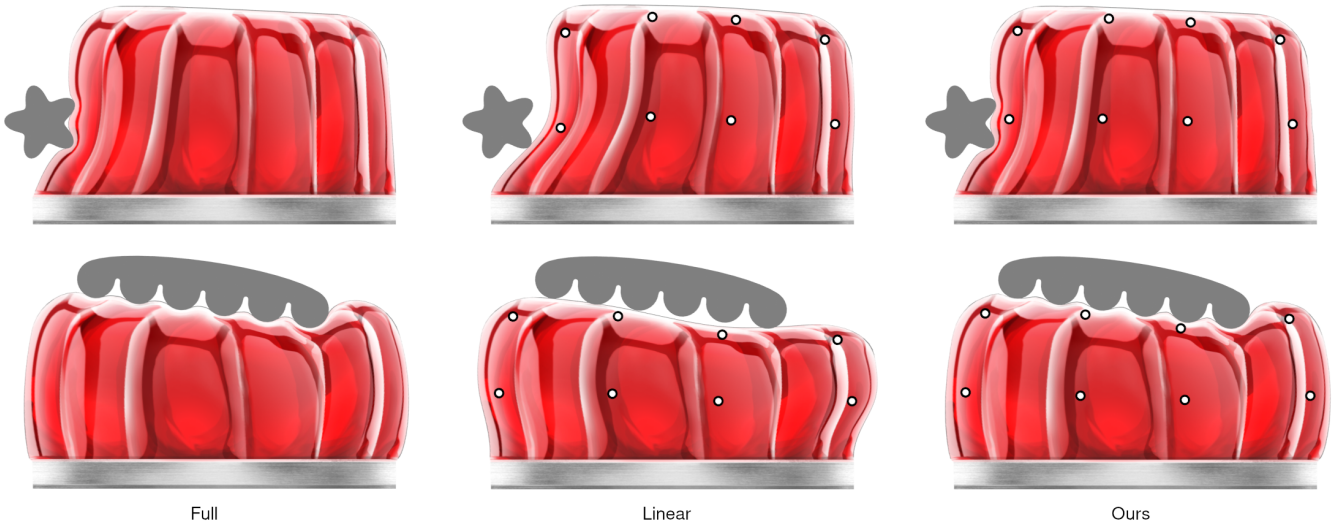

Fig. 1. The left images show a dynamic simulation of an FEM Neo-Hookean jelly with 12,469 triangles. The deformation is rich but slow (20 fps). The central images show the same scene using a linear subspace model built with just 8 point handles. The simulation is fast (420 fps), but it misses all the detail and suffers distortion under moderate forces. The right 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.

This paper 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.

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

Additional Key Words and Phrases: Dynamics, subspace, learning

Authors' address: Cristian Romero; Dan Casas; Jesús Pérez; Miguel Otaduy, first.last@urjc.es, Universidad Rey Juan Carlos, Madrid, Spain.

0730-0301/2021/8-ART1 $15.00
https://doi.org/10.1145/3450626.3459875

ACM Reference Format:
Cristian Romero, Dan Casas, Jesús Pérez, and Miguel Otaduy. 2021. Learning Contact Corrections for Handle-Based Subspace Dynamics. *ACM Trans. Graph.* 40, 4, Article 1 (August 2021), 12 pages. https://doi.org/10.1145/3450626.3459875

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 [An et al. 2008; Barbič and James 2005; Hauser et al. 2003; Krysl et al. 2001; Pentland and Williams 1989]. 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 [Battaglia et al. 2016; Chang et al. 2017; Greydanus et al. 2019; Li et al. 2019; Sanchez-Gonzalez et al. 2018; Wiewel et al. 2019], 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 [Patel et al. 2020; Santesteban et al. 2020, 2019; Song et al. 2020].

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 3. 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, 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 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 Figure 1 and throughout the paper.

2 RELATED WORK

Our work is related to multiple methods for simulation and animation of deformable objects. We classify them into three categories: methods that combine global and local deformations for simulation or animation; subspace simulation or model reduction; and learning-based simulation.

2.1 Aggregation of Global and Local Deformations

Our subspace deformation model can be regarded as a relative of pose-space deformation (PSD) [Lewis et al. 2000], in the sense that local deformations are parameterized by the subspace pose, they are defined in an unposed setting, and they are mapped to the global setting as a function of pose. There are many differences though. PSD is typically used for artist- or mocap-driven animation, not for dynamic simulation; the subspace of PSD is a skeletal pose, not a generic linear deformation model; local corrections model internal deformations, not external interactions; and the mapping of local corrections is explicitly defined by the skinning transformation, not derived from an arbitrary global deformation model as in our case.

Nevertheless, the basic PSD scheme has been extended in multiple ways, and some bring closer ties to some of our model's features. EigenSkin [Kry et al. 2002], for instance, reduces the dimensionality of the correction field for human hands using principal component analysis, as we do. SMPL [Loper et al. 2015] models corrections of human bodies using blendshapes, which are trained from multiple scans. Bailey et al. [2018] recently proposed a machine-learning method to efficiently approximate complex character skinning rigs as a local nonlinear correction to the base linear skinning. Many other works also train correction models from example data, and the approach has been successfully applied to the skeletal animation of cloth [Wang et al. 2010]. As an alternative to learning corrections, the method of Wang et al. [2007] learns deformation gradients and then reconstructs the deformation. Other animation methods use more diverse definitions of pose, to extend beyond skeletal animation, such as local surface deformation for faces [Bickel et al. 2008] or cloth [Kavan et al. 2011; Zurdo et al. 2013]. The recent work of Song et al. [2020] uses an animation rig as a generalization of pose, and learns both global and local deformation as a function of the rig parameters. In contrast, we use a linear subspace model for global deformation and learn only nonlinear corrections, which largely simplifies the model, in particular for its use in dynamic simulation. 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 [Pons-Moll et al. 2015] and cloth [Santesteban et al. 2019].

Three major differences stand out in our work in contrast to PSD and its many derivatives. First, we define a compact subspace model of global plus local deformations for general deformable objects, not just skeleton-driven shapes. Second, our local corrections are also parameterized by external interactions, and hence allow data-driven contact simulation. And third, we use the subspace model for dynamic simulation, which requires derivatives of the model, and careful interpretation and approximation of these derivatives for efficiency.

If we look at dynamic simulations, there are other ways of combining global and local deformations. In this area, the focus is not necessarily on dimensionality reduction, and the local corrections are represented in a high-dimensional space. Separation into global and local deformations may have other advantages, such as better modeling of mechanical phenomena or faster solvers. Two prominent examples of aggregate global-local dynamic simulation methods are Eulerian-on-Lagrangian simulation [Fan et al. 2013] and multifarious hierarchies [Malgat et al. 2015]. Our model shares with these works the need to define a mapping for the local corrections, and we define the mapping similar to Malgat et al., through linearization of the global deformation. However, because in our case both global and local deformations are parameterized by the same subspace, we pay more attention to the interpretation of the mapping and its approximation in the context of dynamic simulation. To the best of our knowledge, no previous work applied this idea to linear subspace models. We show that the mapping can be defined by a modulation of the linear subspace, computed by finding gradients of the basis components.

2.2 Subspace Simulation Methods

In the context of dynamic simulation, the creation of subspace models allows fast approximation to the equations of motion, by ignoring high-frequency deformations. The most popular approach is to use a linear subspace model to approximate global deformations, built from modal analysis [Pentland and Williams 1989], principal component analysis of deformation examples [Krysl et al. 2001], modal derivatives [Barbič and James 2005], sparse frames [Brandt et al. 2018; Gilles et al. 2011], or sparse handles [Wang et al. 2015]. Some notable exceptions, which use nonlinear subspaces, are based on animation rigs [Hahn et al. 2012, 2013] or rotation-strain coordinates [Pan et al. 2015]. Recently, Fulton et al. [2019] introduced the use of variational autoencoders to automatically infer compact nonlinear subspaces for dynamic deformations. On the contrary, we use machine learning to model only nonlinear corrections, and we parameterize these corrections explicitly, as a function of both the linear subspace and external interactions. We achieve detailed deformations not shown by purely learning-based subspace models.

Linear subspace models have also been used to simulate only local deformations on top of some different global deformation model [Kim and James 2011]. Moreover, subspace local deformations can be aggregated with other deformations, such as pose-based blendshapes [Tapia et al. 2021]. As a way to increase the accuracy of subspace deformations, several works [Hahn et al. 2014; Xu and Barbič 2016] blend local linear subspace models in a pose-dependent manner. This approach can be regarded as an alternative to our local nonlinear model. However, finding the appropriate local linear models and blending functions is not a simple task for general deformable objects. We demonstrate that modern machine-learning methods allow one to bypass this task, learning a nonlinear model instead. Note also that in our model local deformations are parameterized by a combination of the global deformation and an interaction state, while the methods cited above use additional degrees of freedom.

A different approach to increase the accuracy of subspace simulation methods, particularly for contact-based interactions, is to locally enrich the simulation model. Harmon and Zorin [2013] enrich a linear subspace model with locally supported basis functions precomputed using a Boussinesq contact model. Teng et al. [2015] 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. Enrichment methods present pros and cons with respect to our approach, and we see them better suited for different types of applications. Our approach is highly optimized for modeling external interactions that admit a compact parameterization, such as interaction with rigid colliders. Enrichment methods, on the other hand, support general interaction, but cannot be optimized for particular interactions. An additional challenge for contact simulations with subspace methods is the efficient yet accurate evaluation of contact forces. To this end, Teng et al. [2014] presented a cubature method for self-collisions in subspace skeletal deformations.

2.3 Learning-Based Deformable Simulation

Prior to the explosion of neural-network methods, de Aguiar et al. [2010] designed a learning-based second-order model of cloth deformation with stability guarantees. Kim et al. [2013] showed how to encode complex dynamics of cloth using motion graphs. In recent years, and mostly based on neural networks, machine learning methods have been used in very diverse ways in deformable object simulation.

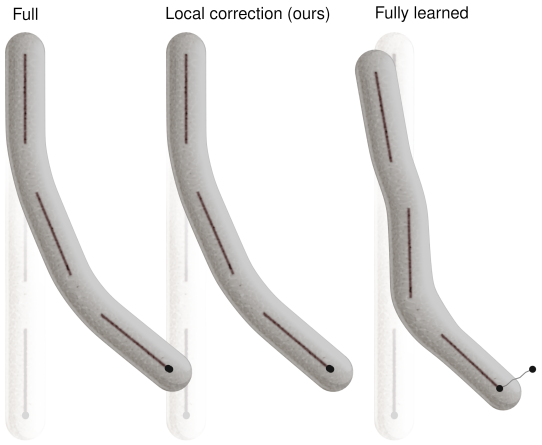

Fig. 2. Our subspace model (center) disentangles the deformations due to three different sources (global linear, local nonlinear internal, local nonlinear external), enabling an efficient learning of nonlinear corrections, and accurate matching of full simulations (left). Directly learning the full deformation, 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.

NNWarp [Luo et al. 2020] learns the correction between linear and nonlinear materials as a warping function, and thus simplifies the simulation of complex nonlinear materials. Holden et al. [2019] propose a learning-based representation of the full dynamic interaction between a dynamic object and some collider(s). Their method bears some similarities to ours; however, by trying to learn the full dynamic behavior, they pay a loss of detail and accuracy. They also model the deforming object using a subspace representation, but deformation detail is limited to the global linear subspace, whereas our approach learns nonlinear corrections with fine detail. Their learning model also takes as input the interaction space described by the configuration of the collider, but dynamics are strongly damped on test scenes, whereas our approach retains full dynamics of the linear subspace. As shown in Figure 2, learning local nonlinear corrections, as we do, is a simpler problem, and leads to higher detail and better generalization.

Several other works have modeled deformations driven by skeletal motion, a problem that falls in the scope of the PSD methods described above in Section 2.1. Some of the interesting developments include the use of convolutional networks for mesh-based deformations [Chentanez et al. 2020], and robust learning of deformation dynamics under scarce training data [Santesteban et al. 2020].

Beyond computer graphics, recent efforts on machine learning look at representations of the common invariants and/or processes

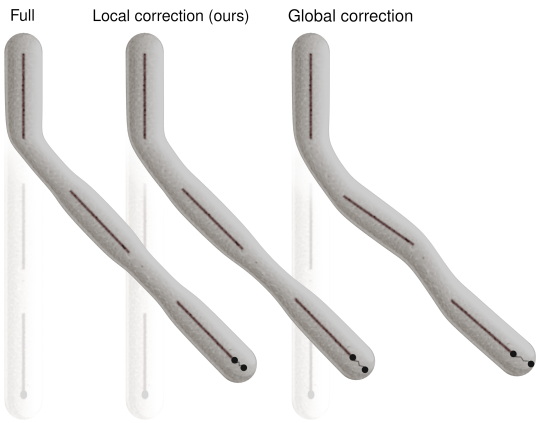

Fig. 3. The deformation behavior of a full simulation (left) is accurately modeled when nonlinear corrections are learned on a local setting (center). Global corrections are more difficult to learn, and suffer artifacts (right). In the example, both local and global corrections use the same training data and neural-network architecture.

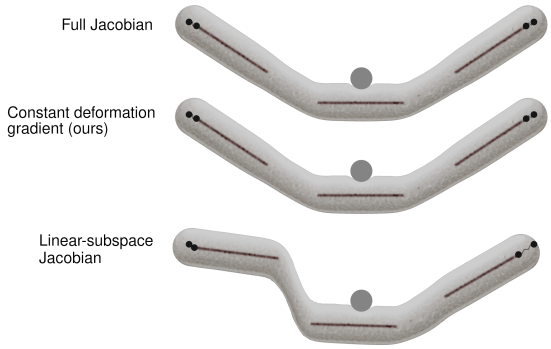

Fig. 4. To maximize runtime efficiency, we have evaluated different approximations to the Jacobian of our deformation model (3). The behavior with the full Jacobian (top) is accurately matched when we ignore the change in the deformation gradient (middle), as in (4). However, deformation errors are evident (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.

involved in mechanics. Some of the examples include modeling collisions and deformations using graph representations [Battaglia et al. 2016], producing generic neural-network representations of mechanical evolution using composable objects and their interactions [Chang et al. 2017], modeling multi-physics phenomena through learned particle-based models [Li et al. 2019], or modeling physical processes by learning invariants and training with measurable functions of these invariants [Greydanus et al. 2019].

3 CORRECTED SUBSPACE DEFORMATIONS

In this section, we present 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.

3.1 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. For the linear deformation, we choose the biharmonic generalized barycentric coordinates (BGBC) [Wang et al. 2015]. 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. Other choices of frame-based models [Brandt et al. 2018; Gilles et al. 2011] 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 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 [Lewis et al. 2000], we express the correction in a local setting. This choice simplifies learning and hence maximizes the accuracy of the correction field, as shown in Figure 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 approximation of a correction applied to the undeformed setting [Malgat et al. 2015]. Adding the linear and nonlinear components together, we can express our nonlinear subspace deformation model as:

$$\mathbf{x}(\mathbf{q}) = \mathbf{U}\,\mathbf{q} + \mathbf{F}(\mathbf{q})\,\mathbf{r}(\mathbf{q}). \tag{1}$$

In practice, we compute the deformation gradient on tetrahedral elements [Irving et al. 2004] and then perform a moving least-squares approximation on nodes [Müller et al. 2004]. In the remainder of the section, we drop the explicit dependency of $\mathbf{q}$ from the various terms in (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 (1) by reversing the order of $\mathbf{F}\,\mathbf{r}$ as $\mathrm{mat}(\mathbf{r})\,\mathrm{vec}(\mathbf{F})$. Furthermore, the deformation gradient can be expressed through a linear operation $\mathrm{mat}(\nabla)$ on the subspace deformation $\mathbf{u}$. Using the matrices $\mathrm{mat}(\mathbf{r})$ and $\mathrm{mat}(\nabla)$, we rewrite our corrected subspace model (1) as

$$\mathbf{x} = (\mathbf{I} + \mathbf{W})\;\mathbf{U}\,\mathbf{q}, \tag{2}$$

with $\mathbf{W} = \mathrm{mat}(\mathbf{r})\,\mathrm{mat}(\nabla)$. 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 (1), after computing $\mathbf{F}$ explicitly. For the transformation of forces to the subspace, however, it is convenient to analyze the basis modulation (2), as we see next.

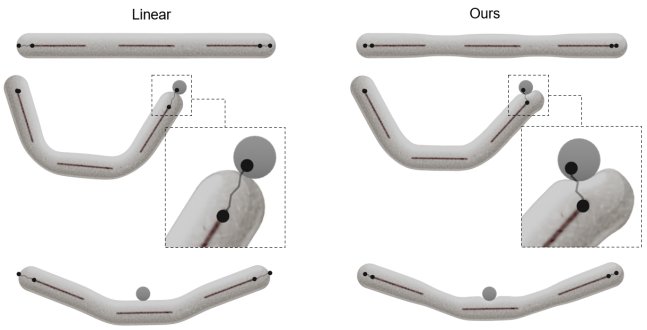

Fig. 5. This example highlights the aggregation of deformations in our model. The left column shows the linear deformation $\mathbf{U}\,\mathbf{q}$. The right column shows the addition of nonlinear corrections. The top-right image includes only internal corrections $\mathbf{r}_{\text{int}}$, which restore 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.

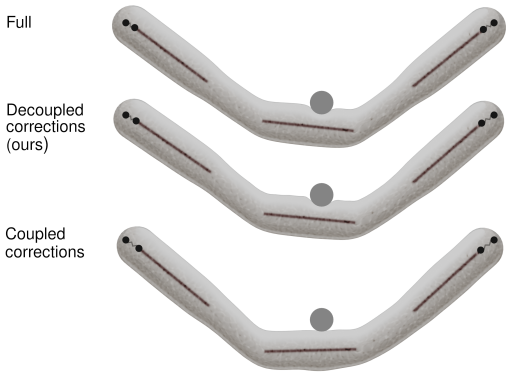

Fig. 6. The nonlinear deformation of a full simulation (top) is accurately matched when internal and external corrections are learned separately (center). Trying to learn both types of corrections together complicates data generation and learning, and fails to reproduce external contact-driven corrections (bottom). In the example, the complexity of the neural-network architecture for coupled learning is equal to the added complexity of the decoupled architectures.

3.2 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 (1) and (2), we obtain:

$$\mathbf{J} = \frac{\partial \mathbf{x}}{\partial \mathbf{q}} = (\mathbf{I} + \mathbf{W})\ \mathbf{U} + \mathbf{F}\,\frac{\partial \mathbf{r}}{\partial \mathbf{q}}. \tag{3}$$

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 $\mathbf{f}_{\mathbf{q}} = \mathbf{J}^T\,\mathbf{f}_{\mathbf{x}}$.

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 (3). Figure 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 [Müller and Gross 2004; Xu et al. 2015], but the effect is even milder for subspace deformations.

Based on our experiments, we conclude to approximate the Jacobian (3) as

$$\mathbf{J} \approx \mathbf{U} + \mathbf{F}\,\frac{\partial \mathbf{r}}{\partial \mathbf{q}}. \tag{4}$$

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.

3.3 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 [Gast et al. 2015; Martin et al. 2011]. As done by Pan et al. [2015], 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

$$\mathbf{q} = \arg\min \frac{1}{2\,h^2}\,(\mathbf{x} - \mathbf{x}^*)^T\,\mathbf{M}\,(\mathbf{x} - \mathbf{x}^*) + V(\mathbf{x}). \tag{5}$$

To time-step the rigid frames in the BGBC reduced DoFs $\mathbf{q}$, we parameterize the rotations in their tangent-space [Taylor and Kriegman 1994]. $\mathbf{M}$ 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 [Smith et al. 2018] with tetrahedral FEM discretization.

The optimality of (5) yields the following nonlinear equations:

$$\mathbf{J}^T\,\frac{1}{h^2}\,\mathbf{M}\,(\mathbf{x} - \mathbf{x}^*) - \mathbf{J}^T\,\mathbf{f}_{\mathbf{x}} = 0. \tag{6}$$

We solve these equations using a quasi-Newton method, where we approximate the Hessian of (5) as $\mathbf{J}^T\left(\frac{1}{h^2}\,\mathbf{M} - \frac{\partial \mathbf{f}_{\mathbf{x}}}{\partial \mathbf{x}}\right)\mathbf{J}$.

As done often for subspace methods, we use a cubature approximation of forces and Hessians [An et al. 2008]. After training cubature points $\{\mathbf{x}_k\}$ and weights $\{w_k\}$ [von Tycowicz et al. 2013], one can approximate subspace forces (and similarly their Jacobian) as $\mathbf{f}_{\mathbf{q}} \approx \sum_k w_k\,\mathbf{J}_k^T\,\mathbf{f}_{\mathbf{x},k}$, 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.

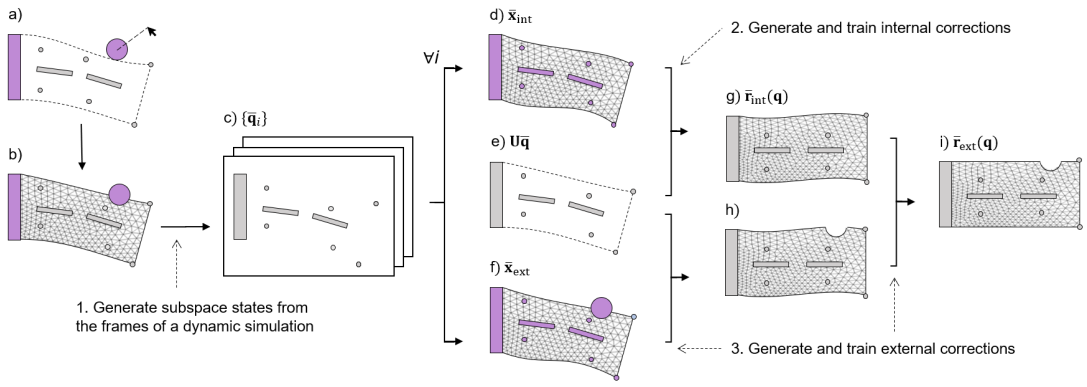

Fig. 7. Data generation pipeline. First, a) we interactively record a linear-subspace dynamic simulation, and b) use the recorded interaction to generate an offline full dynamic simulation. c) For each frame, we extract a representative subspace state $\bar{\mathbf{q}}$. Then, we fix the DoFs corresponding 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.

4 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 corrections $\mathbf{r}_{\text{ext}}$, which correct the additional deviation introduced by contact. This separation into internal and external corrections, highlighted in Figure 5, simplifies the generation of representative training data, and hence maximizes the accuracy of the aggregate correction, as shown in Figure 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 Figure 7. And we conclude with a discussion of implementation details of the learning architecture.

4.1 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, $\mathbf{U}\,\mathbf{q} + \mathbf{F}\,\mathbf{r}_{\text{int}}$. Figure 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:

$$\mathbf{r} = \mathbf{r}_{\text{int}}(\mathbf{q}) + \mathbf{r}_{\text{ext}}(\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z}). \tag{7}$$

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.

4.2 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 [Barbič and James 2005]. 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.

Table 1. Model size and performance data of the examples shown in the paper. 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	Handles (points/frames)	Colliders	Full mesh (tris or tets)	Cubature points	PCA corr. (int/ext)	Neurons (int/ext)	Train frames (int/ext)	Ours fps	Linear fps	Full fps
Jelly+Circle	8/1	1	12,469	599	–/50	–/200	–/9,747	201	444	22.7
Jelly+Comb	8/1	1 (rot.)	12,469	599	–/100	–/3000	–/187,039	135	410	19.0
Jelly+Star	8/1	1 (rot.)	12,469	599	–/100	–/3000	–/270,000	145	432	21.2
Accordion	16/2	0	17,457	836	15/–	200/–	5,880/–	85	119	10.4
Auxetic	16/2	0	12,921	615	15/–	200/–	6,291/–	99	122	11.7
Worm	0/3	1 (×6)	10,656	505	15/120	200/2,000	3,730/138,379	13.6 (6.9)	403	2.5
Bunny	24/1	1	17,062	341	–/120	–/1,500	–/17,425	48	87	7.1
Finger	0/4	1	10,163	203	–/120	–/2,000	–/25,128	10.9	143	1.9

A detailed representation of the data generation pipeline is shown in Figure 7.

To generate representative states, we first execute fast dynamic simulations using the baseline linear subspace model (Figure 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 (Figure 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 $\{\mathbf{U}\bar{\mathbf{q}}_i\}$ (Figure 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 (Figure 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 (1):

$$\bar{\mathbf{r}}_{\text{int},i}(\bar{\mathbf{q}}_i) = \mathbf{F}(\bar{\mathbf{q}}_i)^{-1}\,\left(\bar{\mathbf{x}}_{\text{int},i} - \mathbf{U}\,\bar{\mathbf{q}}_i\right). \tag{8}$$

At this point, we use these internal correction targets to train the internal correction $\mathbf{r}_{\text{int}}(\mathbf{q})$ (Figure 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 $\bar{\mathbf{q}}_i$ (Figure 7, f). From the subspace and full states, we obtain target external corrections simply by undoing the subspace formulation (1) (Figure 7, h). However, this time we also subtract the internal corrections:

$$\bar{\mathbf{r}}_{\text{ext},i}(\mathbf{A}(\bar{\mathbf{q}}_i)^{-1}\,\mathbf{z}_i) = \mathbf{F}(\bar{\mathbf{q}}_i)^{-1}\,\left(\bar{\mathbf{x}}_{\text{ext},i} - \mathbf{U}\,\bar{\mathbf{q}}_i\right) - \bar{\mathbf{r}}_{\text{int},i}(\bar{\mathbf{q}}_i). \tag{9}$$

At this point, we use these external correction targets to train the external correction $\mathbf{r}_{\text{ext}}(\mathbf{A}(\mathbf{q})^{-1}\,\mathbf{z})$ (Figure 7, i).

4.3 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 (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.

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 and video, 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 3.3. Recall that the approximation of the Jacobian of our subspace model (3), requires the Jacobian of nonlinear corrections $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}$, as shown in (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 $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}^T\,\mathbf{v}$. 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 $\frac{\partial \mathbf{r}}{\partial \mathbf{q}}\,\mathbf{v} = \frac{\partial \mathbf{y}}{\partial \mathbf{w}}^T\,\mathbf{v}$.

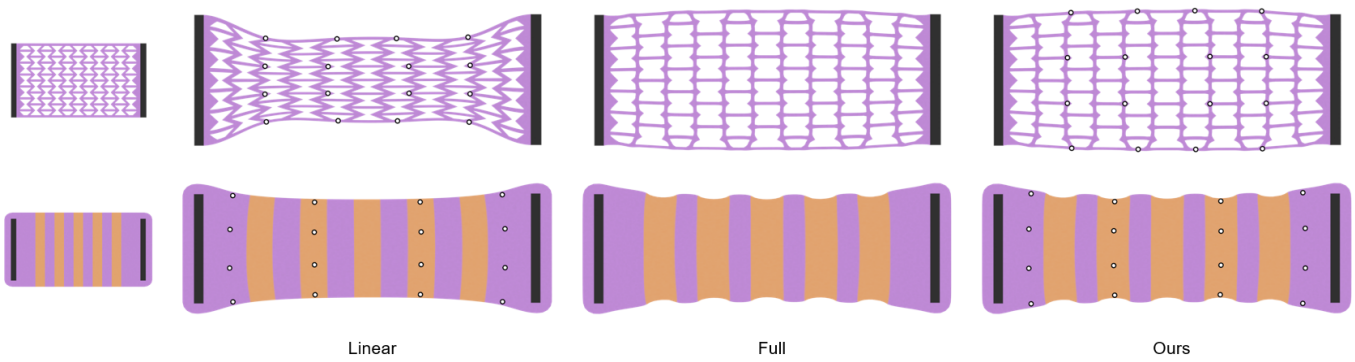

Fig. 8. We simulate two types of microstructures, an auxetic structure (top) and accordion-like heterogeneous stripes (bottom), with subspace models defined by just 2 frames and 16 points. A purely linear model is incapable of showing nonlinear effects produced by material heterogeneity, such as the negative Poisson's ratio of the auxetic structure and the ripples of the striped structure. Our method practically matches the full solution, yet 9× faster.

5 EXPERIMENTS AND DISCUSSION

Table 1 summarizes the settings, model size, and performance of the examples shown in the paper. Please watch the accompanying video for results produced by spontaneous user interactions, which demonstrate generalization outside the training data. 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 [Schumacher et al. 2015]. Nevertheless, numerical coarsening methods assume a linear response of the microstructure with respect to coarse DoFs [Chen et al. 2018; Kharevych et al. 2009; Torres et al. 2016].

We have explored the use of our model for the simulation of microstructures, by augmenting the linear subspace model with nonlinear internal corrections. In Figure 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 Figure 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 [Harmon and Zorin 2013; Teng et al. 2015] 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 2, we compare quantitatively the accuracy of the subspace jelly model for three different colliders: (i) the comb-like collider of Figure 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

Table 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 Figure 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 Figure 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		

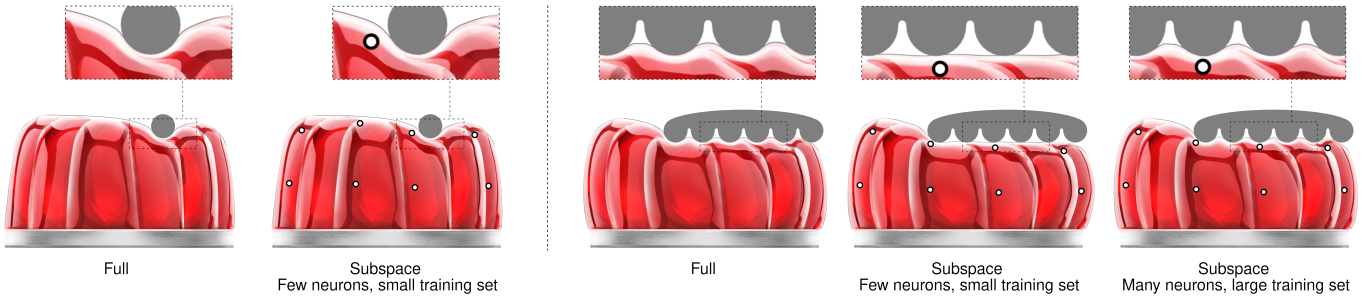

Fig. 9. Our subspace model successfully represents contact deformations due to both small and large colliders with high-resolution features. Nevertheless, large colliders with larger configuration space (e.g., the comb-like object on the right) require a larger training set and larger network architecture. A quantitative analysis of the error is summarized in Table 2.

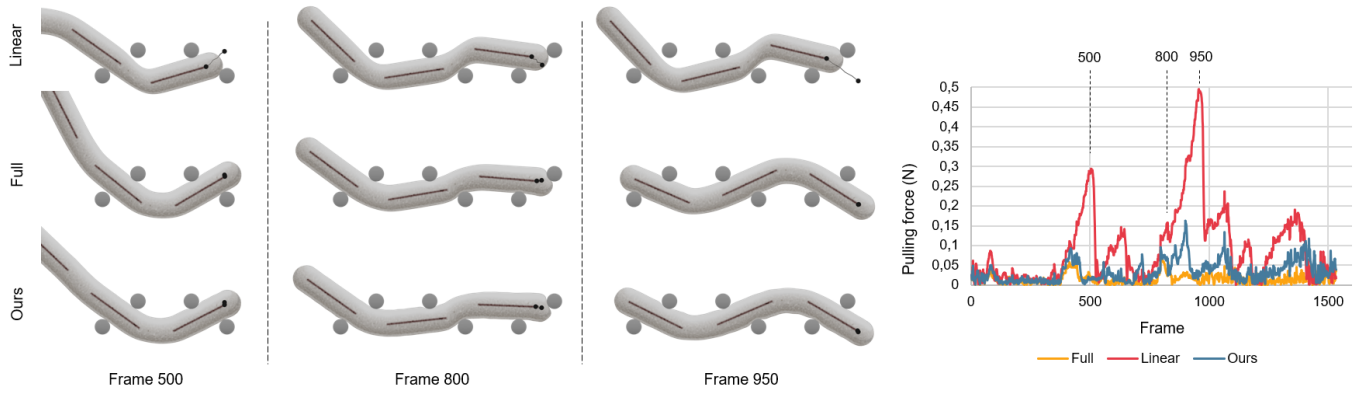

Fig. 10. In this example, we pull a worm-like soft robot through a narrow passage. A purely linear subspace model (top) suffers strong distortions (see the soft regions between bones), and cannot deform locally to conform to the shape of the pins. Our model (bottom), even though it is built from a subspace 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× larger than the full model. With our model, the peak force is just 1.8× 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.

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 Figure 9. The star collider of Figure 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 1.

`Soft-Robot Worm`. We have modeled a worm-like soft robot, with three bones surrounded by soft material (See Figure 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 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. Figure 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.

We also test the worm model on a more complex setting, well outside the training settings. Figure 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

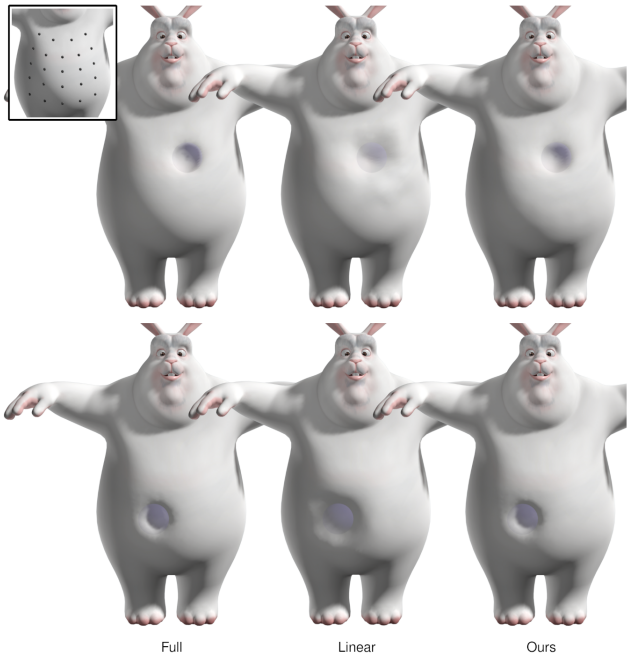

Fig. 11. This model of Big Buck Bunny contains a soft-tissue layer on top of a rigid core. We learn contact-driven corrections to augment a linear subspace model (point frames highlighted in the inset). As shown in the examples, with our method contact-driven deformations do not suffer the resolution limitations of the linear model, and match closely the deformations of a full simulation model.

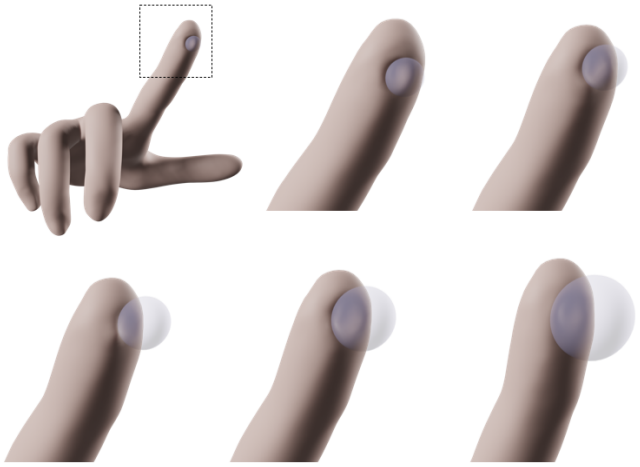

Fig. 12. We model a finger with just 3 frame handles located at the phalanges. The full nonlinear deformation of the surrounding tissue is captured by our learning-based corrections. Moreover, in this example we learn external corrections as a function of the size of the spherical collider, opening the possibility of using parametric shape models.

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 pulling force. With our model, the peak force is 1.8× larger than with the full model. With the purely linear model, however, the peak force is 5.6× 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 Figure 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 Figure 3. We cannot claim that it is not possible to learn full deformations directly; in fact Holden et al. [2019] 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. Figure 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 Figure 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 4.2. 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. Please watch the video to see contact deformations together with dynamics.

`Finger`. To conclude, we have also used our model to simulate deformations of a soft skeletal finger model, shown in Figure 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 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 [Loper et al. 2015; Wu et al. 2016].

6 LIMITATIONS AND FUTURE WORK

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 [Pentland and Williams 1989] or modal derivatives [Barbič and James 2005]. 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 4.1, 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 4.2, 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 [Casas and Otaduy 2018; Holden et al. 2019], or by building a recurrent learning architecture [Santesteban et al. 2019]. 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 [Verschoor et al. 2020].

ACKNOWLEDGMENTS

We wish to thank the anonymous reviewers for their helpful comments. We also thank Igor Santesteban for help with the comparisons, and Jorge López for the artwork in the worm and jelly examples. This work was funded in part by the European Research Council (ERC Consolidator Grant 772738 *TouchDesign*) and the Spanish Ministry of Science (grant RTI2018-098694-B-I00 *VizLearning*).

REFERENCES

Steven S. An, Theodore Kim, and Doug L. James. 2008. Optimizing Cubature for Efficient Integration of Subspace Deformations. *ACM Trans. Graph.* 27, 5, Article 165 (2008).

Stephen W. Bailey, Dave Otte, Paul Dilorenzo, and James F. O'Brien. 2018. Fast and Deep Deformation Approximations. *ACM Trans. Graph.* 37, 4, Article 119 (2018).

Jernej Barbič and Doug L. James. 2005. Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models. *ACM Trans. Graph.* 24, 3 (July 2005), 982–990.

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

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

Christopher Brandt, Elmar Eisemann, and Klaus Hildebrandt. 2018. Hyper-Reduced Projective Dynamics. *ACM Trans. Graph.* 37, 4, Article 80 (2018).

Dan Casas and Miguel A. Otaduy. 2018. Learning Nonlinear Soft-Tissue Dynamics for Interactive Avatars. *Proc. ACM Comput. Graph. Interact. Tech.* 1, 1, Article 10 (July 2018), 15 pages.

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

Jiong Chen, Hujun Bao, Tianyu Wang, Mathieu Desbrun, and Jin Huang. 2018. Numerical Coarsening Using Discontinuous Shape Functions. *ACM Trans. Graph.* 37, 4, Article 120 (2018).

Nuttapong Chentanez, Miles Macklin, Matthias Müller, Stefan Jeschke, and Tae-Yong Kim. 2020. Cloth and Skin Deformation with a Triangle Mesh Based Convolutional Neural Network. *Computer Graphics Forum* 39, 8 (2020), 123–134.

Edilson de Aguiar, Leonid Sigal, Adrien Treuille, and Jessica K. Hodgins. 2010. Stable Spaces for Real-Time Clothing. *ACM Trans. Graph.* 29, 4 (2010).

Ye Fan, Joshua Litven, David I. W. Levin, and Dinesh K. Pai. 2013. Eulerian-on-lagrangian Simulation. *ACM Trans. Graph.* 32, 3 (2013).

Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. 2019. Latent-space Dynamics for Reduced Deformable Simulation. *Computer Graphics Forum* 38, 2 (2019), 379–391.

Theodore F Gast, Craig Schroeder, Alexey Stomakhin, Chenfanfu Jiang, and Joseph M Teran. 2015. Optimization integrator for large time steps. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 21, 10 (2015), 1103–1115.

Benjamin Gilles, Guillaume Bousquet, Francois Faure, and Dinesh K. Pai. 2011. Frame-Based Elastic Models. *ACM Trans. Graph.* 30, 2 (2011).

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. 2019. Hamiltonian Neural Networks. In *Advances in Neural Information Processing Systems*, Vol. 32. 15379–15389.

Fabian Hahn, Sebastian Martin, Bernhard Thomaszewski, Robert Sumner, Stelian Coros, and Markus Gross. 2012. Rig-Space Physics. *ACM Trans. Graph.* 31, 4, Article 72 (July 2012), 8 pages.

Fabian Hahn, Bernhard Thomaszewski, Stelian Coros, Robert W. Sumner, Forrester Cole, Mark Meyer, Tony DeRose, and Markus Gross. 2014. Subspace Clothing Simulation Using Adaptive Bases. *ACM Trans. Graph.* 33, 4 (2014).

Fabian Hahn, Bernhard Thomaszewski, Stelian Coros, Robert W. Sumner, and Markus Gross. 2013. Efficient Simulation of Secondary Motion in Rig-Space. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 165–171.

David Harmon and Denis Zorin. 2013. Subspace Integration with Local Deformations. *ACM Trans. Graph.* 32, 4, Article 107 (July 2013), 10 pages.

Kris K. Hauser, Chen Shen, and James F. O'Brien. 2003. Interactive Deformation Using Modal Analysis with Constraints. In *Proceedings of the Graphics Interface*. 247–256.

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

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

Ladislav Kavan, Dan Gerszewski, Adam W. Bargteil, and Peter-Pike Sloan. 2011. Physics-Inspired Upsampling for Cloth Simulation in Games. *ACM Trans. Graph.* 30, 4 (2011).

Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. 2009. Numerical Coarsening of Inhomogeneous Elastic Materials. *ACM Trans. on Graphics* 28, 3 (2009), 51:1–51:8.

Doyub Kim, Woojong Koh, Rahul Narain, Kayvon Fatahalian, Adrien Treuille, and James F. O'Brien. 2013. Near-Exhaustive Precomputation of Secondary Cloth Effects. *ACM Trans. Graph.* 32, 4 (2013).

Theodore Kim and Doug L. James. 2011. Physics-Based Character Skinning Using Multi-Domain Subspace Deformations. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 63–72.

Paul G. Kry, Doug L. James, and Dinesh K. Pai. 2002. EigenSkin: Real Time Large Deformation Character Skinning in Hardware. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 153–159.

P. Krysl, S. Lall, and J. E. Marsden. 2001. Dimensional model reduction in non-linear finite element dynamics of solids and structures. *Internat. J. Numer. Methods Engrg.* 51, 4 (2001), 479–504.

J. P. Lewis, Matt Cordner, and Nickson Fong. 2000. 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)*. ACM Press/Addison-Wesley Publishing Co., USA, 165–172.

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

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-person Linear Model. *ACM Trans. Graph.* 34, 6, Article 248 (Oct. 2015), 16 pages.

R. Luo, T. Shao, H. Wang, W. Xu, X. Chen, K. Zhou, and Y. Yang. 2020. NNWarp: Neural Network-Based Nonlinear Deformation. *IEEE Transactions on Visualization and Computer Graphics* 26, 4 (2020), 1745–1759.

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

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. 2011. Example-Based Elastic Materials. *ACM Trans. Graph.* 30, 4, Article 72 (2011).

Matthias Müller and Markus Gross. 2004. Interactive Virtual Materials. In *Proceedings of Graphics Interface*. 239–246.

M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa. 2004. Point Based Animation of Elastic, Plastic and Melting Objects. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 141–151.

Zherong Pan, Hujun Bao, and Jin Huang. 2015. Subspace Dynamic Simulation Using Rotation-Strain Coordinates. *ACM Trans. Graph.* 34, 6, Article 242 (Oct. 2015), 12 pages.

Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. 2020. 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.

Alex Pentland and John Williams. 1989. Good Vibrations: Modal Dynamics for Graphics and Animation. *Computer Graphics* 23, 3 (1989), 215–222.

Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J. Black. 2015. Dyna: A Model of Dynamic Human Shape in Motion. *ACM Trans. Graph.* 34, 4, Article 120 (July 2015), 14 pages.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin A. Riedmiller, Raia Hadsell, and Peter Battaglia. 2018. Graph Networks as Learnable Physics Engines for Inference and Control. In *ICML*. 4467–4476.

Igor Santesteban, Elena Garces, Miguel A. Otaduy, and Dan Casas. 2020. SoftSMPL: Data-driven Modeling of Nonlinear Soft-tissue Dynamics for Parametric Humans. *Computer Graphics Forum* 39, 2 (2020), 65–75.

Igor Santesteban, Miguel A. Otaduy, and Dan Casas. 2019. Learning-Based Animation of Clothing for Virtual Try-On. *Computer Graphics Forum* 38, 2 (2019), 355–366.

Christian Schumacher, Bernd Bickel, Jan Rys, Steve Marschner, Chiara Daraio, and Markus Gross. 2015. Microstructures to Control Elasticity in 3D Printing. *ACM Trans. Graph.* 34, 4 (2015), 136:1–136:13.

Breannan Smith, Fernando De Goes, and Theodore Kim. 2018. Stable Neo-Hookean Flesh Simulation. *ACM Trans. Graph.* 37, 2, Article 12 (March 2018), 15 pages.

Steven L. Song, Weiqi Shi, and Michael Reed. 2020. Accurate Face Rig Approximation with Deep Differential Subspace Reconstruction. *ACM Trans. Graph.* 39, 4 (2020). https://doi.org/10.1145/3386569.3392491

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

Camillo J. Taylor and David J. Kriegman. 1994. *Minimization on the Lie Group SO(3) and Related Manifolds*. Technical Report. Yale University.

Yun Teng, Mark Meyer, Tony DeRose, and Theodore Kim. 2015. Subspace Condensation: Full Space Adaptivity for Subspace Deformations. *ACM Trans. Graph.* 34, 4, Article 76 (July 2015), 9 pages.

Yun Teng, Miguel A. Otaduy, and Theodore Kim. 2014. Simulating Articulated Subspace Self-Contact. *ACM Trans. Graph.* 33, 4 (2014).

Rosell Torres, Alejandro Rodríguez, José M. Espadero, and Miguel A. Otaduy. 2016. High-resolution Interaction with Corotational Coarsening Models. *ACM Trans. Graph.* 35, 6 (2016), 211:1–211:11.

Mickeal Verschoor, Dan Casas, and Miguel A. Otaduy. 2020. Tactile Rendering Based on Skin Stress Optimization. *ACM Trans. Graph.* 39, 4, Article 90 (2020).

Christoph von Tycowicz, Christian Schulz, Hans-Peter Seidel, and Klaus Hildebrandt. 2013. An Efficient Construction of Reduced Deformable Objects. *ACM Trans. Graph.* 32, 6, Article 213 (2013).

Huamin Wang, Florian Hecht, Ravi Ramamoorthi, and James F. O'Brien. 2010. Example-Based Wrinkle Synthesis for Clothing Animation. *ACM Trans. Graph.* 29, 4, Article 107 (July 2010).

Robert Y. Wang, Kari Pulli, and Jovan Popović. 2007. Real-Time Enveloping with Rotational Regression. In *ACM SIGGRAPH 2007 Papers* (San Diego, California) *(SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA.

Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. 2015. Linear Subspace Design for Real-Time Shape Deformation. *ACM Trans. Graph.* 34, 4, Article 57 (2015).

S. Wiewel, M. Becher, and N. Thuerey. 2019. Latent Space Physics: Towards Learning the Temporal Evolution of Fluid Flow. *Computer Graphics Forum* 38, 2 (2019), 71–82.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. 2016. 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*. 82–90.

Hongyi Xu and Jernej Barbič. 2016. Pose-Space Subspace Dynamics. *ACM Trans. Graph.* 35, 4, Article 35 (July 2016), 14 pages.

Hongyi Xu, Funshing Sin, Yufeng Zhu, and Jernej Barbič. 2015. Nonlinear Material Design Using Principal Stretches. *ACM Trans. Graph.* 34, 4, Article 75 (2015).

J. S. Zurdo, J. P. Brito, and M. A. Otaduy. 2013. Animating Wrinkles by Example on Non-Skinned Cloth. *IEEE Transactions on Visualization & Computer Graphics* 19, 01 (2013), 149–158.