Learning Contact Deformations with General Collider Descriptors

Cristian Romero
Universidad Rey Juan Carlos
Madrid, Spain
crisrom002@gmail.com

Dan Casas
Universidad Rey Juan Carlos
Madrid, Spain
dan.casas@urjc.es

Maurizio M. Chiaramonte
Meta Reality Labs Research
Redmond, USA
mchiaram@meta.com

Miguel A. Otaduy
Meta Reality Labs Research
Madrid, Spain
otaduy@meta.com

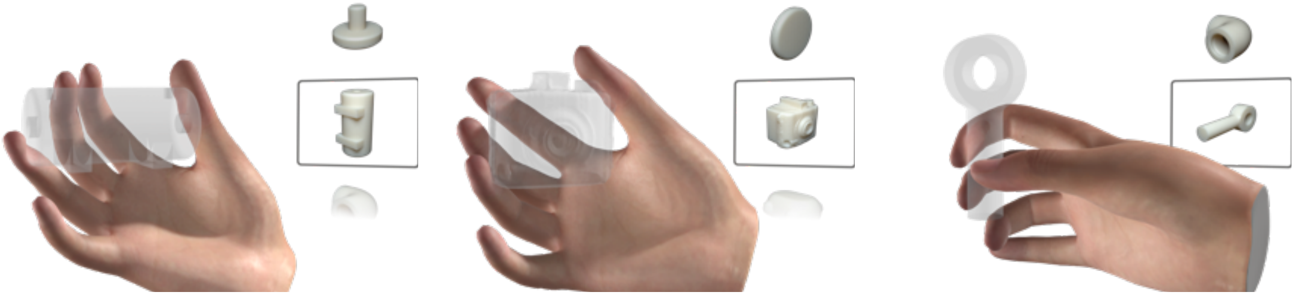


Figure 1: Interactive tactile exploration of diverse objects. We compute detailed contact deformations in real time using a learning-based model that generalizes to collider shapes. None of the colliders shown in the images were used for training.

ABSTRACT

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

CCS CONCEPTS

• **Computing methodologies → Physical simulation**.

KEYWORDS

Contact, learning

ACM Reference Format:
Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. 2023. Learning Contact Deformations with General Collider Descriptors. In *SIGGRAPH Asia 2023 Conference Papers (SA Conference Papers '23), December 12–15, 2023, Sydney, NSW, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3610548.3618229

1 INTRODUCTION

The simulation of contact deformations remains an open challenge in computer graphics and beyond. Despite the continuous development of algorithms for faster, more accurate, and/or more robust contact deformations [Brandt et al. 2018; Harmon and Zorin 2013; Li et al. 2020; Smith et al. 2018], the targets of resolution and performance grow unstopably, and call for even faster and more accurate methods. This is particularly relevant as we envision real-time applications where we interact with objects that appear and feel real.

In recent years, machine learning has been explored as an approach to accelerate the runtime computation of rich deformations [Fulton et al. 2019; Holden et al. 2019; Pfaff et al. 2021; Romero et al. 2021], via exhaustive precomputation of the parameters of neural models. Few of these learning-based methods address the computation of deformations induced by contact, and those that do make many limiting assumptions about the nature of the colliding objects. For instance, to the best of our knowledge, learning-based

SA Conference Papers '23, December 12–15, 2023, Sydney, NSW, Australia

ACM ISBN 979-8-4007-0315-7/23/12…$15.00
https://doi.org/10.1145/3610548.3618229

models for rich contact deformation are trained on specific pairs of objects, and therefore do not scale to the combinatorial complexity of object-object interactions.

In this work, we lift a crucial limitation of learning-based contact deformation models, and we present the first model that generalizes the collider geometry, i.e., it is not trained on a specific collider. Our learning-based deformation model augments a reduced deformable model (which is fast but lacks detail) with rich contact deformations induced by any rigid collider shape. Our key observation is that contact deformations not captured by the reduced model are local, hence we train the learning-based model on local shape descriptors of colliders, not on the full collider shape. In Section 3 we describe the *collider descriptor* in detail, as well as the formulation and training of the neural contact deformation model. Our learning-based deformation model is trained on a library of colliders, but supports general colliders at runtime. In Section 4 we discuss efficient runtime evaluation of the collider descriptor.

Our proposed collider descriptor can be connected to shape descriptors. However, while shape descriptors are typically used for labeling or classification problems [Qi et al. 2017a; Wang et al. 2019], we use our collider descriptor as a conditional signal for a neural deformation field. We find that, for our application of contact deformation, it is convenient to design a descriptor that encodes together local geometry and the relative configuration with respect to this local geometry. Moreover, in the design of the collider descriptor, we have addressed challenges concerning spatial smoothness and rotational invariance.

In our examples, we have integrated the learning-based contact deformation model in interactive dynamic simulations. We show that our model can reach the accuracy of previous works trained on specific colliders, while our model supports general collider shapes and therefore infinitely more diverse runtime contact scenarios, as shown in Figure 1. Moreover, we showcase interesting applications beyond visual animation of deformation detail, such as exploration and manipulation of products for e.g. online retail, or augmentation of contact information suitable for high-fidelity haptics.

2 RELATED WORK

2.1 Learning-Based Deformation and Contact

In recent years, learning methods have been applied to many different types of deformation models. Often, they are used for synthesizing rich deformation detail conditioned by some low-dimensional code, e.g., linear deformation driving nonlinear deformation [Luo et al. 2020], coarse deformation driving numerical coarsening (i.e., high-order shape functions) [Ni et al. 2023], upscaling of dynamics to objects of different topology [Zheng et al. 2021], skeletal motion driving nonlinear soft-tissue deformation [Bailey et al. 2018] or soft-tissue dynamics [Casas and Otaduy 2018; Santesteban et al. 2020], skeletal motion driving cloth deformation [Bertiche et al. 2022; Santesteban et al. 2022a], or upscaling of low-resolution hair simulation [Lyu et al. 2022]. These problems are somewhat similar to ours, as the high-dimensional deformation is conditioned by a low-dimensional code; however the input, output and constraints differ strongly.

In the case of cloth deformation, particular attention has been placed on solving contact, with self-supervised learning of a latent space free of body collisions [Santesteban et al. 2021], by adding a repulsive force unit to the network architecture [Tan et al. 2022b], or untangling multiple cloth layers through projection operations on implicit representations [Santesteban et al. 2022b]. Learning has also been used for resolving self-collisions, e.g., by computing differentiable collision classifiers as a function of mesh deformation [Tan et al. 2022a], or configuration-space distances conditioned by the deformation state of reduced models [Cai et al. 2022]. In our work, the focus is on computing detailed contact deformation between pairs of objects, and we approach it through problem-specific choice of the input and output of the neural model, to maximize learning ability.

We use a learning approach to augment a reduced deformation model. We choose explicitly the reduced kinematic representation, but other works use learning to represent the kinematics in a latent space [Fulton et al. 2019], with constraints to enforce conservation laws [Lee and Carlberg 2021], with learning-based cubature [Shen et al. 2021], or even with a neural continuous representation of the reduced model, completely eliminating discretization of the kinematics [Chen et al. 2023]. Very recent work has also placed the attention on training the reduced representation without example data [Sharp et al. 2023]. Continuous reduced models are even used for modeling muscle activation of soft characters [Yang et al. 2022].

Closest to our work are methods that learn contact response in deformable object simulation. Holden et al. [2019] learned the dynamic update of a reduced simulation upon contact. We aim to learn only contact deformation and solve the reduced state update through state-of-the-art numerical integration. As a result, our method is slower, but it models considerably finer deformation detail. Similar to us, Romero et al. [2021] also solved detailed contact deformations on top of a reduced simulation. In their follow-up work [Romero et al. 2022], they vastly improved learning ability and supported more complex deformations by reformulating the learning problem in a contact-centric manner. Aigerman et al. [2022] applied to the same problem a more general methodology for learning Jacobian fields, and while the approach worked, it suffered a performance penalty. All these works learn collider-specific models, which do not scale to the combinatorial complexity of object interactions. Instead, we model a problem of much higher dimensionality, as we consider general collider geometries as part of the dynamic input to the problem.

2.2 Shape Descriptors in Neural Models

The key component of our work that enables generalization of the collider shape is the collider descriptor. As mentioned in the introduction, this collider descriptor can be related to shape descriptors.

Neural representations for unstructured point clouds, starting with PointNet [Qi et al. 2017a], have opened the possibility to encode rich latent shape information. In a nutshell, they aggregate individual point features into a global signature. PointNet++ [Qi et al. 2017b] extends the original PointNet with a hierarchical structure, allowing the representation of local features at different scales. PCPNet [Guerrero et al. 2018] learns local features robust to common point imperfections (e.g., varying noise level, sampling density, level of detail, missing data), by enforcing perturbations in the

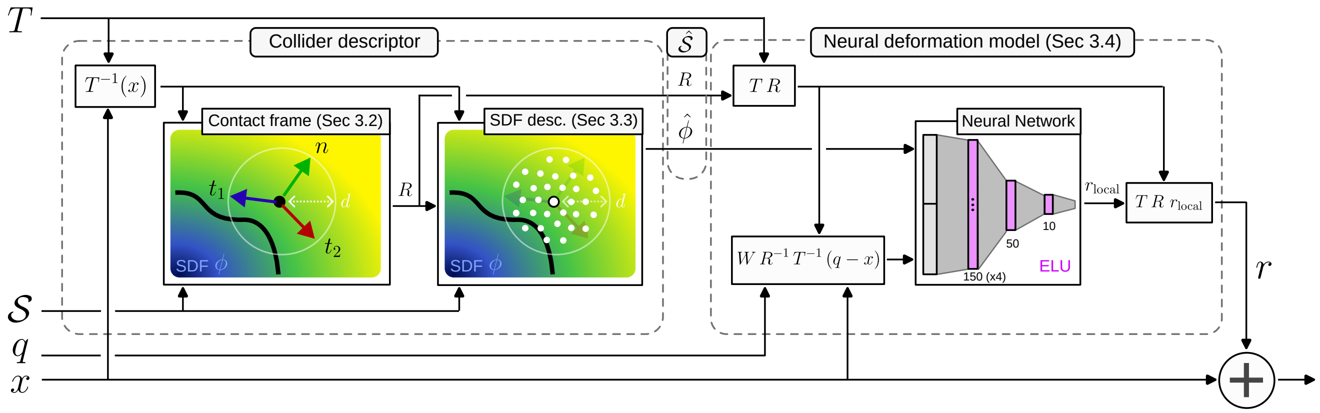


Figure 2: Summary of the computational pipeline of our contact deformation model. The pipeline has two parts: (left) the evaluation of a local collider descriptor, and (right) the evaluation of the neural deformation. The collider descriptor takes as runtime input the shape of the collider S (characterized by its SDF ϕ), the rigid transformation of the collider T, the deformed state of the colliding object q, and the deformed point of interest x. The evaluation of the collider descriptor includes two parts: a contact frame R and a local SDF descriptor $\hat{\phi}$. Then, the deformed point of interest x is transformed to the local reference of the contact frame, it is weighted by sparsifying weights W, and the result is input to the neural deformation model together with the local SDF descriptor. As a result, we obtain a local contact deformation r_{local} that is then transformed back to world space and added to x.

training data. Point2Sequence [Liu et al. 2019] aggregates the information of different local regions thanks to an attention mechanism. EdgeConv [Wang et al. 2019] arranges dynamic graphs on point clouds to enable more powerful operations. While such shape descriptors may encode some of the information relevant for contact deformations, they are mostly used for problems such as object detection, shape classification, or part segmentation. Contact requires information of shape, together with relative configuration, to act as conditional signal for deformation.

The magnitude of contact deformations is inherently invariant to rigid transformations of the colliding objects. However, since deformations constitute a vector field, they require the choice of a reference frame, thus breaking rotation invariance. Several works have studied the neural computation of vector quantities on surfaces, which suffers a similar problem due to the choice of reference frame on the tangent plane. Multi-Directional Geodesic CNNs [Poulenard and Ovsjanikov 2018] address the challenge by computing quantities on multiple frames. Harmonic Surface Networks [Wiersma et al. 2020], on the other hand, extend harmonic nets to surfaces and achieve rotation-equivariance of vector quantities. In our collider descriptor, we adopt three different strategies: we compute a robust normal axis and thus achieve rotation-invariance with respect to the orientation of the normal of the collider, we use multiple random sampling of the tangential direction at training, and we apply bias on the tangential direction for efficiency and coherence at runtime.

Our collider descriptor is built from signed distance field (SDF) information. Other works also use SDF data to construct shape descriptors, such as probabilistic directed distance fields [Aumentado-Armstrong et al. 2022] or neural omnidirectional distance fields [Houchens et al. 2022]. However, these representations are more complex than the actual SDF and are designed for ray queries. Our approach is much simpler, as it only requires regular SDF information, not directional information. Deep Local Shapes [Chabra et al. 2020] reconstruct large surfaces from continuous local deep SDFs. In contrast, we do not use the SDF as intermediate shape representation, but as condition signal for the contact deformation field.

As anticipated in the introduction, in our work we explore a novel use case of shape descriptors for contact simulation. Neural Descriptor Fields [Simeonov et al. 2022] are a distant relative to our work. A neural descriptor encodes object manipulation conditioned by pose, and is used for inverse modeling of contact manipulation tasks. In our work, the descriptor is the input to the neural contact model, not the output. Interestingly, Chun et al. [2023] recently generalized Neural Descriptor Fields to unseen objects by utilizing local descriptors, similar in spirit to our extension to unseen colliders.

3 DEFORMATION BASED ON COLLIDER DESCRIPTORS

Given an arbitrary rigid collider and a certain reduced deformable object, we seek to design a neural deformation model that augments the reduced model with rich and detailed deformations resulting from the interaction with the collider. We start this section by formalizing the definition of the contact deformation model and motivating the collider descriptor that serves as input to the model. We continue with a detailed discussion of the components of the collider descriptor. To conclude, we formalize the neural model and we discuss how it is trained on a library of rigid colliders. Figure 2 outlines the computational pipeline of our contact deformation model.

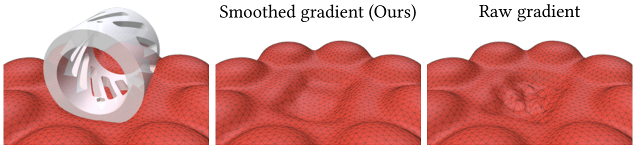


Figure 3: This collider with holes (left) produces many SDF gradient discontinuities. Using the raw gradient to define the contact frame (right) makes learning difficult, while our smoothed gradient (middle) helps learning.

3.1 Motivation of the Collider Descriptor

Let us formally define the kinematics and shape of both the rigid collider and the deformable object, and then the main components of our proposed model. We characterize the rigid collider by its rest-shape $\mathcal{S} \subset \mathbb{R}^3$ and a rigid transformation $T \in SE(3)$. Conversely, we characterize the deformable object by its reduced degrees of freedom (DoFs) $q \in \mathbb{R}^{n_r}$. Given a rest-shape parameterization $\bar{x} \in \mathbb{R}^3$, the baseline reduced deformed state is $x(\bar{x}, q) = \bar{x} + u(\bar{x}, q)$, with $u \in \mathbb{R}^3$ the reduced displacement field. We have demonstrated our neural deformation model on handle-based reduced deformations [Wang et al. 2015], but our formulation is agnostic of the reduced model for the most part. We only leverage the handle-based representation to parameterize several heuristics.

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

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

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

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

3.2 Contact Frame

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

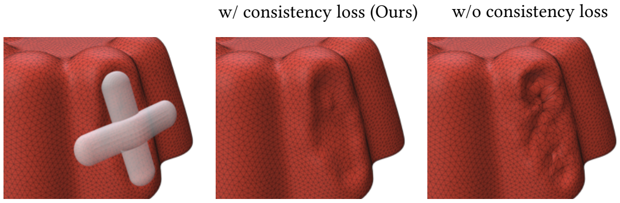


Figure 4: Even under smooth colliders (left), our consistency loss is necessary to ensure robustness with respect to tangent rotations of the contact frame (middle); deformations are not correctly learned without this loss (right).

$$r(x(\bar{x})) = T\, R(x)\, r_{\text{local}}(x). \tag{1}$$

By an abuse of notation, here T denotes only the rotational part of the collider transformation.

As shown by Romero et al. [2022], a contact-centric representation increases the learning ability of contact deformations. However, unlike their method, which used simply the transformation T of the collider and did not generalize to arbitrary colliders, we fit the frame R to the local shape of the collider, thus avoiding a collider-specific choice.

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

Normal Axis. To fit the normal axis, we analyze the surface geometry of the collider in the vicinity of the point of interest x. The SDF $\phi(x) \in \mathbb{R}$ of the collider serves as representation of this local geometry, hence we fit n based on the gradient of the SDF. However, as opposed to just evaluating the SDF gradient at x, we compute a smooth approximation of the gradient operation on the d-ball centered at x. This provides a continuous and smooth normal for the frame, which simplifies the learning task as shown in Figure 3.

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

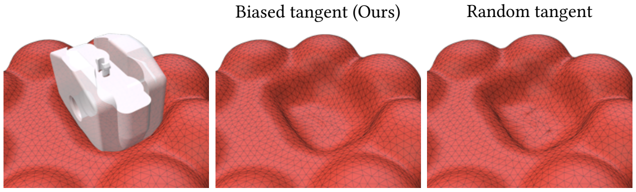
$$\nabla_B \phi(x) = \sum_{i=1}^{3} \frac{\phi(T^{-1}(x) + b_i) - \phi(T^{-1}(x) - b_i)}{2\,\|b_i\|^2}\, b_i. \tag{2}$$

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

$$\tilde{\nabla}\phi(x) = \frac{1}{N} \sum_{j=1}^{N} \nabla_{B_j} \phi(x). \tag{3}$$

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

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



Biased tangent (Ours)

Random tangent

Figure 5: For runtime inference, we bias the tangent rotation of the contact frame to maximize spatial and temporal coherence (middle). Using random rotations at runtime leads to noise (right).

Tangent Axes during Training. To make the deformation model r_{local} robust to relative transformations between the deformable object and the collider, we train it for random tangent rotations, but we enforce consistency of the output. To generate random rotations, we draw the tangent axis t_1 from a uniform distribution of orientations normal to n, and we define $t_2 = n \times t_1$. To enforce output consistency, we generate several random frames R for each point of interest x, and we add an explicit consistency loss term that penalizes the difference in the resulting contact deformation r for all these random frames. Figure 4 compares the result with and without the consistency loss. Please see more details about model training in Section 3.4.

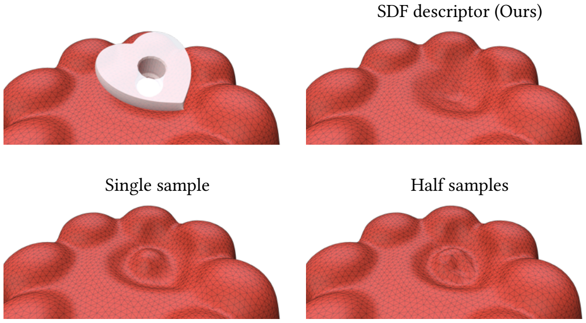
As an alternative to random rotations, we explored aligning the tangents t_1 and t_2 to principal directions of the SDF, similar to how we align the normal n to the gradient of the SDF. However, to deal with SDF discontinuities and locally flat regions, we found it was necessary to apply a similar randomization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

Tangent Axes at Runtime. While random sampling of tangent axes makes the model robust to rotations during training, it adds some flickering during runtime inference. As an alternative, at runtime we opt to bias model inference, to achieve both temporal and spatial coherence, as demonstrated in Figure 5. We pick arbitrary tangent directions as $t_1 = \frac{n\times(1,0,0)}{\|n\times(1,0,0)\|}$ and $t_2 = n \times t_1$.

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

3.3 Local SDF Descriptor

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



SDF descriptor (Ours)

Single sample

Half samples

Figure 6: This collider has a cylindrical void, not a hole passing all the way through (top left). Our local SDF descriptor correctly captures the deformation produced by the flat collider face, unaffected by the void (top right). When using a single SDF sample (bottom left) or half the samples as descriptor (bottom right), local shape is not correctly represented, and the void erroneously affects the deformation.

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

$$\hat{\phi} = \{\phi(T^{-1}(x) + R\,d\,w_i)\}. \tag{4}$$

We choose the (normalized) sampling pattern $\{w_i\}$ to be the cubature rule for degree-7 integration inside a sphere (which yields 64 samples) [Stroud 1971], plus a sample at the center. By orienting the sampling pattern according to the frame R, we achieve invariance of the descriptor with respect to rotations of the collider normal. Recall that R does not provide rotation invariance on the tangent plane, but the randomized training discussed in Section 3.2 makes the model robust to rotation.

Figure 6 demonstrates how the local SDF descriptor provides relevant local shape information for the computation of contact deformations. We compare our descriptor to using (a) $\hat{\phi} = \phi(T^{-1}(x))$, i.e., just the SDF at the point of interest x, and (b) $\hat{\phi} = \{\phi(T^{-1}(x) + R\,d\,w_i) \mid w_i^T\ (1,0,0)\ \leq\ 0\}$, i.e., only half of the sampling pattern along $-n$.

3.4 Neural Deformation Model

Given both components R and $\hat{\phi}$ of the collider descriptor $\hat{\mathcal{S}}$, we can formulate a neural model to compute the local contact deformations r_{local} in (1).

We input to the model the local SDF descriptor $\hat{\phi}$ and the deformation of the object q. Again, we adapt from Romero et al. [2022] the definition of the deformation input in local coordinates of the contact, but we use the local contact frame R, which generalizes to arbitrary colliders, as opposed to their global collider frame T, which is collider-specific. We also follow Romero et al. in using sparsifying weights $W(\bar{x}) = \text{diag}(U(\bar{x}))$, where U denotes the basis of the deformation model. This sparsification is a benefit of the handle-based reduced model.

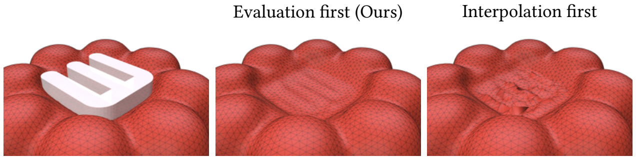


Figure 7: Given a precomputed grid of collider descriptor values, we fetch the descriptor values from grid points, evaluate the neural contact deformation model, and then interpolate the deformation result (middle). Interpolating the descriptor and then evaluating the model only once saves cost, but suffers strong noise (right).

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

$$r_{\text{local}}(x(\bar{x})) = \mathbb{N}\left(\hat{\phi}(x), W(\bar{x})\left(R(x)^{-1}\, T^{-1}\,(q-x)\right)\right). \qquad (5)$$

To express the deformation input q in local coordinates of the contact frame R, again we leverage the handle-based discretization of our reduced deformation model. For point handles in q, which define a translation, we use the transformation to R as expressed in (5) above. For frame handles, we use the same transformation for the translation part, and we simply omit the subtraction $-x$ for the rotation part.

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

Training of the Network. We train the model $\mathbb{N}$ for a particular deformable object in a supervised manner, providing ground-truth data of contact deformations for a variety of collider objects n_o, reduced object deformations n_d, and contact configurations n_c. The major loss term simply compares the error between predicted and ground-truth values of contact deformations. As already mentioned in Section 3.2, we add a self-supervised consistency loss for robust handling of the tangential orientation of the contact frame.

For each collider object, we start by executing an interactive simulation between the reduced deformable object and the collider, and we pick the n_d most distant deformations of the deformable object. For the rest of the ground-truth data generation, we freeze the reduced deformed state, and we execute full-space simulations constrained to this reduced state. We sample n_c contact configurations, which include a contact location on the surface of the deforming object, a relative transformation of the collider, and a collision depth. This procedure is similar to the one in [Romero et al. 2022],

For each training deformation, we include in the supervision data a random set of points of interest x. To select these points, we consider the interior of the collider object together with points within a d-ball from the collider surface, but we sample with lower probability points outside the collider, according to a normal distribution with standard deviation d. Furthermore, we force ground-truth deformations to attenuate toward 0 for points of interest at distance d from the surface of the collider, to ensure that the trained model does not suffer discontinuities.

Table 1: Complexity of the deformable objects used in the experiments, their training data set size (see Section 3.4), train and test error, and runtime performance.

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

4 RUNTIME MODEL EVALUATION

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

4.1 Precomputation of the Collider Descriptor

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

Recall that the collider descriptor $\hat{\mathcal{S}}$ is evaluated at points of interest x, which are transformed through T^{-1} to the local reference system of the collider. Therefore, we choose to precompute $\hat{\mathcal{S}}$ (both the contact frame R and the local SDF descriptor $\hat{\phi}$) on a grid surrounding the collider. We set up the grid on the bounding box of the collider enlarged by the ball radius d. In our experiments, we have used regular grids, with size 50 along the largest dimension for small colliders, and 90 for large colliders (in the finger and soft-ball examples, see Section 5). Finally, we cull the precomputation of the descriptor on points that are further than d from the surface of the collider. In our experiments, we obtained speed-ups between 1.6× and 3.3× (larger as d grows) when precomputing the collider descriptor.

Note that we only precompute the collider descriptor for runtime inference. As discussed in Section 3.2, for training we randomize the contact frame R, and hence we cannot leverage precomputation.

4.2 Grid Interpolation

Given collider descriptors at grid points, we need to evaluate the contact deformation model at arbitrary points of interest x. To do this, we transform x to the local reference system of the collider, locate its grid cell, fetch the collider descriptor at all 8 grid corners, evaluate the neural contact deformation model $\mathbb{N}$ at the grid corners, and trilinearly interpolate the deformations. First interpolating the

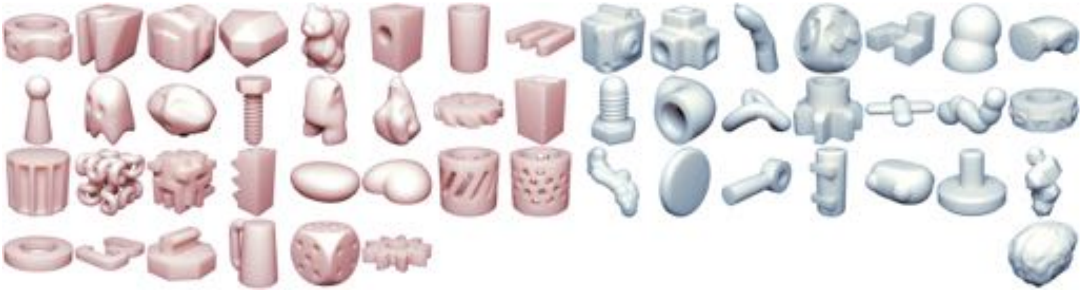


Figure 8: Collider objects (all from Thingi10K [Zhou and Jacobson 2016]) used for training (left, in red) and quantitative testing (right, in blue) our deformation model. Average error of our model is 17.0% on the train colliders and 17.8% on the test colliders, demonstrating its generalization capabilities.

collider descriptor and then evaluating the neural model just once is more efficient, but far less robust due to the interpolation of frames, as shown in Figure 7.

5 EXPERIMENTS AND RESULTS

5.1 Objects and Accuracy

Figure 8 shows the 30 colliders used for training and the 22 colliders used for quantitative testing in our experiments. All collider objects are part of the Thingi10K library [Zhou and Jacobson 2016], and they were selected to represent a variety of smooth and sharp shapes, voids and holes, and organic and synthetic objects. Note that in some experiments we use scene-specific colliders too, but they were not used for quantitative testing.

Table 1 summarizes the deformable objects used in our experiments. For each object, we indicate its reduced-model and mesh complexity, the number of objects × deformations × contact samples used for training, the train and test error, and runtime performance. To quantify error, we define as 100% error the mean square of ground-truth $r(x)$, i.e., the difference in deformation between the underlying linear reduced model and the full-space deformation. As shown in the table, the average error with train colliders is 17.0%, and it barely grows to 17.8% with test colliders, demonstrating the generalization capabilities of the proposed model.

We have also compared the accuracy of our model to collider-specific previous work [Romero et al. 2022]. The Jelly object was trained by Romero et al. with a pointy collider (see Figure 9), reporting 57% error on a contact interaction sequence. With our model, even if the pointy collider is not used for training, error barely changes to 58% (see a side-by-side comparison in the video).

5.2 Dynamic Simulation and Performance

In our experiments, we show the contact deformation model fully embedded in dynamic simulations. We formulate and solve the simulations using the optimization version of backward Euler [Kane et al. 2000], which boils down to solving for the deformation DoFs that minimize an energy $\Psi(r(q))$. We use a Newton solver with just one iteration per time step and conjugate gradient for the linear solve. We compute gradients $\frac{\partial \Psi}{\partial r}\frac{\partial r}{\partial q}$, which requires back propagation of the neural model, and we approximate Hessian-vector products in the linear solve as $\frac{\partial r}{\partial q}^T \frac{\partial^2 \Psi}{\partial r^2}\frac{\partial r}{\partial q} \cdot vector$, using an auxiliary back propagation [Romero et al. 2021].

We have executed all our experiments on an Intel Core i7-7700K 4-core 4.20 GHz PC with 32 GB of RAM, and we have implemented the neural networks using LibTorch on C++/CUDA. Table 1 summarizes the performance of our experiments. Speed-ups with respect to full-space simulations range between 9× and 26×.

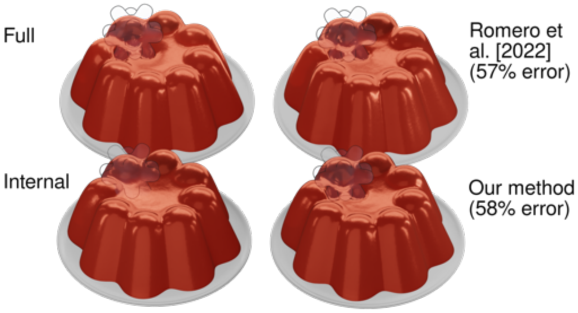


Figure 9: We compare our method to previous collider-specific work [Romero et al. 2022], and we achieve on par accuracy. We use as baseline for comparison their reduced model with data-driven internal corrections.

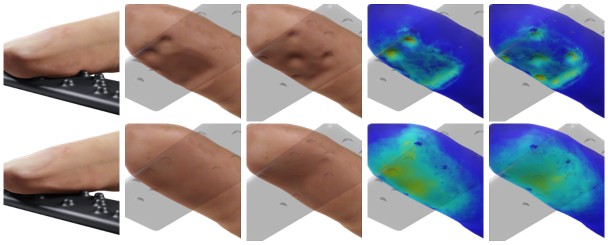


Figure 10: A soft finger reading braille. Our model (top) resolves the detailed deformations produced by braille dots, while the linear reduced model (bottom) fails. The right-most images show the norm of Green strain.

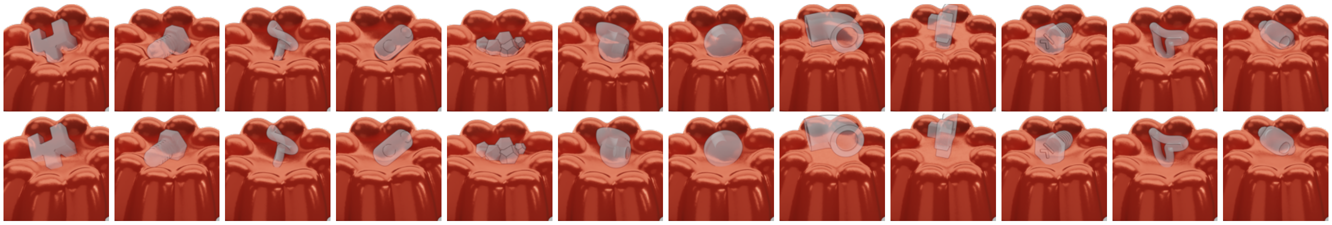


Figure 11: Dynamic rigid colliders falling on the jelly. With our model (top), the colliders produce detailed contact deformations. With the linear reduced model (bottom), deformations are too smooth.

5.3 Examples

In Figure 11, we showcase an example where the dynamics of the rigid collider are also simulated. Deformations and contact are detailed and robust in this case too, and they are excessively smooth for the linear reduced model.

In Figure 13, we showcase an example where the soft object, a spiky ball, has surface detail, showing that our method works well for objects whose surface is not smooth. With the linear reduced model, deformations are too smooth and the spikes retain much of their shape. With our model, on the other hand, the bars produce clear indentations, and the spikes collapse under contact, much like with the full model.

Figure 1 shows detailed contact deformations of a hand model exploring interactively different objects. None of the colliders in this example were used for training. This example also demonstrates the applicability to fine tactile exploration of objects in e-commerce applications, where the objects can be readily used for simulation without object-specific precomputation.

Finally, Figure 10 demonstrates the applicability of our method for virtual touch problems. A soft finger model reads the word "touch" written in braille. While the linear reduced model fails to resolve the deformations produced by braille dots, our model produces highly detailed deformations. Note that the braille collider is far from those used for training. The high-resolution strain field is also suitable for driving high-fidelity haptics [Verschoor et al. 2020].

6 DISCUSSION AND FUTURE WORK

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

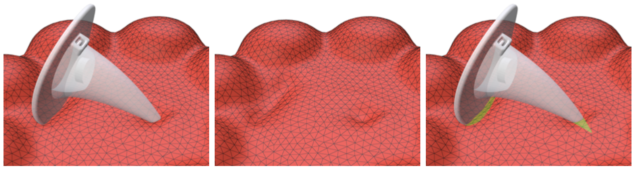


Figure 12: The quality of the deformations produced by our model is limited by the sampling resolution of the descriptor and the mesh resolution of the input deformation dataset. For this hat collider with a pointy end (left), our model fails to resolve a deformation with sufficient detail (middle), resulting in noticeable intersections (right).

Despite the novelty of our approach and results, there are still limitations that could inspire future work. In terms of low-level technical limitations, one is that the accuracy of deformations may be limited by the sampling resolution of the SDF descriptor as well as the mesh resolution of the input deformation dataset. Deformations with pointy features are not always resolved correctly, as shown in the example in Figure 12. Another low-level technical limitation is that our model needs to be trained with respect to rotations of the collider. This is for two reasons: the need to represent the result of the model (i.e. a vector quantity, the displacement field) in some reference system, and the definition of the SDF descriptor through sampling aligned with the local geometry. It would be interesting to bake the rotational invariance directly in the neural model, and avoid the construction of an explicit contact frame.

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

ACKNOWLEDGMENTS

We wish to thank the anonymous reviewers for their helpful comments, and Javier Tapia for help with the video. This work was funded in part by the European Research Council (ERC Consolidator Grant 772738 *TouchDesign*) and the Spanish Ministry of Science (grant TED2021-132003B-I00 *BLESIM*).

REFERENCES

Noam Aigerman, Kunal Gupta, Vladimir G. Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. 2022. Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes. *ACM Trans. Graph.* 41, 4, Article 109 (2022), 17 pages.

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

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

Hugo Bertiche, Meysam Madadi, and Sergio Escalera. 2022. Neural Cloth Simulation. *ACM Trans. Graph.* 41, 6, Article 220 (2022), 14 pages.

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

Xinhao Cai, Eulalie Coevoet, Alec Jacobson, and Paul Kry. 2022. Active Learning Neural C-space Signed Distance Fields for Reduced Deformable Self-Collision. In *Graphics Interface 2022*. https://openreview.net/forum?id=r3G_ReFNpM9

Dan Casas and Miguel A Otaduy. 2018. Learning nonlinear soft-tissue dynamics for interactive avatars. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1, 1 (2018), 10.

Rohan Chabra, Jan E. Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. 2020. Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction. In *Computer Vision â€“ ECCV 2020: 16th European Conference, Glasgow, UK, August 23â€“28, 2020, Proceedings, Part XXIX* (Glasgow, United Kingdom). Springer-Verlag, Berlin, Heidelberg, 608â€“625.

Peter Yichen Chen, Jinxu Xiang, Dong Heon Cho, Yue Chang, G A Pershing, Henrique Teles Maia, Maurizio M Chiaramonte, Kevin Thomas Carlberg, and Eitan Grinspun. 2023. CROM: Continuous Reduced-Order Modeling of PDEs Using Implicit Neural Representations. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=FUORz1tG8Og

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

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.

Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J. Mitra. 2018. PCPNet: Learning Local Shape Properties from Raw Point Clouds. *Computer Graphics Forum* 37, 2 (2018), 75–85. https://doi.org/10.1111/cgf.13343

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

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*.

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

C. Kane, J. E. Marsden, M. Ortiz, and M. West. 2000. Variational integrators and the Newmark algorithm for conservative and dissipative mechanical systems. *Internat. J. Numer. Methods Engrg.* 49, 10 (2000), 1295–1325.

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

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Incremental Potential Contact: Intersection-and Inversion-Free, Large-Deformation Dynamics. *ACM Trans. Graph.* 39, 4, Article 49 (2020), 20 pages.

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

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.

Qing Lyu, Menglei Chai, Xiang Chen, and Kun Zhou. 2022. Real-Time Hair Simulation With Neural Interpolation. *IEEE Transactions on Visualization and Computer Graphics* 28, 4 (2022), 1894–1905. https://doi.org/10.1109/TVCG.2020.3029823

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

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. 2021. Learning Mesh-Based Simulation with Graph Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=roNqYL0_XP

Adrien Poulenard and Maks Ovsjanikov. 2018. Multi-Directional Geodesic Neural Networks via Equivariant Convolution. *ACM Trans. Graph.* 37, 6, Article 236 (2018), 14 pages.

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

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

Cristian Romero, Dan Casas, Maurizio M. Chiaramonte, and Miguel A. Otaduy. 2022. Contact-Centric Deformation Learning. *ACM Trans. Graph.* 41, 4, Article 70 (2022), 11 pages.

Cristian Romero, Dan Casas, Jesús Pérez, and Miguel Otaduy. 2021. Learning Contact Corrections for Handle-Based Subspace Dynamics. *ACM Trans. Graph.* 40, 4 (2021).

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 Otaduy, Nils Thuerey, and Dan Casas. 2022b. ULNeF: Untangled Layered Neural Fields for Mix-and-Match Virtual Try-On. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 12110–12125. https://proceedings.neurips.cc/paper_files/paper/2022/file/4ee3ac2cd119023c79b0d21c4a464dc7-Paper-Conference.pdf

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

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

Nicholas Sharp, Cristian Romero, Alec Jacobson, Etienne Vouga, Paul G. Kry, David I. W. Levin, and Justin Solomon. 2023. Data-Free Learning of Reduced-Order Kinematics. arXiv:2305.03846 [cs.GR]

Siyuan Shen, Yin Yang, Tianjia Shao, He Wang, Chenfanfu Jiang, Lei Lan, and Kun Zhou. 2021. High-Order Differentiable Autoencoder for Nonlinear Model Reduction. *ACM Trans. Graph.* 40, 4, Article 68 (2021).

Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B. Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. 2022. Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*. 6394–6400. https://doi.org/10.1109/ICRA46639.2022.9812146

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.

A. H. Stroud. 1971. *Approximate calculation of multiple integrals*. Prentice-Hall.

Qingyang Tan, Zherong Pan, Breannan Smith, Takaaki Shiratori, and Dinesh Manocha. 2022a. N-Penetrate: Active Learning of Neural Collision Handler for Complex 3D Mesh Deformations. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 21037–21049.

Qingyang Tan, Yi Zhou, Tuanfeng Wang, Duygu Ceylan, Xin Sun, and Dinesh Manocha. 2022b. A Repulsive Force Unit forÂ Garment Collision Handling inÂ Neural Networks. In *Computer Vision – ECCV 2022*, Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (Eds.). Springer Nature Switzerland, Cham, 451–467.

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

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).

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. 2019. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions on Graphics (TOG)* (2019).

Ruben Wiersma, Elmar Eisemann, and Klaus Hildebrandt. 2020. CNNs on Surfaces Using Rotation-Equivariant Features. *ACM Trans. Graph.* 39, 4, Article 92 (2020), 12 pages.

Lingchen Yang, Byungsoo Kim, Gaspard Zoss, Baran Gözcü, Markus Gross, and Barbara Solenthaler. 2022. Implicit Neural Representation for Physics-Driven Actuated Soft Bodies. *ACM Trans. Graph.* 41, 4, Article 122 (2022), 10 pages.

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

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

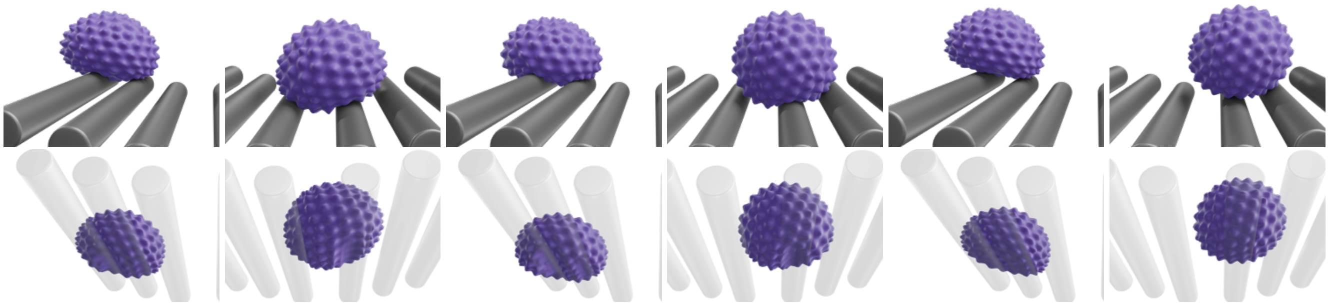


Figure 13: The images compare our method (4 left images), a full simulation (4 middle images), and the linear reduced model (4 right images), simulating contact of a soft spiky ball with rigid bars not seen at training. With the linear reduced model, deformations are too smooth and the spikes retain much of their shape. With our model, the bars produce clear indentations, and the spikes collapse under contact, much like with the full model.