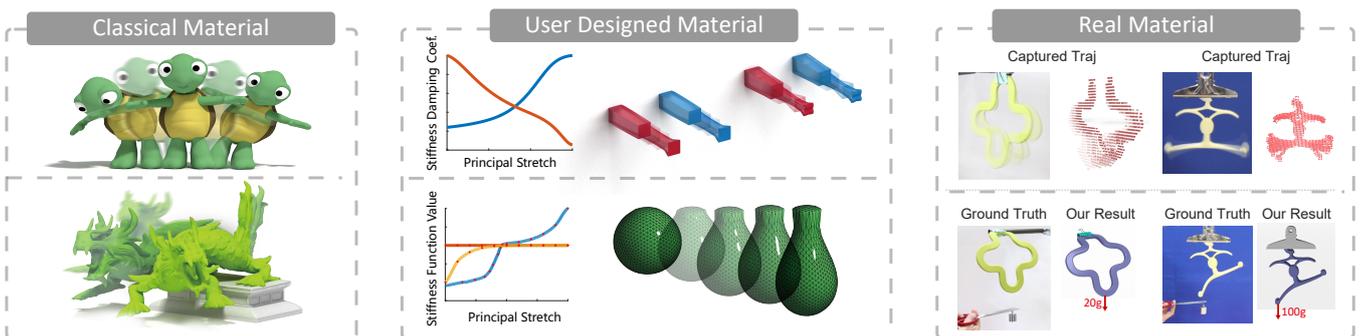


# Learning Elastic Constitutive Material and Damping Models

Bin Wang<sup>1</sup>, Yuanmin Deng<sup>†2</sup>, Paul Kry<sup>3</sup>, Uri Ascher<sup>4</sup>, Hui Huang<sup>‡5</sup> , Baoquan Chen<sup>§6</sup>

<sup>1</sup>Beijing Film Academy <sup>2</sup>Shandong University <sup>3</sup>McGill University <sup>4</sup>University of British Columbia <sup>5</sup>Shenzhen University <sup>6</sup>Peking University



**Figure 1:** Our parametric material model learns a correction to a nominal material model from kinematic data alone, allowing us to accurately capture the nonlinearity of different constitutive material models. Left: classical nonlinear constitutive material. Middle: user designed elasticity and damping. Right: real world material.

## Abstract

Commonly used linear and nonlinear constitutive material models in deformation simulation contain many simplifications and only cover a tiny part of possible material behavior. In this work we propose a framework for learning customized models of deformable materials from example surface trajectories. The key idea is to iteratively improve a correction to a nominal model of the elastic and damping properties of the object, which allows new forward simulations with the learned correction to more accurately predict the behavior of a given soft object. Space-time optimization is employed to identify gentle control forces with which we extract necessary data for model inference and to finally encapsulate the material correction into a compact parametric form. Furthermore, a patch based position constraint is proposed to tackle the challenge of handling incomplete and noisy observations arising in real-world examples. We demonstrate the effectiveness of our method with a set of synthetic examples, as well with data captured from real world homogeneous elastic objects.

## CCS Concepts

• **Computing methodologies** → Collision detection; • **Hardware** → Sensors and actuators; PCB design and layout;

## 1. Introduction

The simulation of deformable objects is ubiquitous in computer graphics and robotics research due to the large number of varied applications, including animation, movie making, medical treatment and manufacturing. These applications benefit from high fidelity deformation simulation, which is dependent upon the underlying

constitutive material model. Subsequently, the material parameters must be carefully tuned in order to fit empirical data. Data-driven methods have recently exhibited great potential in this direction. Advanced scanning technologies can be used to faithfully capture a deformation behavior under external force, and in turn the data can be used to estimate the parameters of the mathematical model.

However, there is currently no standard method for choosing the appropriate constitutive models, especially for large deformations, real-world materials, heterogeneous models, and artist designed cartoon physics. The situation for modeling damping is even worse. Indeed, there is no agreement about models in the mechanical en-

<sup>†</sup> Joint first author

<sup>‡</sup> Corresponding author (hhzhiyan@gmail.com)

<sup>§</sup> Corresponding author (baoquan@pku.edu.cn)

engineering literature which can describe various damping effects in a unified way. Many use the Rayleigh model, which can be inadequate for visual purposes [XB17]. Based on these observations, in this paper we propose a more general material inference framework obtained by directly learning constitutive laws rather than fitting parameters from real-world motion data. Learning a constitutive model is challenging because: (1) the model should be able to encapsulate all the variations of material properties in a generic way; (2) there may be no obvious source of training data; and (3) real-world motion data is typically sparse and noisy. We address these challenges and present the following contributions:

- a constitutive material model designed as a combination of empirical baseline model and a parametric correction;
- an inverse learning framework capable of learning a complex constitutive material from sparse motion trajectories; and
- a differentiable patch based position constraint for probabilistic correspondences, which allows our system to work faithfully on real-world captured data.

Fig. 1 shows a preview of our approach and results. We demonstrate the performance on several problems, including synthetic examples, coarsening applications, and captured data. The variety of results described in Section 7 leads us to believe that our work contributes a useful technique and an important step in advancing data-driven constitutive elastodynamic force models.

## 2. Related work

Data-driven material parameter optimization offers great potential for computer graphics applications, such as fabrics, soft objects, and human organs and faces [PDJ\*01; SLS04; BT07; WOR11; MBT\*12; BBO\*09]. Bickel et al. [BBO\*09] fit material parameters with an incremental loading strategy to better approximate nonlinear strain-stress relationships. Wang et al. [WOR11] propose a piecewise linear elastic model to reproduce nonlinear, anisotropic stretching and bending of cloth. Other approaches directly optimize nonlinear stress-strain curves based on measurements [MBT\*12], and estimate internal friction [MTB\*13]. Miguel et al. [MMO16] model example based cloth and elastic solid materials with energy functions.

A common limitation with previous methods is that they require a dense force displacement field. While Bhat et al. [BTH\*03] avoid the need for force capture by using video tracking of cloth, they still assume a trivial cloth reference shape. Yang et al. [YLL17] present a learning-based algorithm to recover material properties of cloth from videos, using training data sets generated by physics-based simulators. Here, material type estimation is the focus due to inconsistencies between real and synthetic data and sparse material space sampling. Davis et al. [DBC\*17] estimate material and damping properties by extracting small vibration modes from high-speed and regular frame-rate video.

Both Wang et al. [WYY\*15] and the more recent Hahn et al. [HBBC19] estimate material properties from partially observed surface trajectories of an object's passive dynamics. A gradient-free Nelder-Mead optimization algorithm is employed by Wang et al., whereas Hahn et al. compute the gradient with respect to material parameters using either direct sensitivity analysis or an adjoint state

method. Our work has a similar setting, but focuses on correcting the errors that arise when starting with simple elastic and damping force models and differs in the methods used in the pipeline.

Another popular trend in computer graphics is to directly fit parametric functions as a material description. Xu et al. [XSZB15] provide a method to design isotropic and anisotropic (orthotropic) nonlinear solid elastic materials using a piecewise spline interface, which can provide local control on deformation behavior. A hyper-elasticity model based on energy addends [MMO16] allows modeling of various nonlinear elasticity effects in a separable manner. Instead of explicitly modeling the stress-strain relationship, Martin et al. [MTGG11] and Schumacher et al. [STC\*12] promote an art-directed approach to solid simulation, which constructs a manifold of preferred deformation examples, to which the object is guided.

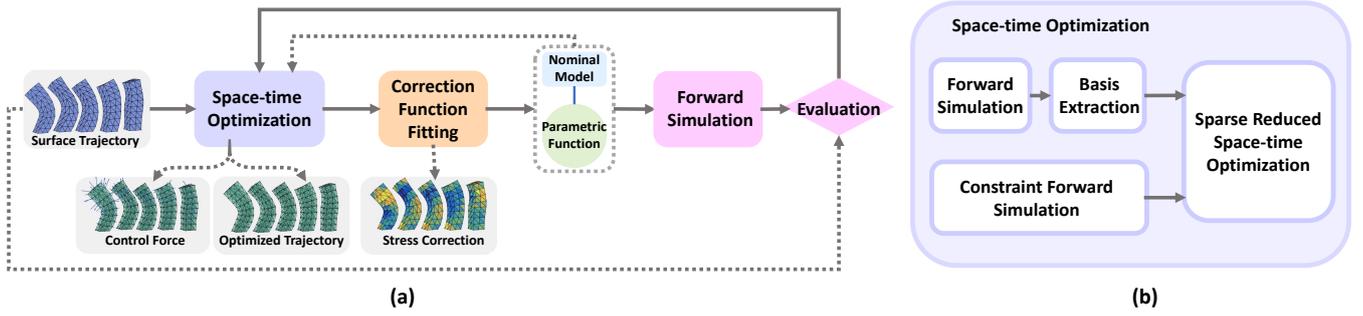
Also related are coarsening techniques, for instance, in applications for computational design for fabrication [CLSM15; PZM\*15; CLMK17; CLK\*19], where equivalent physics based models are important. Kharevych et al. [KMOD09] take an energy based approach to coarsening composite elastic objects through the use of global harmonic displacements. Nesme et al. [NKJF09] create nonlinear shape functions and projected fine-level mass, stiffness, and damping matrices to produce coarse composite elements, while Torres et al. [TREQ16] introduce an improved element based coarsening method that deals with corotation.

In comparison to elasticity modeling, few publications have focused on the design of damping models. Xu et al. [XB17] present a method for designing anisotropic and/or nonlinear damping effects. Banderas et al. [MSBAO18] model damping based on dissipation potentials using strain rate to control damping. (We use a similar approach.) Targeting cloth hysteresis effects, Miguel et al. [MTB\*13] propose an internal friction model based on an augmented reparameterization of Dahl's model. In our work, inspired by Xu et al. [XSZB15], we use principal stretches to formulate nonlinear elastic and damping force corrections. However, instead of using splines, we adopt radial basis functions (RBF) to parameterize the material correction.

Neural networks have also been proposed as solutions in the study of diverse phenomena which are not as yet accessible to physical modeling [GGJW91; GPZHA98]. Jung and Ghaboussi [JG06] modeled rate-dependent materials. Stefanos and Gyan [SG15] used the length of the strain trajectory traced by a material point, also called intrinsic time, as an additional input parameter in training because of its importance in cyclic and transient loading situations. In the computer graphics community, deep learning technology for deformation modeling has gradually gained more attention. The DeepWarp technique [LSW\*18] attempts to learn a mapping from a linear elasticity simulation to its nonlinear counterpart. Fulton et al. [FMD\*19] perform time integration of the elastodynamic system in a learned nonlinear reduced latent space, which is represented using a neural network. In comparison, our work focuses on the underlying constitutive model for FEM simulation and employs an RBF representation.

## 3. Overview

An overview of our progressive material learning framework is



**Figure 2:** Schematic overview of our data-driven parametric material learning framework. Our algorithm iteratively learns a correction to a nominal material model that allows us to accurately reproduce the captured trajectory, even when the nominal model differs significantly from the corrected one.

given in Fig. 2(a). Under this framework, the material model is represented in a two component manner, as described in Section 4. A baseline constitutive elastodynamic model, called *nominal model*, is assumed to be given. The baseline model correction is encapsulated in a *parametric function*. The core of our approach is to infer this parametric function through trajectory fitting.

The input to the system is a set of *surface trajectories* of an object moving dynamically, unforced, in response to an initial perturbation. The main loop alternates between solving a *space-time optimization problem* (Section 5), and *fitting a correction function* (Section 6). The space-time optimization injects a gentle *control force* to keep the system trajectories close to desired input trajectories while still obeying Newton’s physical laws. The key insight is that this gentle control force identifies what is currently missing due to material model inaccuracy and should be compensated using the correction model. Additionally, we solve a local overdetermined algebraic problem to identify the missing corrective stress on each tetrahedron from the vertex control forces, taking all the frames into consideration. Then, the best fit correction model is distilled from the strain, strain rate, and stress data. Since the correction is added to both the *forward simulation* and the *space-time optimization*, it in turn improves how the forward simulation matches the example trajectory. Trajectory similarity is monitored to determine the convergence of the overall learning algorithm. As the iterations progress, the correction is gradually refined to provide better accuracy.

To enhance performance, we use sparse reduced space-time optimization (Section 5.1) as illustrated in Fig. 2(b). The *reduced basis* is extracted by performing singular value decomposition on “snapshots” of the full simulation result at the beginning of each iteration using the refined material model. Moreover, a *constrained forward simulation* (Section 5.2) is performed to provide a good starting point for the space-time optimization procedure, which quickly converges to a solution that identifies a plausible trajectory for unobserved nodes and the corresponding gentle control forces.

#### 4. Material Model

Even though there are plenty of empirical hyperelastic constitutive material models such as the nonlinear St. Venant-Kirchhoff, neo-Hookean, Ogden or Mooney-Rivlin materials, they do not account

for all deformation phenomena that may arise. Choosing a correct model to fit measurement data is already a difficult task. Our data-driven approach allows us to learn a parametric material model that can encapsulate a wide range of elastic and damping properties in a compact and unified correction function.

##### 4.1. Nominal Material Model and Assumptions

Our deformable models are constructed using linear shape functions. In order to handle large deformations of soft objects, the nominal material is described in terms of the widely adopted corotated linear FEM, formulated using principal stretches [XSZB15]. Corotated linear elasticity models combine the simplicity of the stress-deformation relationship in a linear material with just enough nonlinear characteristics to secure rotational invariance, and without suffering the non-physical zero stress configurations of St. Venant-Kirchhoff materials under extreme compression.

The deformation gradient  $F$  for each tetrahedron is diagonalized by SVD,  $F = U\hat{F}V^T$ , and the first Piola-Kirchhoff stress is computed with the principal stretches,  $\hat{P}(\hat{F}) = 2\mu(\hat{F} - I) + \zeta \text{tr}(\hat{F} - I)I$ , where  $\mu$  and  $\zeta$  are Lamé parameters. The diagonal stress is then transformed back to the world frame,  $P = U\hat{P}(\hat{F})V^T$ . An element’s contribution to its vertex forces is  $PB_m$ , where  $B_m$  is the inverse material space shape matrix (see [SB12]). Summing the contributions of all elements, we can build a large sparse matrix  $B$ , which combines the entries in  $U$ ,  $V$ , and  $B_m$ , and can be multiplied by the block vector of all element diagonal stresses  $\hat{p}$  to give a block vector of all vertex forces  $f_e$ ; that is,  $f_e = B\hat{p}$ .

We include Rayleigh damping in our nominal model, with forces computed by  $f_d = D\dot{x} = (\alpha_0 M + \alpha_1 K)\dot{x}$ , where  $D$  is the damping matrix,  $\dot{x}$  are the FEM vertex velocities,  $M$  is the lumped mass matrix, and  $K$  is the stiffness matrix assembled from per-element stiffness matrices. The nominal model parameters are assigned manually or computed using the method of Wang et al. [WWY\*15].

##### 4.2. Parametric Material Correction Model

Our parametric material correction model is designed to encapsulate a correction for the elasticity and damping inaccuracies. Instead of explicitly adjusting Young’s modulus and Poisson’s ratio

for elasticity compensation, we directly manipulate the relationship between strain and stress. Similar to the work of Irving et al. [ITF04], the stress correction is computed in a rotated frame from diagonalized strain as  $\Delta\hat{P}(\hat{F})$ , and then rotated back to current world frame by the same orthogonal matrices  $U$  and  $V$  used to diagonalize  $F$ . The per element stress corrections in the world frame are  $P_n = U\Delta\hat{P}(\hat{F})V^T$ . Using principal stretches reduces the complexity of our function approximation problem, yet it still permits complex stress corrections and strain dependent damping. Specifically, we write the strain dependent stress correction employing an RBF with  $m$  basis functions and corresponding weights  $w_k \in \mathbb{R}^3$  as

$$\Delta\hat{P}(\hat{F}) = \sum_{k=1}^m w_k \phi_k(\hat{F}). \quad (1)$$

Therefore, all vertex elastic force corrections can be written as

$$\Delta f_e = B\Delta\hat{p}, \quad (2)$$

in which  $\Delta\hat{p}$  concatenates the block vector of all element diagonal stress corrections.

We also include an element-wise damping correction through a strain dependent modification  $\Delta\alpha_1(\hat{F})$  of the Rayleigh parameter  $\alpha_1$ . Analogous to Eq. 1, the correction is also encapsulated in an RBF parametric representation as

$$\Delta\alpha_1(\hat{F}) = \sum_{k=1}^m w_k^{\alpha_1} \phi_k(\hat{F}), \quad (3)$$

with weights  $w_k^{\alpha_1} \in \mathbb{R}$ . Summing up the contributions from all elements, we can build a large sparse damping correction matrix  $\Delta D$ . The effective damping correction force is then represented as

$$\Delta f_d = \Delta D \dot{x}. \quad (4)$$

As demonstrated in Fig. 2(a), the weights  $w$  and  $w^{\alpha_1}$  are updated during each optimization iteration.

## 5. Force Correction Estimation

The general philosophy of the algorithm described in this section is to gradually learn the corrections necessary to produce forward simulations that replicate the captured desired trajectory. The purpose of using space-time optimization, in turn, is to compute a set of gentle control forces that drive the simulation to fit captured data. Consequently, information to correct our currently estimated parametric material model such that the simulation follows captured data can be distilled from the space-time optimization result.

### 5.1. Sparse Reduced Space-Time Optimization

Many variants of the space-time constraints approach of Witkin and Kass [WK88] have been proposed. To deal with the large number of degrees of freedom in our deformable models, we use reduction and sparse constraints taking inspiration from recent work [BSP09; STSH14]. We compute a reduced basis  $\Phi$  using a proper orthogonal decomposition (POD) data driven method, which performs principal component analysis (PCA) on the forward simulation trajectory with the provided initial conditions for our current approximation

of the material model. † We solve the space-time optimization as an error minimization problem with an objective function that consists of two parts: physical, and sparse trajectory misfits. In practice, we optimize with reduced coordinates  $z_i$ , where  $x_i = \Phi z_i$ . Thus, approximating the acceleration using a finite difference scheme the reduced physics errors at each time step are

$$\begin{aligned} C_{f_i} \equiv & h^{-2} \Phi^T M \Phi (z_{i-1} - 2z_i + z_{i+1}) - \Phi^T f_{ext} \\ & - \Phi^T B_{i+1} \hat{p}_{n+1} - \Phi^T D_{i+1} \Phi \dot{z}_{i+1} \\ & - \Phi^T B_{i+1} \Delta \hat{p}_{n+1} - \Phi^T \Delta D_{i+1} \Phi \dot{z}_{i+1}. \end{aligned} \quad (5)$$

This equation corresponds to our forward integration method because the force term is evaluated at the end of the time step. Next, the desired example trajectory is sparse because it comes from an incomplete scan of the surface. Letting vector  $s_i$  contain the desired point positions at time step  $i$ , we can write the sparse trajectory error at each time step as

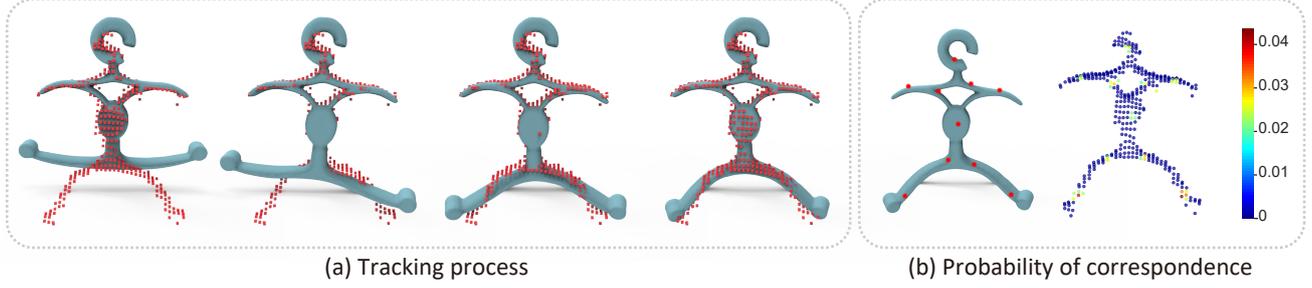
$$C_{z_i} \equiv \eta (S \Phi z_i - s_i), \quad (6)$$

where the wide sparse selection matrix  $S$  extracts the components of the desired positions by having one non-zero entry per row. The scalar  $\eta$  is used to specify the weight of position constraints given that the combination of physics and position constraints are solved in a soft manner.

Letting  $C(z)$  concatenate all physics errors  $C_f$  on top of all position errors  $C_z$ , our goal is to find a reduced trajectory  $z$  that minimizes  $\|C\|_2$ . We solve this problem (minimizing  $\frac{1}{2}\|C\|_2^2$ ) using quadratic programming, in which we need to evaluate the objective's gradient  $(\frac{\partial C}{\partial z})^T C$ . We do not assemble the Jacobian matrix  $\frac{\partial C}{\partial z}$  directly. Instead, we use the chain rule and keep it in the factored form  $\frac{\partial C}{\partial x} \frac{\partial x}{\partial z}$ , where  $\frac{\partial x}{\partial z}$  simply contains copies of the basis matrix  $\Phi$ . The matrix  $\frac{\partial C_f}{\partial x}$  has a very simple part that links vertices at different time steps through the acceleration term, and a more complex part where the chain rule must be applied to compute the force gradient. This would normally include a contribution from the parametric material correction, but generally we note better convergence when we omit it, leaving only the gradient of the nominal material. The reason is that the first order derivative of an inaccurate parametric function may introduce more noise into the system. Because we still have the parametric material correction on the right hand side, we only change the convergence and not the solution. Thus, this second part sprinkles off diagonal terms into the matrix, linking vertices that are adjacent to a common element. The matrix  $\frac{\partial C_z}{\partial x}$  simply contains copies of the selection matrix  $S$ . While the Jacobian matrix is very large, it is also very sparse. We compute the solution using the CUSP [DBOG14] library's sparse least square conjugate gradient implementation on GPU.

We check for convergence to solution  $z^*$  by monitoring our progress in reducing errors  $C(z)$ . Once converged, the physical misfits  $C_f$  provide the necessary control forces to refine our current parametric material correction model. That is, given an optimized

† An alternative would be to formulate the basis by linear combination of a much larger set of modes, and optimize the weighting matrix simultaneously [LHG\*14].



**Figure 3:** Point cloud tracking and shape similarity matching: (a) the physics-based probabilistic tracking method gradually deforming the mesh to fit the point cloud; (b) the final maximum correspondence for each point in the point cloud to a selection of 10 surface points, which shows tight localized correspondence between the point cloud and the selected points.

reduced trajectory  $\mathbf{z}$ , the gentle control force is computed as

$$\begin{aligned} \mathbf{f}_{i+1} = & h^{-2} \Phi^T \mathbf{M} \Phi (\mathbf{z}_{i-1} - 2\mathbf{z}_i + \mathbf{z}_{i+1}) - \Phi^T \mathbf{f}_{ext} \\ & - \Phi^T \mathbf{B}_{i+1} \hat{\mathbf{p}}_{n+1} - \Phi^T \mathbf{D}_{i+1} \Phi \dot{\mathbf{z}}_{i+1} \\ & - \Phi^T \mathbf{B}_{i+1} \Delta \hat{\mathbf{p}}_{n+1} - \Phi^T \Delta \mathbf{D}_{i+1} \Phi \dot{\mathbf{z}}_{i+1}. \end{aligned} \quad (7)$$

While it may be desirable to solve for control stresses at each element, as these are what is required for learning a correction, our approach permits an easier solution that directly provides a control force at each vertex.

## 5.2. Warm Start

The sparse reduced space-time optimization needs a reasonable initial guess (starting trajectory). While the forward simulation with the current parametric material correction could serve this purpose, we find it valuable to simulate a trajectory that is also constrained to follow the desired surface motion. For the forward simulation, we solve at each step the equation

$$\mathbf{A} \Delta \mathbf{v} = \mathbf{h} \mathbf{f}, \quad (8)$$

where  $\mathbf{f} = \mathbf{B} \Delta \hat{\mathbf{p}} + \Delta \mathbf{D} \dot{\mathbf{x}} + \mathbf{f}_{ext}$ . Here  $\Delta \hat{\mathbf{p}}$  is the block vector of stress corrections,  $\Delta \mathbf{D}$  is the Rayleigh damping matrix correction at the current time step,  $\mathbf{f}_{ext}$  is the external gravity force, and  $\mathbf{A} = \mathbf{M} - \mathbf{h} \mathbf{D} - \mathbf{h}^2 \mathbf{K}$ , where  $\mathbf{D}$  and  $\mathbf{K}$  are assembled using the nominal model. Many of our models are rigidly attached to the world, and we typically remove these degrees of freedom from the system. We can further divide the vertices into two groups,

$$\begin{pmatrix} \mathbf{A}_{uu} & \mathbf{A}_{uc} \\ \mathbf{A}_{cu} & \mathbf{A}_{cc} \end{pmatrix} \begin{pmatrix} \Delta \mathbf{v}_u \\ \Delta \mathbf{v}_c \end{pmatrix} = \mathbf{h} \begin{pmatrix} \mathbf{f}_u \\ \mathbf{f}_c \end{pmatrix}, \quad (9)$$

where we have treated the observable surface nodes as dynamic boundary constraints, and use subscripts  $u$  for unconstrained and  $c$  for constrained. Note next that  $\Delta \mathbf{v}_c$  at time step  $i$  can be directly computed by  $h^{-1}(\mathbf{x}_{i-1} - 2\mathbf{x}_i + \mathbf{x}_{i+1})$ , in which the positions are already given as observation constraints. Consequently, the second block row can be discarded, leaving a smaller system to solve, namely,

$$\mathbf{A}_{uu} \Delta \mathbf{v}_u = \mathbf{h} \mathbf{f}_u - \mathbf{A}_{uc} \Delta \mathbf{v}_c. \quad (10)$$

Here, the effect of dynamic boundary constraints are taken into consideration on the right hand side. We solve these large sparse systems using PARDISO [PSLG14; PSA14].

## 5.3. Space-time Optimization on Noisy Real Data

For real observations, the data can be incomplete (i.e., only partial surface scans), noisy, and in the form of unparameterized point cloud. Thus, the exact desired point positions  $\mathbf{s}_i$  at each time step  $i$  needed in Eq. 6 are no longer available. Since now each surface tetrahedral mesh node corresponds to a local point cloud patch when it is close to the surface scans, we change the node-wise position constraints in the space time optimization to instead be patch-based position constraints. We do this at every iteration for finding  $\mathbf{z}^*$  in a manner inspired by the physics-based probabilistic tracking method proposed by Schulman et al. [SLHA13] and extended by Wang et al. [WWY\*15].

For a frame consisting of  $N$  points at a given time instant (time step  $i$ ), we denote point coordinates in the point cloud by  $\mathbf{c}_n$  for  $n = 1 \dots N$ , and node positions in the tetrahedral mesh surface by  $\mathbf{s}_k$  for  $k = 1 \dots K$ . Let the probability of correspondence between the point cloud and the mesh nodes be  $p_{kn}$ . Assuming that  $\mathbf{c}_n$  is normally distributed around  $\mathbf{s}_k$  as  $\mathbf{c}_n \sim \mathcal{N}(\mathbf{s}_k, \Sigma_k)$  with an isotropic covariance matrix  $\Sigma_k = \sigma^2 \mathbf{I}$ , then we compute the probability that nodal value  $\mathbf{s}_k$  of the surface mesh corresponds to the observation  $\mathbf{c}_n$  as

$$\mathbb{P}_{kn} = \frac{1}{\sqrt{(2\pi)^3 \|\Sigma_k\|}} \exp\left(-\frac{1}{2}(\mathbf{c}_n - \mathbf{s}_k)^T \Sigma_k^{-1} (\mathbf{c}_n - \mathbf{s}_k)\right).$$

Note that parameter  $\sigma$  is chosen to be approximately the distance between nodes on the object. We only assign point  $\mathbf{c}_n$  to  $\mathbf{s}_k$  if the probability  $\mathbb{P}_{kn}$  is above a threshold. Fig. 3 shows how the mesh gradually conforms to point cloud data, along with final probabilities of correspondence between the point cloud and selected example nodes.

Now, we can reformulate the trajectory constraints of Eq. 6 as a weighed distance between each node and its corresponding patch of point clouds as

$$\mathbf{C}_{zi} \equiv \eta \sum_{k,n} \mathbb{P}_{kn} \Sigma_k^{-1} (\mathbf{c}_n - \Phi_k \mathbf{z}_i), \quad (11)$$

where  $\Phi_k$  gives the position of surface point  $k$  from reduced coordinates  $\mathbf{z}_i$ , i.e.,  $\mathbf{s}_k = \Phi_k \mathbf{z}_i$ . The Jacobian matrix required for space-time optimization of this modified position error is straightforward to compute.

During space-time optimization, the correspondence probability

between point cloud and mesh surface is updated, which allows the mesh nodes to move freely across on the point cloud surfaces. This manner of building parameterized surface trajectories effectively uses the point cloud data as a soft constraint on the nominal model, and permits reasonable estimation of surface node trajectories.

We use a very similar approach for building the initial starting point for space time optimization from real data. The tracking procedure in Fig. 3 clearly demonstrates that the mesh can track the point cloud data correctly, even in the presence of large discrepancies.

## 6. Parametric Material Correction

This section describes how we fit the parametric material correction model defined in Section 4.2 using the strain and the vertex control force trajectory derived from the space-time optimization.

We assume to always have a variety of deformations across elements and time (i.e., the principal stretches in the data we are fitting are not all the same). With k-means clustering, we select  $m$  stretches to define RBFs  $\phi_k(\hat{F}) = \|\hat{F} - \hat{F}_k\|$ , for the cluster centers  $k = 1, \dots, m$ . Then, for each time step  $i$ , we can assemble a tall matrix  $R_i$  containing the basis functions evaluated with  $\hat{F}$  for all elements, and write an equation for computing interpolated stress corrections as  $\Delta \hat{p}_i = R_i w$ , where the unknown weights are assembled here into a block vector  $w$ . When damping correction is taken into consideration, the same tall matrix  $R_i$  is used as well for Rayleigh correction interpolation weights  $w^{\alpha_1}$ .

These corrections must explain the gentle forces of Eq. 7, that is,

$$f_i = BR_i w + \left( \sum_j (R_{ij} w^{\alpha_1}) K_j \right) \dot{x}, \quad (12)$$

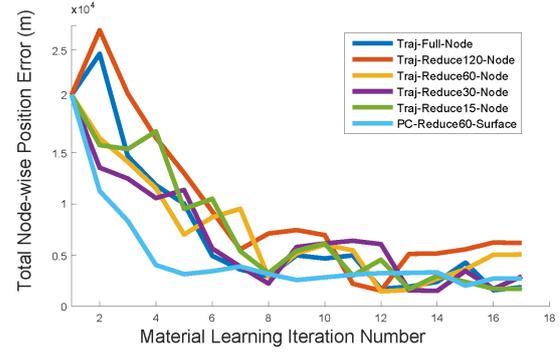
where  $B$  relates element stresses to nodal forces,  $K_j$  is the contribution of element  $j$  to the element stiffness matrix with its current strain, and  $R_{ij}$  being row  $j$  of matrix  $R_i$ . The gentle control forces may not be entirely self-consistent (i.e., an element might need different forces to correct a given state of strain at different parts of the trajectory). Therefore, we solve for  $w$  and  $w^{\alpha_1}$  simultaneously by least squares using the data from all time steps. The topology independence feature of our RBF based representation enables its transfer to other objects.

## 7. Results and Discussion

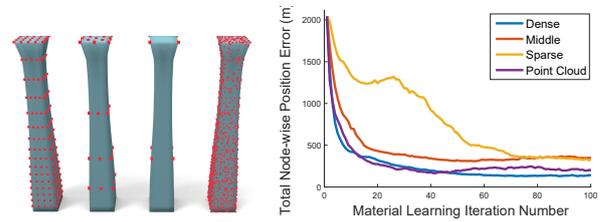
In this section, we describe experiments that help reveal what is taking place in each step of the algorithm. To validate the accuracy of our algorithm, we use both real captured data and synthetic data generated by forward simulations with known material properties.

### 7.1. Space-Time Optimization

Space-time optimization is the critical step in the entire pipeline to get training sets from pure kinematic trajectories. For this section we designed three tests using synthetic data to illustrate how our scheme can lead to the convergence of the entire algorithm. To better reflect real captured data in this evaluation, we use virtual scans as input trajectories.



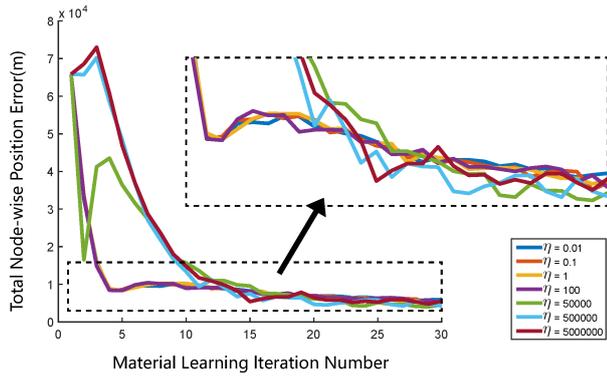
**Figure 4:** Convergence comparison for learning a material correction with different space-time optimization strategies. Input is either synthetic surface trajectories (Traj) or synthetic point cloud sequence (PC). Full and Reduce# distinguish between full and reduced space optimization, where # is the number of basis functions.



**Figure 5:** Convergence comparison under different observation conditions. A neo-Hookean material with edited compression regions is the learning target. From left to right, 100%, 13%, and 6% of the surface nodes are assumed visible. Red dots indicate the exact location of visible nodes. The rightmost bar imitates a real application where the surface information is represented as point cloud data. The error norms in nodal positions are plotted on the right with different colors. They all converge sufficiently well.

For a large scale system or a long trajectory, we must solve space-time optimization in reduced space. We test different strategies using the same synthetic example (turtle with neo-Hookean material) and compare how well they converge. The results are shown in Fig. 4. Tests using node based position constraints or point cloud patch based position constraints are distinguished by keywords ‘Node’ and ‘Surface’. From the comparison of the first five curves in Fig. 4, node-wise position constraints lead to large control forces during the first several learning iterations, when the nominal material is far from ground truth. Consequently, this introduces some overshooting. Through reduced space-time optimization, these inconveniently large force residuals are smoothly distributed throughout the entire object’s domain. We observe good convergence as seen in the last curve in Fig. 4 for learning from point cloud trajectories.

Our algorithm can handle noisy and sparse observations. We tested it on the same synthetic examples with different observation conditions. The learning target is a neo-Hookean material with edited compression regions. Notice that in Fig. 5, the number of observation points in the first three cases drops sharply. For all these



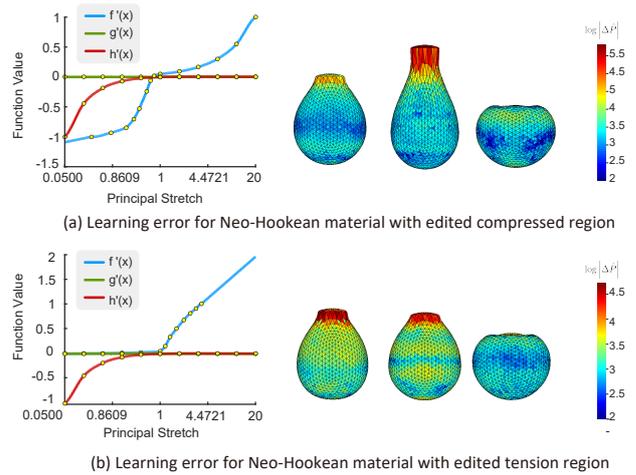
**Figure 6:** Convergence comparison with different position constraint enforcement weights. The learning target is a neo-Hookean material with edited compression regions. Position error norms at each iteration are plotted for seven different  $\lambda$  values in Eq. 6. Strong position constraints introduce vibration at early iterations, while soft ones mildly sacrifice accuracy.

experiments, our algorithm can converge to a sufficiently accurate solution. We also used virtual scans to imitate a real capture situation, where point cloud data cover the whole surface. Our patch based position constraint adopted in the last case (Section 5.3) performs better than the sparse observation case and only mildly worse than the accurate full surface observation case.

As described in Eq. 6,  $\lambda$  controls the enforcement weight of position constraints, which influences the control force. We chose different  $\lambda$  values in the wide range [0.01, 500000], and simulated the same example (turtle with neo-Hookean material as learning target, all surface nodes assumed visible). From Fig. 6, we observe that larger  $\lambda$  values introduce some vibration or even overshooting at the early optimization stages. The unbalanced ratio between force residual constraint and position constraint causes a large force deviation in the internal points, which consequently decreases the quality of generated training data. In contrast, the curves corresponding to smaller  $\lambda$  are much smoother along the optimization iteration axis. From a global perspective, different  $\lambda$  will not necessarily produce significantly different final results, even though there are still subtle accuracy drops for smaller  $\lambda$  cases as illustrated in the zoom-in of Fig. 6. We expect it should be possible to adaptively tune  $\lambda$  between iterations to speed up convergence. For real captured data,  $\lambda$  is chosen to be small, depending on the confidence in the observations.

## 7.2. Nonlinear Constitutive Material Modeling

To validate the generality of our parametric material model estimation algorithm, we test its ability to learn a variety of different materials using a corotational model for the nominal material. Ground truth trajectories are either generated in the VEGAFem [BSS12] library using classical hyperelastic material model, or they are user defined elasticity and damping models, or captured by Kinect sensors. Table 1 shows the statistics of all our testing cases. Qualitative results for these cases can be seen in the supplementary video. Each case is discussed below, while reconstruction errors are listed in the last column of Table 1.



**Figure 7:** Visualization of the learning result accuracy for user designed elastic material examples [XSZB15]. Plots (left) show edited compression and tension regions of a Neo-Hookean material. Images (right) use color to show principal stress error distribution of our learned material correction to a corotational model.

### 7.2.1. Classical Hyperelastic Material

In Fig. 1, the turtle is made of neo-Hookean material, and the dragon is made of StVK material. We use two deliberately designed test trajectories to validate our learning result. The first test has a similar deformation scale as the training trajectory, while the second test has a much larger range of deformation. Table 1 and the supplementary video show that the learning result can reproduce similar deformation with high accuracy; the results for different deformations also demonstrate low error. Vibration differences can only be observed towards the end of a sequence, and as such, can largely be attributed to error accumulation.

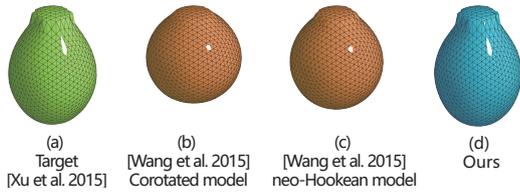
### 7.2.2. User Designed Elasticity and Damping Model

Our algorithm can also be extended to model user designed nonlinear elasticity and damping material models. Our third example is a soft solid sphere whose top part is keyframed in an up-down motion (Fig. 7). Following the method of Xu et al. [XSZB15], the internal elastic forces and tangent stiffness matrices are formulated in a polynomial space of principal stretch. Customized materials are designed by editing a single stress-strain curve using a spline interface. We tested two different designed nonlinear materials which edit tension and compression starting from the a neo-Hookean material as shown in Fig. 7. The supplementary video shows indistinguishable simulation trajectories, while Fig. 7 shows small stress reconstruction errors produced by our parametric material correction of a nominal corotational model. We also compared with the parameter fitting based method of Wang et al. [WWY\*15] and a modification of the method which replaces the simple corotated elasticity model with a neo-Hookean model. The reported side by side comparison in Fig. 8 and the supplementary video both clearly demonstrate that our method is superior to this parameter fitting algorithm, especially when the default model is simple.

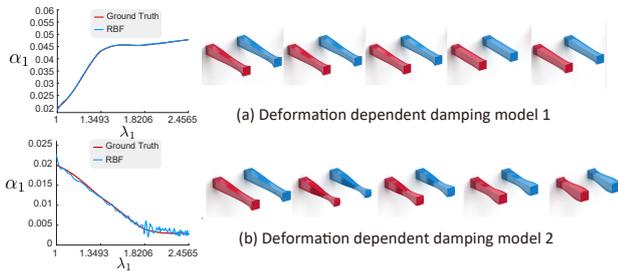
Although linear viscous damping is widely used in the computer

**Table 1:** Parameters and errors for different test cases. For each test object we list ground truth (G) and nominal (N) parameters for Young's modulus  $E$  in MPa, Poisson ratio  $\nu$ , Rayleigh damping  $\alpha_0$  and  $\alpha_1$ , and use U and R to denote user designed and real materials, respectively. The maximum position errors are measured using percentage of object size in simulations using a time step of 0.001 seconds.

Case	Material, Ground Truth (G)	$E_G$	$\nu_G$	$\alpha_{0G}$	$\alpha_{1G}$	Material (N)	$E_N$	$\nu_N$	$\alpha_{0N}$	$\alpha_{1N}$	$err_L$
Turtle	neo-Hookean	2e4	0.43	0.0	0.0	Corotation	3.5e4	0.43	0.0	0.0	2.8
Dragon	StVK	1e5	0.40	0.0	0.0	Corotation	1.2e5	0.40	0.0	0.0	0.4
Sphere1	neo-Hookean(tension)	U	U	0	0.2	Corotation	1.2e5	0.43	0.0	0.2	4.1
Sphere2	neo-Hookean (compression)	U	U	0	0.2	Corotation	1.2e5	0.43	0.0	0.2	6.0
Bar (Damping1)	Corotation + strain-dep damping	4e4	0.43	U	0.2	Corotation	4e4	0.43	0.0	0.2	0.1
Bar (Damping2)	Corotation + strain-dep damping	4e4	0.43	U	0.2	Corotation	4e4	0.43	0.0	0.2	0.8
Pot Holder	Real Material	R	R	R	R	Corotation	2.4e6	0.43	0.0	0.0	3.0/7.0
Hanger	Real Material	R	R	R	R	Corotation	4.5e5	0.43	0.001	1.0	4.0
Silicon Bar	Real Material	R	R	R	R	Corotation	3.0e5	0.45	0	0.0	6.0
Bar (Heterogeneous)	Corotation	1e5/1e7	0.40	0.0	0.0	Corotation	3e6	0.40	0.0	0.0	0.2/0.5



**Figure 8:** Comparing material learning techniques where (a) the target trajectory is generated using neo-Hookean material with modified compression region [XSZB15]; (b) [WWY\*15]; (c) [WWY\*15], replacing the corotated model with the neo-Hookean model; and (d) our algorithm.



**Figure 9:** Visualization of the learning result accuracy for user designed deformation dependent damping models. Red is used to show ground truth, while blue shows simulated poses with a learned parametric material correction.

graphics community, this only constitutes a small subset of all viscous damping models. Under many circumstances, the damping matrix  $C$  can depend nonlinearly on both deformation and velocity. To validate the accommodation of our material model estimation algorithm for damping compensation, Fig. 9 shows our tests on a strain-dependent damping model. We start from a Rayleigh damping model, and substitute the original constant stiffness damping coefficient  $\alpha_1$  with a polynomial function of the first principal stretch  $\lambda_1$ . The function of deformation dependent  $\alpha_1$  is controlled using a spline tool.

Since our parametric material correction model is independent

of topology, it can be easily transferred to other simulation scenarios. The neo-Hookean material with modified tension region model which we learned from a ball example can be transferred, for instance, to a chubby bunny. As can be gleaned from the video and Fig. 10 (top), the bunny belly vibrates in a lively fashion during jumps. From a side by side comparison with the ground truth result in the video, the simulated trajectories match well initially but then drift apart due to error accumulation. This inaccuracy could be due to the learned material itself and/or the mismatch in the geometric model. We also transfer the two deformation dependent damping models of Fig. 9 to different leaves of a taro plant to assign separate properties for young and old leaves as shown in the video and Fig. 10 (bottom).

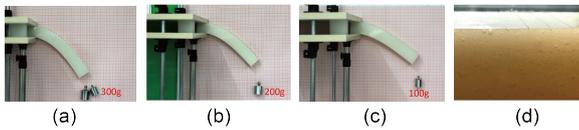
### 7.2.3. Real Material Fitting

We first validate our algorithm on a silicon bar example. The object is made by casting an elastomer material of type Silicon-601 into a bar-shaped mold. The target material properties depend on the amount of added curing agent, which has no default nominal value. Moreover, the numerous small air bubbles seen in Fig. 11(d) add more variance to the material properties. Thus, in the presence of these bubbles, it is worthwhile to allow an algorithm like ours to infer the material properties. As illustrated in Fig. 11(a) through (c), the object is fixed at one end, and three different external loads (100 g, 200 g, and 300 g) are added at the free end. We released the load and recorded its vibration. The trajectory of the 300 g case is used as training data, while the other two are used for testing. We learn the material using both our method and the method of Wang et al. [WWY\*15]. We encourage the reader to watch the side by side comparison in the accompanying video.

We also validated and compared our algorithm with Wang et al. [WWY\*15] for the real silicon pot holder and hanger examples which are shown in Figs. 12 and 13, respectively. The captured trajectories seen in (a) of both figures are used to obtain material corrections. The raw point cloud data are fused, and severe outliers are removed using Artec Studio. The results are validated through external loading tests. More specifically, we fix the objects at one end, either horizontally or vertically, and attach different weights at the other end. The external weights are suddenly released and the vibrations of the soft object are simulated and compared with the ground truth. A side by side comparison can be seen in the accompanying video. For the hanger and silicon bar example, we ob-



**Figure 10:** Material transfer. Top: A chubby jumping bunny uses the learned material of the ball in Fig 7(b). Bottom: A plant responds to user interactions, where the left and middle leaves use learned damping properties of the bars in Fig. 9.



**Figure 11:** Silicon bar examples: (a) 300 g external loading case is used as training data, (b) 200 g external loading case, and (c) 100 g external loading case are used as test data; (d) irregular bubbles can be observed in a magnified view.

**Table 2:** Performance statistics measured for different testing cases. From left to right: the test object, number of vertices, number of tet elements, number of frames for training data, number of reduced modes, number of learning iterations, number of RBF kernels, and total computation time in hours for material learning.

Case	#vert	#tet	#frm	#mode	#iter	#kernel	CPU
Turtle	347	1185	400	200	29	46	0.9
Dragon	959	2590	400	200	19	140	1.1
Sphere 1	2655	12712	400	200	27	140	9.1
Sphere 2	2655	12712	400	200	23	140	7.7
Bar (Damp1)	425	1536	300	150	7	400	0.4
Bar (Damp2)	425	1536	800	400	18	500	2.5
Pot Holder	3031	8843	400	200	10	140	4.2
Hanger	1740	5888	375	200	4	500	2.0
Silicon Bar	650	2400	600	full	6	300	0.5

served that the original constant mass damping coefficient  $\alpha_0$  must be substituted with a polynomial function of principal stretch  $\lambda$ .

### 7.3. Material Coarsening

The algorithm proposed in this paper can also be used for material coarsening. In Fig. 14, a high resolution bar ( $8 \times 8 \times 34$ ) is composed of two different constitutive materials, with Young modulus values of  $1e5$  and  $1e7$ , respectively. The two materials are composited in a layer by layer manner, represented by the light and dark green colors. The low resolution mesh is the result of coarsening by factor 2 along three axis directions. Two principal deformation

modes (bend and twist) are used as training data. The equivalent coarsened material property found by our algorithm can produce very similar motion to the original high resolution heterogeneous model.

### 7.4. Performance

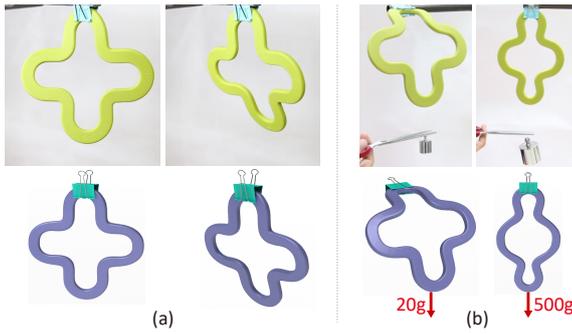
We measured the computational cost for each critical step on a 10-core 3.0 GHz Intel i7-6950X desktop. The performance for space-time optimization, listed in Table 2, correlates with the number of tetrahedral elements, the number of frames in the motion trajectory, the number of RBF kernels, and the dimension of selected reduced basis.

### 8. Conclusion and Future Work

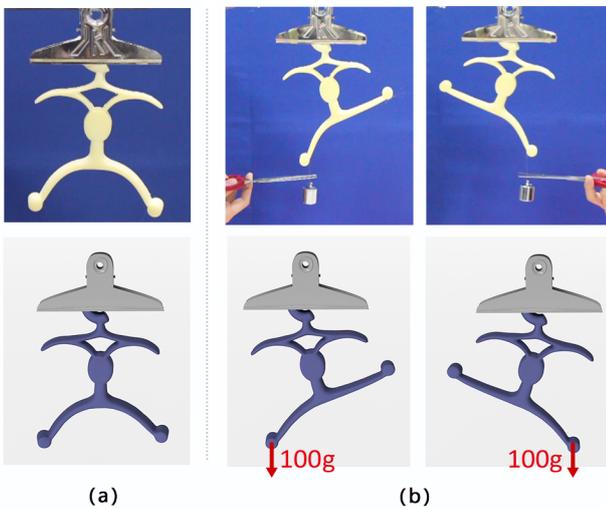
We have presented a new method for estimating nonlinear constitutive models from trajectories of surface data. The key insight is to have a parametric material correction model learn the error of the elastic and damping properties of a nominal material. A framework for gradually learning this correction from only kinematic data is described. We have demonstrated our method with several examples, illustrating the ability of our approach to learn classical material models, user designed materials (cartoon physics), and real world captured data.

The desire to work with realistic constitutive models when simulating complex motion has been shared for a long while by researchers from many fields, not just computer graphics. The possibility of employing machine learning towards such a goal is tantalizing. We solved an interesting and timely problem which is critical for all machine learning algorithms for generating annotated data automatically. We believe our present work is an important step in that direction.

There are a number of interesting ideas to explore in future research. First, we note that extending our approach to accommodate a variety of numerical integration techniques would help in avoiding or reducing step size dependent numerical damping effects in our results. Second, we only address heterogeneous materials in the case of numerical coarsening to a homogeneous material. There are



**Figure 12: Real Material Fitting:** (a) Two captured trajectories and the corresponding tracking result. (b) Static loading tests: a silicon pot holder is bent and pulled by external weights; the holder is fixed at one end horizontally and vertically.

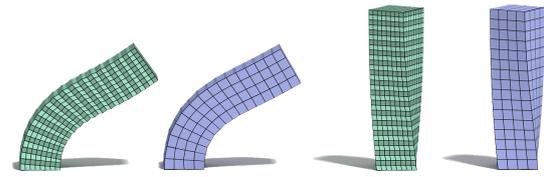


**Figure 13: Real Material Fitting:** (a) One captured trajectory and the corresponding tracking result. (b) Static loading tests: a silicon hanger is pulled by external weights; the hanger is fixed at top.

interesting extensions that can be considered for dealing with varying material properties across a model, for instance, by adding a latent material parameter to our representation. Also, note that our framework is compatible with many other parametric representations. For instance, a spline-based representation [XSZB15] could be another promising option to explore. Finally, there are still a variety of potential damping effects that we cannot capture with our approach. The models we estimate do not account for any hysteresis in the damping model, while this can be common in real materials, as can also be the presence of plastic deformation. Capturing a larger variety of complex plastic and damping behaviors is indeed a very interesting avenue for future work.

### Acknowledgments

We sincerely thank the reviewers for their constructive comments. This work was supported in part by National Key R&D Program (2019YFF0302902, 2018YFB1403900), NSFC (61861130365,



**Figure 14: Material coarsening.** The green bar shows the fine mesh with a layered material distribution; the purple bar is the corresponding coarsened mesh with homogeneous material distribution. Bend (left) and twist (right) deformation trajectories of fine mesh are used as training data, and the purple bars are the reconstruction results after learning.

61761146002), GD Talent Program (2019JC05X328), GD Science and Technology Program (2020A0505100064, 2018KZDXM058), and NSERC Canada Discovery Grants (2018-05665, 84306).

### References

[BBO\*09] BICKEL, BERND, BÄCHER, MORITZ, OTADUY, MIGUEL A., et al. “Capture and modeling of non-linear heterogeneous soft tissue”. *ACM Trans. on Graphics* 28.3 (2009), 89:1–89:9 2.

[BSP09] BARBIČ, JERNEJ, da SILVA, MARCO, and POPOVIĆ, JOVAN. “Deformable Object Animation Using Reduced Optimal Control”. *ACM Trans. Graph.* 28.3 (July 2009), 53:1–53:9 4.

[BSS12] BARBIČ, JERNEJ, SIN, FUN SHING, and SCHROEDER, DANIEL. *Vega FEM Library*. <http://www.jernejbarbic.com/vega>. 2012 7.

[BT07] BECKER, MARKUS and TESCHNER, MATTHIAS. “Robust and Efficient Estimation of Elasticity Parameters using the linear Finite Element Method”. *Proc. Simulation und Visualization*. 2007, 15–28 2.

[BTH\*03] BHAT, KIRAN S., TWIGG, CHRISTOPHER D., HODGINS, JESSICA K., et al. “Estimating Cloth Simulation Parameters from Video”. *Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation*. 2003, 37–51 2.

[CLK\*19] CHEN, YU JU (EDWIN), LEVIN, DAVID I. W., KAUFMAN, DANNY M., et al. “EigenFit for Consistent Elastodynamic Simulation Across Mesh Resolution”. *Proceedings SCA* (2019). [doi.org/10.1145/3309486.3340248](https://doi.org/10.1145/3309486.3340248) 2.

[CLMK17] CHEN, DESAI, LEVIN, DAVID I. W., MATUSIK, WOJCIECH, and KAUFMAN, DANNY M. “Dynamics-aware Numerical Coarsening for Fabrication Design”. *ACM Trans. Graph.* 36.4 (July 2017), 84:1–84:15 2.

[CLSM15] CHEN, DESAI, LEVIN, DAVID I. W., SUEDA, SHINJIRO, and MATUSIK, WOJCIECH. “Data-driven Finite Elements for Geometry and Material Design”. *ACM Trans. Graph.* 34.4 (July 2015), 74:1–74:10 2.

[DBC\*17] DAVIS, ABE, BOUMAN, KATHERINE L., CHEN, JUSTIN G., et al. “Visual Vibrometry: Estimating Material Properties from Small Motions in Video”. *IEEE Trans. Pattern Analysis & Machine Intelligence* 39.4 (2017), 732–745 2.

[D BOG14] DALTON, STEVEN, BELL, NATHAN, OLSON, LUKE, and GARLAND, MICHAEL. *Cusp: Generic Parallel Algorithms for Sparse Matrix and Graph Computations*. Version 0.5.0. 2014 4.

[FMD\*19] FULTON, LAWSON, MODI, VISMAY, DUVENAUD, DAVID, et al. “Latent-space Dynamics for Reduced Deformable Simulation”. *Computer Graphics Forum* (2019) 2.

[GGJW91] GHABOUSSI, J, GARRETT JR, JH, and WU, XIPING. “Knowledge-based modeling of material behavior with neural networks”. *Journal of Engineering Mechanics* 117.1 (1991), 132–153 2.

- [GPZHA98] GHABOUSSI, JAMSHID, PECKNOLD, DAVID A., ZHANG, MINGFU, and HAJ-ALI, RAMI M. “Autoprogressive training of neural network constitutive models”. *International Journal for Numerical Methods in Engineering* 42.1 (1998), 105–126 2.
- [HBBC19] HAHN, DAVID, BAZET, POL, BERN, JAMES M., and COROS, STELIAN. “Real2Sim: Visco-elastic parameter estimation from dynamic motion”. *ACM Trans. on Graphics* (2019) 2.
- [ITF04] IRVING, G., TERAN, J., and FEDKIW, R. “Invertible Finite Elements for Robust Simulation of Large Deformation”. *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '04. Grenoble, France: Eurographics Association, 2004, 131–140. ISBN: 3-905673-14-2 4.
- [JG06] JUNG, SUNGMOON and GHABOUSSI, JAMSHID. “Neural network constitutive model for rate-dependent materials”. *Computers & Structures* 84.15 (2006), 955–963 2.
- [KMOD09] KHAREVYCH, LILY, MULLEN, PATRICK, OWHADI, HOUMAN, and DESBRUN, MATHIEU. “Numerical Coarsening of Inhomogeneous Elastic Materials”. *ACM Trans. Graph.* 28.3 (July 2009), 51:1–51:8 2.
- [LHG\*14] LI, SIWANG, HUANG, JIN, de GOES, FERNANDO, et al. “Space-time Editing of Elastic Motion Through Material Optimization and Reduction”. *ACM Trans. on Graphics* 33.4 (2014), 108:1–108:10 4.
- [LSW\*18] LUO, RAN, SHAO, TIANJIA, WANG, HUAMIN, et al. “NNWarp: Neural Network-based Nonlinear Deformation”. *IEEE Trans. Visualization & Computer Graphics* (2018) 2.
- [MBT\*12] MIGUEL, E., BRADLEY, D., THOMASZEWSKI, B., et al. “Data-Driven Estimation of Cloth Simulation Models”. *Computer Graphics Forum* 31.2 (2012), 519–528 2.
- [MMO16] MIGUEL, EDER, MIRAUT, DAVID, and OTADUY, MIGUEL A. “Modeling and Estimation of Energy-Based Hyperelastic Objects”. *Computer Graphics Forum* 35.2 (2016), 385–396 2.
- [MSBAO18] M. SÁNCHEZ-BANDERAS, ROSA and A. OTADUY, MIGUEL. “Strain Rate Dissipation for Elastic Deformations”. *Computer Graphics Forum* 37.8 (2018), 161–170 2.
- [MTB\*13] MIGUEL, EDER, TAMSTORF, RASMUS, BRADLEY, DEREK, et al. “Modeling and Estimation of Internal Friction in Cloth”. *ACM Trans. Graph.* 32.6 (Nov. 2013), 212:1–212:10 2.
- [MTGG11] MARTIN, SEBASTIAN, THOMASZEWSKI, BERNHARD, GRINSPUN, EITAN, and GROSS, MARKUS. “Example-based Elastic Materials”. *ACM Trans. Graph.* 30.4 (2011), 72:1–72:8 2.
- [NKJF09] NESME, MATHIEU, KRY, PAUL G., JEŘÁBKOVÁ, LENKA, and FAURE, FRANÇOIS. “Preserving Topology and Elasticity for Embedded Deformable Models”. *ACM SIGGRAPH 2009 Papers*. SIGGRAPH '09. New Orleans, Louisiana: ACM, 2009, 52:1–52:9. ISBN: 978-1-60558-726-4 2.
- [PDJ\*01] PAI, DINESH K., DOEL, KEES VAN DEN, JAMES, DOUG L., et al. “Scanning Physical Interaction Behavior of 3D Objects”. *Proc. of SIGGRAPH*. 2001, 87–96 2.
- [PSA14] PETRA, COSMIN G., SCHENK, OLAF, and ANITESCU, MIHAI. “Real-time stochastic optimization of complex energy systems on high-performance computers”. *IEEE Computing in Science & Engineering* 16.5 (2014), 32–42 5.
- [PSLG14] PETRA, COSMIN G., SCHENK, OLAF, LUBIN, MILES, and GÄRTNER, KLAUS. “An augmented incomplete factorization approach for computing the Schur complement in stochastic optimization”. *SIAM Journal on Scientific Computing* 36.2 (2014), C139–C162 5.
- [PZM\*15] PANETTA, JULIAN, ZHOU, QINGNAN, MALOMO, LUIGI, et al. “Elastic Textures for Additive Fabrication”. *ACM Trans. Graph.* 34.4 (July 2015), 135:1–135:12 2.
- [SB12] SIFAKIS, EFTYCHIOS and BARBIC, JERNEJ. “FEM Simulation of 3D Deformable Solids: A Practitioner’s Guide to Theory, Discretization and Model Reduction”. *ACM SIGGRAPH 2012 Courses*. SIGGRAPH '12. Los Angeles, California, 2012, 20:1–20:50. ISBN: 978-1-4503-1678-1 3.
- [SG15] STEFANOS, DRAKOS and GYAN, PANDE. “On Neural Network Constitutive Models for Geomaterials”. *Journal of Civil Engineering Research* 5.5 (2015), 106–113 2.
- [SLHA13] SCHULMAN, JOHN, LEE, ALEX, HO, JONATHAN, and ABBEEL, PIETER. “Tracking Deformable Objects with Point Clouds”. *Proc. IEEE Int. Conf. on Robotics & Automation*. 2013 5.
- [SLS04] SCHONER, JEFFREY, LANG, JOCHEN, and SEIDEL, HANS-PETER. “Measurement-Based Interactive Simulation of Viscoelastic Solids”. *Computer Graphics Forum* 23.3 (2004), 547–556 2.
- [STC\*12] SCHUMACHER, CHRISTIAN, THOMASZEWSKI, BERNHARD, COROS, STELIAN, et al. “Efficient Simulation of Example-based Materials”. *Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation*. 2012 2.
- [TSH14] SCHULZ, CHRISTIAN, von TYCOWICZ, CHRISTOPH, SEIDEL, HANS-PETER, and HILDEBRANDT, KLAUS. “Animating Deformable Objects Using Sparse Spacetime Constraints”. *ACM Trans. Graph.* 33.4 (July 2014), 109:1–109:10 4.
- [TREO16] TORRES, ROSELL, RODRÍGUEZ, ALEJANDRO, ESPADERO, JOSÉ M., and OTADUY, MIGUEL A. “High-resolution Interaction with Corotational Coarsening Models”. *ACM Trans. Graph.* 35.6 (Nov. 2016), 211:1–211:11 2.
- [WK88] WITKIN, ANDREW and KASS, MICHAEL. “Spacetime Constraints”. *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '88. New York, NY, USA: ACM, 1988, 159–168. ISBN: 0-89791-275-6 4.
- [WOR11] WANG, HUAMIN, O'BRIEN, JAMES F., and RAMAMOORTHY, RAVI. “Data-driven Elastic Models for Cloth: Modeling and Measurement”. *ACM Trans. on Graphics* 30.4 (2011), 71:1–71:12 2.
- [WY\*15] WANG, BIN, WU, LONGHUA, YIN, KANGKANG, et al. “Deformation Capture and Modeling of Soft Objects”. *ACM Trans. Graph.* 34.4 (July 2015), 94:1–94:12 2, 3, 5, 7, 8.
- [XB17] XU, HONGYI and BARBIČ, JERNEJ. “Example-based Damping Design”. *ACM Trans. Graph.* 36.4 (July 2017), 53:1–53:14 2.
- [XSZB15] XU, HONGYI, SIN, FUNSHING, ZHU, YUFENG, and BARBIČ, JERNEJ. “Nonlinear Material Design Using Principal Stretches”. *ACM Trans. Graph.* 34.4 (July 2015), 75:1–75:11 2, 3, 7, 8, 10.
- [YLL17] YANG, SHAN, LIANG, JUNBANG, and LIN, MING C. “Learning-Based Cloth Material Recovery From Video”. *The IEEE International Conference on Computer Vision (ICCV)*. 2017 2.