Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
P. Kry and J. Lee (Editors)

Enriching Coarse Interactive Elastic Objects with High-Resolution Data-Driven Deformations

M. Seiler J. Spillmann M. Harders [†]

Computer Vision Lab, ETH Zurich, Switzerland

Abstract

Efficient approximate deformation models allow to interactively simulate elastic objects. However, these approaches usually cannot reproduce the complex deformation behavior governed by geometric and material nonlinearities. In addition, objects having slender shapes require dense simulation meshes, which necessitates additional computational effort. We propose an approach where a dynamic interactive coarse simulation *is enriched with details stemming from a more* accurate quasi-static simulation *in a data-driven way. While the coarse simulation is based on a low-resolution (low-res) mesh and a fast linear deformation model the accurate simulation employs a quasi-static non-linear deformation model at a higher mesh resolution (high-res). We pre-compute pairs of low-res mesh deformations and corresponding high-res details by applying a series of training interactions on both the coarse and the accurate model. At run-time, we only run the coarse simulation and correlate the current state to the training states. Subsequently, we blend detail data in order to obtain a spatio-temporally smooth displacement field that we super-impose on the surface skin, resulting in a plausible display of the non-linearly deformed object at real-time rates. We present examples from both computer animation and medical simulation.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Animation—Data-driven

1. Introduction

In this paper, we address the real-time setting where a user interacts through a single virtual probe with a virtual elastic object whose complex deformation behavior cannot be simulated in real-time. This is, for example, a common situation in surgery training simulators. The simulated objects may consist of heterogeneous or nearly incompressible materials, or may have a non-linear stress-strain relationship. A typical case are objects having a slender shape, such as the knee menisci, and in turn exhibit small-scale wrinkles and bulges around the interaction region. To represent such objects more convincingly, we propose a phenomenological approach that enriches an approximative *coarse real-time simulation* with *pre-computed detail data*. In contrast to previous works in the field of data-driven animation, our setting comes with a unique combination of challenges that our contributions address accordingly:

- *Unpredictable interactions*: The user's interactions come in an unpredictable manner and the space of possible interactions is nearly infinite. Our proposed solution handles novel deformations robustly by falling back to the coarse simulation, because providing training data for all possible interactions is infeasible and often unnecessary, as the coarse approximation is sufficient for most kinds of interaction (Sec. 5.1).
- *Robust displacement correlation*: We show that (fast) linear correlation cannot be used to select detail-data in the context of volumetric elastic objects. Consequently, we derive a robust *non-linear correlation measure* between displacement vectors that allows to compute correlation coefficients for deformations of elastic objects (Sec. 5.2).
- *Interpolation artifacts*: Standard vertex-based surface interpolation methods lead to visible artifacts if a probing-tool is moved over the object's surface. We propose *stamping*, a displacement interpolation method that accounts for moving boundary conditions by carrying the data *with* the tool while preserving high-frequency details (Sec. 5.3).

[†] {seiler,spillmann,mharders}@vision.ee.ethz.ch

DOI: 10.2312/SCA/SCA12/009-017

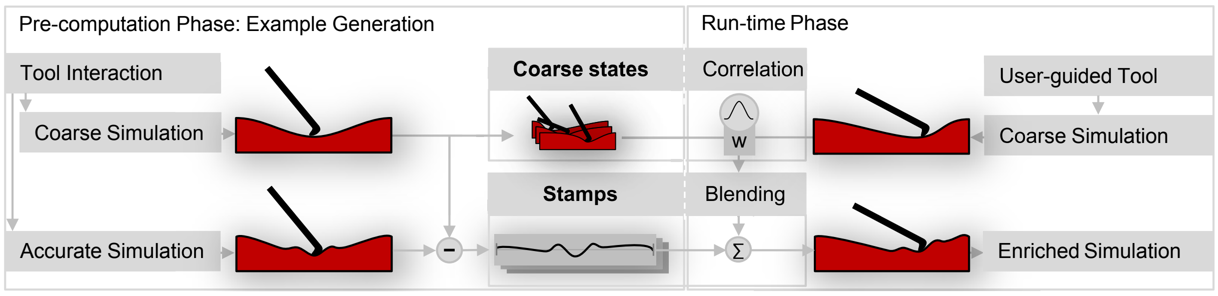

Figure 1: *Our method can be divided into two main phases: the pre-computation phase and the run-time phase. During pre-computation, the coarse simulation's input is a mix of tracked or procedurally generated tool interactions. The tool collides with the object and causes a deformation of the latter. The same procedure is done a second time, but with a more accurate quasi-static simulation. For each example interaction, the resulting difference between the two simulations is then used to create a displacement field, that we call stamp, for later usage. At run-time, only the coarse simulation runs in order to obtain an approximative deformed object. Our non-linear correlation allows to compute weights* $\mathbf{w}$ *that are used to blend* ($\sum$) *the example displacement stamps. Subsequently, the blended stamp is applied to the coarse object. The final result is an object with a detail-enriched surface.*

We focus on one tool (actually one interaction point) that probes an object. We also do not consider the deformation history, i.e. the data-driven details do not depend on how a certain deformation state is reached. This effectively avoids a combinatorial explosion of the number of required examples. Especially in the case of endoscopic surgery, our main motivation, this simplification is well-justified and effectively enables real-time performance as demonstrated by our results. We support our claims by illustrating various objects exhibiting complex non-linear deformations that are simulated and enriched with detail in real-time.

2. Related Work

The interactive simulation of elastic objects has a long research history in computer graphics; there exists a wealth of techniques to accomplish this goal. We briefly discuss them before focusing on work closely related to our approach of data-driven animation.

Efficient deformation models usually approximate elastic material behavior in order to obtain interactive speed. We refer to the report [NMK*05] for a listing of different simulation methods. In particular, we employ the linear co-rotational finite element (FE) method [MG04] for simulating the coarse mesh. However, the resolution required to represent detailed deformations prohibits real-time use. One possible remedy is adaptive methods [DDCB00, GKS02, MKB*08] which arrange the degrees-of-freedom (DOF) of the simulation mesh in regions of interest, or geometric techniques [MHTG05, MHHR07] that rely on explicit integration. In [ZWP05], a method is proposed where non-linear displacements are interpolated polynomially at the nodes, however, they do not increase the resolution of the mesh. Texture based deformation models, such as [GOM*06] who simulate an object using dynamic deformation textures, inspired our displacement interpolation method.

Surface embedding techniques improve the visual appearance of the elastic object by embedding a high-res surface mesh or *skin* into the coarser simulation mesh. This can be done by simple linear interpolation [MtLTM88], or non-linear techniques such as dual quaternion blending [KCZO07] or minimum least squares [KMBG09] which resolve the blending artifact of the linear method. While these methods improve the geometric richness of the objects, they do, however, not improve the deformation richness, since the deformation is still entirely governed by a coarser simulation mesh. It has also been shown that the deformation of a simulation mesh can be driven by an underlying material discretization at a higher resolution [NKJF09, KMOD09]. Still, these approaches cannot reproduce non-linear behavior such as buckling. Our method can be understood as a skinning variant, where pre-computed displacements are added to a linearly embedded skin.

Procedural detail generation approaches also employ a coarse simulation mesh, but then procedurally introduce details, based on the state of the coarse simulation. [KTJG08] add procedurally generated small-scale turbulence to a coarse smoke simulation. [RPC*10] augment a coarse cloth simulation with procedurally generated wrinkles along the principal stress directions. A promising alternative has been proposed in [MC10] where the wrinkles of a cloth are generated by evaluating a cheap static solver.

Data-driven methods exactly follow this idea: Instead of running a full simulation of a complex dynamic scene, parts of the scene are pre-computed or captured from real entities, and then added during the simulation, thereby saving computation time. This approach is not limited to geometric details, instead, also lighting parameters [MPBM03] can, *e. g.*, be added in a data-driven way. There exist different ways to obtain the data: Some approaches capture the motion of the real object, as shown in the context of facial animation [LCXS07, BLB*08], hand animation and grasping [KP06, HZY*11], soft-tissue modeling [BBO*09], or cloth animation [PZB*09]. Instead of visually capturing the deformed object, it can also be probed, *e. g.*, with a robot arm [LPW02]. Alternatively, the data can be artist-generated (in this context, one usually employs the term 'example' for a data record). This is usually done when exaggerated, non-physical deformations are desired [MTGG11]. However, the most frequently used approach is to pre-compute the data based on an off-line simulation method.

An important characteristic of data-driven approaches is the way the *correlation* between simulation states and the data points (say, 'examples') is performed. [DDSA11] select examples based on the position of the virtual tool, and then use radial basis functions (RBF) for the regression. Similar in spirit is the work of [Fon09] who render the deformations of virtual elastic objects based on the position of the contact point.

Most approaches rely on a low-dimensional, interactive control structure whose correlation between the current state and the pre-computed states determines interpolation weights. Some approaches employ a skeleton as low-dimensional control structure. In [KJP02], the influences of bones onto the skin are pre-computed on the basis of a principal component analysis (PCA), thus enabling the interpolation at run-time. Likewise, [FKY08] pre-compute the influence of a set of control points on the examples onto abstract bones, which are in turn employed at run-time to deform the skin. Later, [FYK10] combined their own method with the one of [KJP02] in order to limit the number of bones for small-scale details. In both works, it is proposed to interpolate the examples with a canonical correlation analysis, which is less prone to over-fitting than RBF. [dASTH10] opt for linear regression resulting from a least-squares objective, which is then combined with a simple dynamic model in order to animate clothed characters.

Pre-computed wrinkle examples are used in [WHRO10] to augment a simulated coarse cloth mesh. Similar, coarse states are 'up-sampled' using a linear operator in [KGBS11]. Additional harmonic regularization avoids over-fitting but smooths out details. The benefit of their approach is that the detail addition at run-time reduces to a single matrix-vector product. In contrast, we compute correlation coefficients in a non-linear way, thus avoiding artifacts that linear operators cannot overcome. Due to our local approach, our method exploits sparsity, and therefore, the performance is less dependent on the resolution of the coarse mesh. Further, our novel stamping method preserves high-frequency details even for a very small number of training data.

3. Overview

Our approach, schematically illustrated in Fig. 1, is based on the idea that the non-linearly deformed geometry of an object can be described in terms of a superposition of displacements stemming from an approximate deformation model and displacements which describe the deviation between the approximative model and the geometrically exact, non-linear deformed geometry [ZWP05]. The motivation for this decomposition comes from the observation that an approximative but efficient model, *i. e.* a mass-spring method, a linear co-rotational method [MG04], or a geometric method [MHTG05], computed on a coarse simulation mesh, is often sufficient to determine a plausible response, given time-varying boundary conditions. However, many real-world objects are composed of heterogeneous materials having spatially varying material parameters, which in turn induce deformations which cannot be captured with the aforementioned methods. To give an example, consider the inflated pillow depicted in Fig. 8: A plausible answer to a collision induced by a probe can easily be obtained by discretizing the interior of the pillow into a few volumetric elements and simulating the resulting mesh with a linear co-rotational FE method. However, in doing so, the fine wrinkles in the region of interaction cannot be reproduced, because they stem from the interplay between an inextensible surface sheet and the compressed interior material.

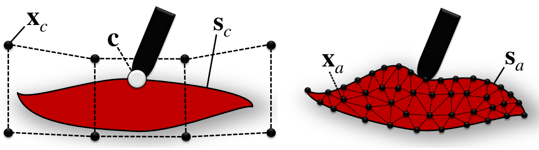

Figure 2: *A low-res mesh with nodal positions $\mathbf{x}_c$ is employed to interactively simulate the elastic object, and the surface skin $\mathbf{s}_c$ is linearly embedded. A key concept in our algorithm is the contact point $\mathbf{c}$, the center of the contact region. The accurate surface $\mathbf{s}_a$ is embedded in discrete points $\mathbf{x}_a$ that are deformed using a non-linear quasi-static simulation.*

We compute the deviation between the fast model and a physically more accurate, non-linear model as a pre-computation step. Similar to previous methods, we discretize the deformation function χ, that maps any point $\hat{\mathbf{x}} \in \mathbb{R}^3$ from the body's reference configuration to the deformed configuration $\mathbf{x} = \chi(\hat{\mathbf{x}}) \in \mathbb{R}^3$. First, we approximate the object with a coarse mesh with nodal positions stacked in a vector $\mathbf{x}_c \in \mathbb{R}^{3N_c}$ and second, with a high-res mesh with its corresponding positions stacked in $\mathbf{x}_a \in \mathbb{R}^{3N_a}$. Here, N_c is the number of coarse mesh nodes and N_a the number of high-res mesh nodes. Further, a high-res surface skin $\mathbf{s} \in \mathbb{R}^{3N_s}$ is used for the graphical representation, where N_s is the number of skin vertices (Fig. 2). The skin $\mathbf{s}$ is deformed into surface meshes $\mathbf{s}_c$ and $\mathbf{s}_a$ by the coarse and accurate simulation.

Both are obtained by linear interpolation, *i. e.* $\mathbf{s}_c = \mathbf{A}_c\mathbf{x}_c$ and $\mathbf{s}_a = \mathbf{A}_a\mathbf{x}_a$, where $\mathbf{A}_c$ and $\mathbf{A}_a$ are the interpolation operators. The quantity $\mathbf{d} = \mathbf{s}_a - \mathbf{s}_c$ then corresponds to the deviation between the embedded surface meshes of the fast, coarse and the non-linear, accurate method.

As fast deformation model we use the dynamic linear co-rotational FE method of [MG04], but we underline that our approach also works with other deformation models. Further, we use different accurate deformation models and mesh topologies depending on the particular geometric and material properties of the object to be simulated (see Sec. 6). Of note is that we use a static solver for the accurate method and consequently, we neglect dynamic effects in the data, which is similar to [MC10]. We then simulate both the fast model on the coarse object representation $\mathbf{x}_c$ and the accurate method on the high-res object representation $\mathbf{x}_a$ in tandem. To reproduce the deformations, a series of *training interactions* is performed, *i. e.* boundary conditions are induced by a virtual tool on both object representations, which results in deformed geometries $\mathbf{x}_c[i]$ and $\mathbf{x}_a[i]$, where i is a specific training interaction. The resulting surface pairs $(\mathbf{s}_c[i], \mathbf{s}_a[i])$ define high-res detail displacements $\mathbf{d}[i] = \mathbf{s}_a[i] - \mathbf{s}_c[i]$, which denote the deviation between the two methods and are stored in permanent memory. Besides the extracted displacements we also store additional information, namely the *contact point* $\mathbf{c}[i]$ (Fig. 2).

At run-time, we interactively simulate the coarse object representation $\mathbf{x}_c$. Based on the current deformed geometry $\mathbf{x}_c(t)$ at time t, we perform a non-linear correlation on the displacement data in order to select and interpolate high-res surface displacements $\mathbf{d}(t)$. These displacements are then added to the linearly interpolated skin $\mathbf{s}_c$ to obtain the enriched final result $\mathbf{s}_a(t)$ at time t,

$$\mathbf{s}_a(t) = \underbrace{\mathbf{A}_c\mathbf{x}_c(t)}_{\mathbf{s}_c(t)} + \mathbf{d}(t) \qquad (1)$$

To simplify the notation, we subsequently omit the time parameter t, that is, we define $\mathbf{u} = \mathbf{x}_c(t) - \mathbf{x}_c(0)$ to be the time-dependent coarse mesh displacements, and $\mathbf{d} = \mathbf{d}(t)$ to be the time-dependent high-res surface displacements. In the following text, we first describe how the training data is generated, covering questions on the choice of interaction sequences and tracking. Then, we provide details for the run-time stage.

4. Pre-computation Phase

The goal of the pre-computation phase is to obtain a set of example deformations, given deformation states of the coarse simulation mesh. We decided to obtain deformed states from a user interacting with the object via a *probe*, for instance the replica of a surgical tool. We call the set of interaction sequences *training interactions*.

4.1. Data Generation

To generate the examples we simulate both the coarse- and the accurate simulation in tandem. We emphasize again, that our method is independent of the employed deformation models of the two simulations. The tool geometry is moved according to a recorded 6-DOF position, imposing fixed displacement boundary conditions on the surface of the elastic object. As Kavan *et al.* pointed out [KGBS11], one must prevent divergence between the two simulations in order to produce meaningful results. In our case, divergence happens if the collision handling produces different results, *i. e.* the object might slip away from the tool in one simulation, but not in the other. In previous work, tracking has usually been handled by imposing hard constraints [MC10] or constraint forces [BMWG07] on the nodes of the high-res mesh. However, we noticed that this prohibits in some cases the occurrence of non-linear buckling behavior, which is of particular interest to us. Therefore, in our case, we opt for a constraint based approach: We first handle only collisions between the tool and the coarse representation. Then, we impose position- and velocity-constraints on the high-res nodes such that the skin vertices $\mathbf{s}_a = \mathbf{A}_a\mathbf{x}_a$ interpolated from the high-res simulation mesh correspond to the vertices $\mathbf{s}_c = \mathbf{A}_c\mathbf{x}_c$ interpolated from the coarse mesh in the contact region between the tool and the elastic object.

5. Run-time Phase

In this section, we develop our approach that allows to enrich a coarse simulation of an elastic solid with detailed surface deformations. In the pre-computation phase, we have generated n pairs $(\mathbf{u}[i], \mathbf{d}[i])$ using the coarse mesh displacements $\mathbf{u}[i]$ and corresponding high-res surface displacements $\mathbf{d}[i]$, $i \in [1, n]$. The enrichment process computes an interpolated displacement field $\mathbf{d}$, given the current coarse displacements $\mathbf{u}$.

Our method can be decomposed into two separate stages, notably a *correlation* that answers the question which pre-recorded examples i are best suited to enrich the current state with details by computing the per-example weights $\mathbf{w} = [w_i]$, and subsequently *data interpolation* of the displacements $\mathbf{d}[i]$ that are then blended using the previously computed weights w_i.

5.1. Linear Regression

Linear regression is a generic approach that performs correlation and interpolation in one stage, see *e. g.* [dASTH10, KGBS11]. We will now derive our non-linear method by first starting from a linear operator $\mathbf{G}$ obtained by linear regression. A generic regression function f computes $\mathbf{d} = f(\mathbf{u})$ and its parameters are obtained by minimizing $\sum_i ||f(\mathbf{u}[i]) - \mathbf{d}[i]||$, *i. e.* it minimizes the difference at the data-points in a least-squares sense. If we now restrict f to be a linear function we can write the equation in matrix form:

$$\mathbf{d} = \mathbf{G}\mathbf{u} \tag{2}$$

The linear operator $\mathbf{G}$ needs to be computed such that it minimizes $\sum_i ||\mathbf{G}\mathbf{u}[i] - \mathbf{d}[i]||$. This can be achieved by first stacking all example pairs $\mathbf{u}[i]$ and $\mathbf{d}[i]$ column-wise into matrices $\mathbf{U} = [\mathbf{u}[i]]$ and $\mathbf{D} = [\mathbf{d}[i]]$. Subsequently, since $\mathbf{U}$ is in general not a square matrix, we compute $\mathbf{G}$ by using the left Moore-Penrose pseudo-inverse:

$$\mathbf{D} = \mathbf{G}\mathbf{U} \tag{3}$$

$$\mathbf{G} = \mathbf{D}(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T \tag{4}$$

This set of equations can be used to analyse the linear operator. For example, Eqn. 5 reveals the linear blending factors $\mathbf{w}_l$ of the example data $\mathbf{D}$:

$$\mathbf{d} = \mathbf{G}\mathbf{u} = \mathbf{D}\underbrace{(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{u}}_{\mathbf{w}_l} \tag{5}$$

However, if the basis vectors $\mathbf{u}[i]$ are nearly co-linear, the condition of the system degrades rapidly and *over-fitting* is the undesired result. This problem can be attacked by improving the basis vectors [FYK10, dASTH10] or adding additional constraints [KGBS11]. Unfortunately, even if the over-fitting problem is solved, the special properties of the linear operator $\mathbf{G}$ in the context of co-linear vectors give rise to an artefact that we call the *stretching problem*.

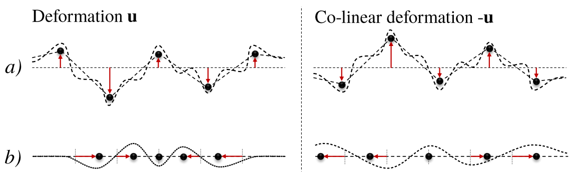

Figure 3: *For linear operators co-linear deformations* $\mathbf{u}$ *and* $-\mathbf{u}$ *always result in co-linear interpolated details* $\mathbf{d}$ *and* $-\mathbf{d}$. *While this is plausible in the case of orthogonal bending modes of thin shells (a), it is undesired if the details are induced by in-plane deformations (b). As expected, a compression results in wrinkles* $\mathbf{d}[1]$ *(b left). In contrast, false details* $-\mathbf{d}[1]$ *are added for stretching (b right).*

The stretching problem can be understood if we first compress the object and take an example $(\mathbf{u}[1], \mathbf{d}[1])$. Let us assume that $\mathbf{d}[1]$ is some wrinkle pattern. If we now stretch the object by the displacements $\mathbf{u} = -\mathbf{u}[1]$ we obtain the detail pattern $\mathbf{d} = \mathbf{G}\mathbf{u} = -\mathbf{d}[1]$ that corresponds to the inverted pattern of $\mathbf{d}[1]$ (Fig. 3). This behavior is physically implausible for most materials, as elastic objects do not exhibit a wrinkled surface when stretched. The problem is that common deformation patterns of elastic objects are *co-linear*, *i. e.* $\mathbf{u}[1] = s\,\mathbf{u}[2]$ for a scalar $s \in \mathbb{R}$, but the corresponding detail pattern is not. Simply switching to positions $\mathbf{x}_c$ instead of displacements $\mathbf{u}$ does not work as we can, due to linearity, always decompose the problem using $\mathbf{x}_c(t) = \mathbf{x}_c(0) + \mathbf{u}(t)$. Similarly, any other linear operators $\mathbf{H}$ can be combined with $\mathbf{G}$ to $\mathbf{G}' = \mathbf{H}\mathbf{G}$ or $\mathbf{G}'' = \mathbf{G}\mathbf{H}$. Therefore, this problem cannot be solved with any linear operator. To remedy this issue, our idea is to use a non-linear correlation measure that factors out the scaling factor between co-linear deformations and therefore allows to chose different detail patterns even for co-linear coarse vectors $\mathbf{u}$.

5.2. Deformation Correlation

Consequently, we factorize the coarse matrix $\mathbf{U}$ into a matrix $\mathbf{N} = [\frac{\mathbf{u}[i]}{||\mathbf{u}[i]||}]$ with normalized column vectors $\mathbf{u}[i]$ and a diagonal scaling matrix $\mathbf{S} = [s_{ii} = ||\mathbf{u}[i]||]$ that contains the lengths of the vectors. Similarly, the current coarse state $\mathbf{u} = \mathbf{n}s$ can be decomposed into the normalized vector $\mathbf{n} = \frac{\mathbf{u}}{||\mathbf{u}||}$ and scalar scaling $s = ||\mathbf{u}||$. For the preceding Eqn. 5 we obtain

$$\mathbf{w}_l = (\mathbf{S}^T\mathbf{N}^T\mathbf{N}\mathbf{S})^{-1}\mathbf{S}^T\mathbf{N}^T\mathbf{n}s \tag{6}$$

$$= s\mathbf{S}^{-1}(\mathbf{N}^T\mathbf{N})^{-1}\mathbf{N}^T\mathbf{n} \tag{7}$$

and it is revealed after reordering terms and cancelling $\mathbf{S}^{-1}$ with $\mathbf{S}$, that the inversion $(\mathbf{N}^T\mathbf{N})^{-1}$ is at the core of the linear regression. To illustrate its workings, let us put the i-th training sample $\mathbf{u} = \mathbf{u}[i]$ into Eqn. 5. We obtain the weight vector $\mathbf{w}$ that contains all zeros, except the i-th component contains a 1. It is precisely this inversion that allows for interpolating the data points, but at the same time it causes undesirable over-fitting and is unable to handle opposing deformations, as revealed by the stretching problem. Therefore, we opt to replace the inversion with a non-linear function $\Phi : \mathbb{R}^n \to \mathbb{R}^n$, where n is the number examples. Thus, our equation to compute the interpolation weights $\mathbf{w}$ evolves into

$$\mathbf{w} = s\mathbf{S}^{-1}\Phi(\mathbf{N}^T\mathbf{n}). \tag{8}$$

where the non-linear function Φ replaces the previous linear function that relied on $(\mathbf{N}^T\mathbf{N})^{-1}$.

Interestingly, the term $\mathbf{r} = [r_i] = \mathbf{N}^T\mathbf{n}$ corresponds to the well known *normalized cross-correlation* with the *correlation coefficients* $r_i \in [-1, 1]$ that correlate the current (normalized) coarse mesh deformation $\mathbf{n}$ to the example deformations stacked in $\mathbf{N}^T$. Intuitively, in case the cross-correlation r_i between an example i and the current deformation results in $r_i = 1$, then the current deformation has exactly the same direction as the example deformation. On the other hand, if it assumes $r_i = -1$ then the deformations are exactly in the opposite direction.

By using this, we can now solve the stretching problem and filter out opposing directions, *i. e.* remove all $r_i < 0$. In order to do this smoothly, we rely on the truncated Gaussian kernel G with kernel width d:

$$G(\sigma, x) = \begin{cases} \exp(-\frac{x^2}{\sigma^2}) & \text{if } x < 2\sigma \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Specifically, we define Φ_σ as a weighted-average interpolant that maps the cross-correlations $\mathbf{r}$ to non-linear correlations $\tilde{\mathbf{r}}$:

$$\tilde{\mathbf{r}} = \Phi_\sigma(\mathbf{r}) = \left[\phi_i = \frac{G(\sigma, r_i)}{\sum_j G(\sigma, r_j)} \right] \tag{10}$$

The parameter $\sigma \in [0, 0.5]$ needs to be chosen carefully in order to obtain satisfying results. If it is set too small then the interpolation becomes spiky and loses its smoothness. In contrast, we lose the ability to distinguish between different examples if it is set too large. Instead of tuning σ directly, we let the user pick the amount of examples n that she wants to mix. We then set the kernel width σ to the average of the n closest examples for each example found in $\mathbf{N}^T\mathbf{N}$.

Our goal was to handle opposing deformations and overfitting. Therefore, we replaced the multiplication with the matrix $(\mathbf{N}^T\mathbf{N})^{-1}$ with a non-linear function Φ_σ based on a truncated Gaussian kernel and weighted-average interpolation. Instead of strictly interpolating example data we approximate it. However, as one of our core assumptions is that the deformation around a contact point looks similar if we slightly shift the contact region (Fig. 4) exact interpolation is not a strict requirement in our setting. Summarizing, the final weight vector $\mathbf{w}$ can be computed based on Eqn. 8

$$\mathbf{w} = \left[w_i = \frac{||\mathbf{u}||}{||\mathbf{u}[i]||} \Phi_\sigma(\mathbf{N}^T \frac{\mathbf{u}}{||\mathbf{u}||}) \right] \tag{11}$$

and constitutes the input to the interpolation stage. In case one of the denominators becomes zero, we gracefully fade-out and do not add any displacements in the limit. This either means we have no deformation ($||\mathbf{u}|| \to 0$) or no correlating data ($\sum_j G(\sigma, r_j) \to 0$).

5.3. Displacement Interpolation

A first interpolation attempt to compute $\mathbf{d}$ could look like this:

$$\mathbf{d} = \mathbf{D}\mathbf{w} = \sum_i \mathbf{d}[i]\, w_i \tag{12}$$

In doing so, the resulting high-res surface displacements $\mathbf{d}$ converge towards $\mathbf{0}$ for current states which do not correlate, thus the surface converges towards $\mathbf{s}_c$. This *graceful degradation property* is particularly interesting in scenarios where an elastic object exhibits a non-linear behavior only in a certain region, while a large part of the elastic object can well be represented with an approximative model. However, the simple linear blending $\mathbf{d} = \mathbf{D}\mathbf{w}$ does not respect slight shifts δ of the contact point $\mathbf{c}$. This causes disturbing artifacts if the tool probes between two examples (Fig. 4 on the left), a situation that occurs inevitably if the tool is moved over the object's surface.

Blending deformation stamps is an elegant alternative to blending the displacements at the vertices, based on the observation that most real-world materials are composed of

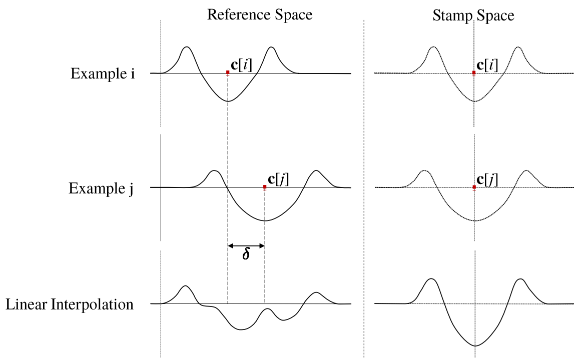

Figure 4: *When the tool is displaced from $\mathbf{c}[i]$ to $\mathbf{c}[j]$ having a distance of δ in the reference configuration, then simple linear interpolation of the two deformations in the reference space is subject to interpolation artifacts (left column). We propose to perform linear displacement interpolation in the* stamp space *originating at the contact point $\mathbf{c}$, thereby increasing the plausibility of the deformation pattern (right column).*

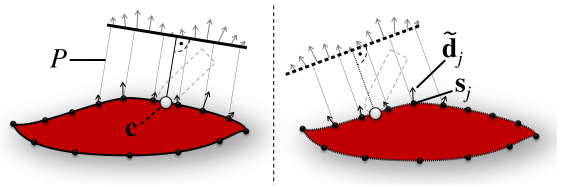

Figure 5: *The projection plane moves with the contact point $\mathbf{c}$ between the tool and the pillow surface. Each surface vertex $\mathbf{s}_j$ of the surface mesh $\mathbf{s}$ (depicted as black dots) is projected from the reference space to the stamp space using P. The sampled displacement $\tilde{\mathbf{d}}_j$ are then, after a rotation to surface tangential frames, added to the deformed surface vertices.*

spatially smooth material. To see what this means, assume that a virtual tool induces a coarse mesh deformation $\mathbf{u}$ and a corresponding high-res surface displacement $\mathbf{d}$. When the virtual tool is now shifted by a distance of δ, then shifting the high-res deformation accordingly yields better results (Fig. 4 right). To exploit *local material displacement smoothness* for the interpolation, we rely again on the *contact point* $\mathbf{c}$, that moves in the reference configuration on the surface $\mathbf{s}$ of the elastic object. In a deformation-based point of view, $\mathbf{c}$ will be situated in the center of the resulting deformation. Therefore, we may parametrize the surface displacements $\mathbf{d}[i]$ of example i in terms of a system (τ, θ) of surface coordinates originated at a contact point $\mathbf{c}[i]$. We denote the function $\Psi(\tau, \theta) : \mathbb{R}^2 \to \mathbb{R}^3$, that furnishes a parametric representation of the detail displacements, as the *stamp* at the contact point $\mathbf{c}$. In the pre-processing phase, we compute a stamp $\Psi[i]$ for each detail example $\mathbf{d}[i]$. Consequently, instead of using linearly blended displacements $\mathbf{d}_j \in \mathbb{R}^3$ for each surface vertex $j \in \{1...N_s\}$, we sample the surface displacements $\tilde{\mathbf{d}} = [\tilde{\mathbf{d}}_j]$ from a blended stamp chart $\Psi(\tau,\theta)$ that we compute using the weights $\mathbf{w}$ as $\Psi = \sum_i \Psi[i]\, w_i$. The required parametrization τ and θ are computed given the reference position $\mathbf{s}_j$ of the surface vertex. We propose to use a simple orthogonal projection to the tangent plane at the contact point $\mathbf{c}$ in reference space (Fig. 5). This orthogonal projection operator $P : \mathbb{R}^3 \to \mathbb{R}^2$ depends on the current contact point $\mathbf{c}$ and transforms points from the reference space to the stamp space. Accordingly, we can formulate the equation that is used to sample the displacements:

$$\tilde{\mathbf{d}}_j = \Psi(P(\mathbf{s}_j)) \qquad (13)$$

Notice that the linearly blended displacements $\mathbf{d}$ (Eq. 12) can be combined with stamping based displacements $\tilde{\mathbf{d}}$ (Eq. 13). For instance, in the pillow simulation, we prefer stamping in a region around the contact point and use the linear blending for the rest of the object (Fig. 8).

6. Results

In this section, we evaluate our approach and present examples from the fields of computer animation and medical simulation. The employed simulation methods are described below. All experiments have been staged on an Intel Core i7 with an Nvidia GTX 460 graphics card.

The coarse simulation uses a linear co-rotational FE method deformation model based on hexahedral elements that allows to simulate the coarse mesh in real-time [MG04]. Further, the surface skin is embedded with a moving least squares approach [KMBG09] in order to obtain a smooth surface mesh $\mathbf{s}_c$. To detect the collisions, we employ spatial hashing [THM*03], and to handle the collisions, a simple position-based approach is applied. We use the Coulomb friction model to mimic dry friction. Note, that this simulation is not only used to integrate the body at run-time and compute deformation descriptors for each tool in contact, but also to compute the boundary constraints of the accurate simulation during the data generation phase.

The accurate simulation, that generates the detailed deformations given fixed boundary constraints, uses a thin-shell deformation model based on [BMF03]. In order to obtain the bulging stemming from the orthogonal compression the compressed triangle elements are stretched accordingly. Further, to simulate the wrinkled pillow, we additionally compute a volume preservation term [MHHR07].

6.1. Meniscus

Our approach can be employed to enrich coarse simulations with high-res details in areas that have been trained previously, while it gracefully degrades to the linearly embedded skin for interactions away from the training states. We exemplify this property on the basis of an interactive simulation of the medial meniscus in the knee joint. Meniscus tissue is almost incompressible and exhibits a characteristic bulging behavior when compressed orthogonally. By employing our method, we can plausibly mimic this behavior when the user presses the meniscus down onto the tibial plateau (Fig. 6 c). Nevertheless, the linear co-rotational FE method is sufficient to reproduce the deformation of the meniscus when it is lifted up (Fig. 6 d). Thus, we only train interactions that compress the meniscus, while the graceful degradation property smoothly transitions to the linear method for all other interactions. In turn, this greatly reduces the number of necessary training interactions. For this setting, we have employed 119 training examples, requiring 8 MB of memory. The coarse mesh consists of 80 elements, the high-res mesh consists of 890 elements, and the surface skin has 3212 vertices. The time for the correlation and regression is 1.7ms, while the simulation runs at 208 frames per second, including the rendering.

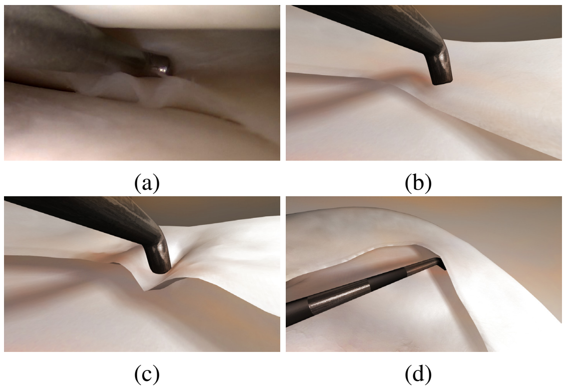
(a) (b) (c) (d)

Figure 6: *A real meniscus deforms characteristically when probed with a surgical tool (a). The coarse simulation with the linearly embedded skin cannot reproduce the characteristic local deformation pattern (b). We add the displacements resulting from the deviation to the non-linear model in a data-driven way (c). However, for interactions that lift up the meniscus the coarse method is sufficient to reproduce the deformations. The smooth transition between trained and new deformations is enabled by the graceful degradation property of our approach (d).*

6.2. Wedge

Regarding performance, our approach depends linearly on the number of training examples, but correlation stays well below 1 ms even for more than 100 examples (Table 6.2). However, computing an initial embedding and especially blending stamp textures is more expensive; but, as we will see, sparsity can easily be exploited to improve performance. To show this, we perform an experiment where a wedge-shaped object is interactively manipulated. In the preprocessing phase, we choose a varying number of train-

	#Ex.	Corr.	Blend	Embed & Render
Wedge	128	0.11 ms	1.1 ms	0.2 ms
Meniscus	80	0.09 ms	0.3 ms	0.13 ms
Pillow	32	0.08 ms	0.6 ms	0.5 ms

Table 1: *The example number (#Ex.) has only minor impact on the total run-time. The correlation (Corr.) is computed on the CPU and controls how the stamps are blended. Blending (Blend) 10 stamps is done on the GPU as well as the final vertex embedding (Embed & Render) that is combined with rendering, as it is implemented in the same GPU shader.*

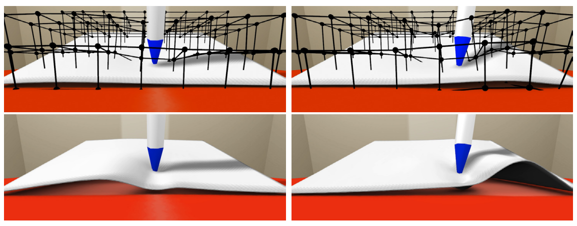

Figure 7: *Our method can be employed to handle deformations induced by dry friction. By displacing the tool tangentially to the surface during the training and pre-computing the corresponding high-res mesh, the bulging effects stemming from the dry friction can be reproduced at run-time. Two different states are shown (left and right). While the top row shows the coarse simulation the bottom row shows the enriched result.*

ing interactions. When less interactions are trained, then the method tends to quickly fall back to the linearly embedded skin. We found that from at least 40 training interactions selecting the 10 most correlating (determined using $\tilde{\mathbf{r}}$) example stamps is sufficient in order to obtain faithful results. This scenario also illustrates that our approach can handle frictional deformation effects. That means, interactions are trained where the tool is displaced tangentially to the surface of the object, resulting in a tangentially deformed coarse simulation mesh and a corresponding bulge in the high-res simulation (Fig. 7). The coarse mesh consists of 128 elements, the high-res mesh consists of 570 elements, and the surface skin has 6.9K vertices.

6.3. Inflated Pillow

In this scenario, we simulate an inflated pillow with a diameter of 40 cm. It exhibits characteristic patterns of wrinkles and folds around the interaction area. By employing our stamping approach (Eq. 13) only in a region of about 10 cm around the contact point and by fading into displacements obtained using the vector method (Eq. 12) we are able to preserve the details (Fig. 8) remarkably well. We have trained the simulation with 36 interactions, the coarse mesh consists of 32 elements, the high-res simulation mesh consists

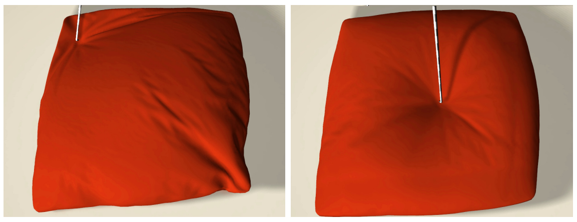

Figure 8: *A coarse simulation of an inflated pillow is globally enriched with high-frequency details stemming from a more accurate simulation. Although some artifacts are visible, our method reproduces stunning non-linear effects using less than 1 ms additional computation time.*

of 29K elements, and the surface skin has also 29K vertices. The interactive simulation runs at 58 frames per second. As our stamping approach moves the wrinkles of a static equilibrium coherently along the surface, artifacts become visible when the tool moves over the surface. With the current method, this can be partially remedied by adding more examples.

7. Conclusion

We have devised an approach that allows to enrich a coarse dynamic interactive simulation of an elastic object with static high-res detail deformations in a data-driven way. By carefully analyzing the decomposition of a purely linear approach we proposed a non-linear strategy that preserves the performance characteristic of a linear method. In order to increase the quality of the interpolation, we proposed *stamping*. A simple but effective approach that projects the displacement stamps relative to the contact point, and thus produces less artifacts.

Still, our solution has limitations that we will address in future work. We currently only support one contact-point. Multiple tools produce deformations which might differ substantially from what one records during interaction with one tool, thus the added displacements might interfere and look implausible. However, our method could be extended by considering only a local region around multiple contact points. Further extensions are possible in the stamp blending procedure as well as in the parameterization operator P, especially if stamping is used in geometrically more complex regions or when dynamics play an important role. Consequently, we plan to extend our method to handle dynamic deformations.

8. Acknowledgement

This work has been supported by the Swiss CTI project *RapidPRO 12888.1*. Many thanks go to the team at VirtaMed and to the anonymous reviewers for their helpful comments on earlier versions of this paper.

References

[BBO*09] Bickel B., Baecher M., Otaduy M., Matusik W., Pfister H., Gross M.: Capture and modeling of non-linear heterogeneous soft tissue. *Proceedings of ACM SIGGRAPH 28* (2009). 3

[BLB*08] Bickel B., Lang M., Botsch M., Otaduy M. A., Gross M.: Pose-space animation and transfer of facial details. In *ACM SIGGRAPH/SCA* (2008), pp. 57–66. 3

[BMF03] Bridson R., Marino S., Fedkiw R.: Simulation of clothing with folds and wrinkles. In *ACM SIGGRAPH/Eurographics SCA* (2003), pp. 28–36. 7

[BMWG07] Bergou M., Mathur S., Wardetzky M., Grinspun E.: Tracks: toward directable thin shells. *ACM TOG (SIGGRAPH) 26*, 3 (2007), 50. 4

[dASTH10] de Aguiar E., Sigal L., Treuille A., Hodgins J. K.: Stable spaces for real-time clothing. *ACM TOG 29*, 3 (2010). 3, 4, 5

[DDCB00] Debunne G., Desbrun M., Cani M.-P., Barr A. H.: Adaptive simulation of soft bodies in real-time. In *Computer Animation* (2000), pp. 15–20. 2

[DDSA11] De S., Deo D., Sankaranarayanan G., Arikatla V. S.: A physics-driven neural networks-based simulation system (phynness). *Presence: Teleoper. Virtual Environ. 20* (2011), 289–308. 3

[FKY08] Feng W.-W., Kim B.-U., Yu Y.: Real-time data driven deformation using kernel canonical correlation analysis. *ACM TOG 27* (August 2008), 91:1–91:9. 3

[Fon09] Fong P.: Sensing, acquisition, and interactive playback of data-based models for elastic deformable objects. *The International Journal of Robotics Research 28*, 5 (2009), 630. 3

[FYK10] Feng W.-W., Yu Y., Kim B.-U.: A deformation transformer for real-time cloth animation. In *ACM SIGGRAPH* (2010), pp. 108:1–108:9. 3, 5

[GKS02] Grinspun E., Krysl P., Schröder P.: CHARMS: a simple framework for adaptive simulation. *ACM TOG (SIGGRAPH) 21*, 3 (2002), 281–290. 2

[GOM*06] Galoppo N., Otaduy M. A., Mecklenburg P., Gross M., Lin M. C.: Fast simulation of deformable models in contact using dynamic deformation textures. In *Proc. Eurographics/ACM SIGGRAPH SCA* (2006), pp. 73–82. 2

[HZY*11] Huang H., Zhao L., Yin K., Qi Y., Yu Y., Tong X.: Controllable hand deformation from sparse examples with rich details. In *ACM SIGGRAPH/SCA* (2011), pp. 73–82. 3

[KCZO07] Kavan L., Collins S., Zára J., O'Sullivan C.: Skinning with dual quaternions. In *Proc. symposium on Interactive 3D graphics and games* (2007), pp. 39–46. 2

[KGBS11] Kavan L., Gerszewski D., Bargteil A. W., Sloan P.: Physics-inspired upsampling for cloth simulation in games. In *ACM SIGGRAPH* (2011), pp. 93:1–93:10. 3, 4, 5

[KJP02] Kry P. G., James D. L., Pai D. K.: Eigenskin: Real time large deformation character skinning in hardware. In *In ACM SIGGRAPH SCA* (2002), ACM Press, pp. 153–159. 3

[KMBG09] Kaufmann P., Martin S., Botsch M., Gross M.: Flexible simulation of deformable models using discontinuous galerkin fem. *Graph. Models 71*, 4 (2009), 153–167. 2, 7

[KMOD09] Kharevych L., Mullen P., Owhadi H., Desbrun M.: Numerical coarsening of inhomogeneous elastic materials. *ACM TOG 28*, 3 (2009), 51. 2

[KP06] Kry P. G., Pai D. K.: Interaction capture and synthesis. *ACM TOG 25* (2006), 872–880. 3

[KTJG08] Kim T., Thürey N., James D., Gross M.: Wavelet turbulence for fluid simulation. *ACM TOG 27* (2008), 50:1–50:6. 2

[LCXS07] Lau M., Chai J., Xu Y.-Q., Shum H.-Y.: Face poser: interactive modeling of 3d facial expressions using model priors. In *ACM SIGGRAPH/Eurographics SCA* (2007), pp. 161–170. 3

[LPW02] Lang J., Pai D., Woodham R.: Acquisition of elastic models for interactive simulation. *The International Journal of Robotics Research 21*, 8 (2002), 713–733. 3

[MC10] Müller M., Chentanez N.: Wrinkle meshes. In *ACM SIGGRAPH/SCA* (2010), pp. 85–92. 2, 4

[MG04] Müller M., Gross M.: Interactive virtual materials. In *Proc. Graphics Interface* (2004), pp. 239–246. 2, 3, 4, 7

[MHHR07] Müller M., Heidelberger B., Hennix M., Ratcliff J.: Position based dynamics. *Journal of Visual Comm. and Image Representation 18*, 2 (2007), 109–118. 2, 7

[MHTG05] Müller M., Heidelberger B., Teschner M., Gross M.: Meshless deformations based on shape matching. *ACM TOG (SIGGRAPH) 24*, 3 (2005), 471–478. 2, 3

[MKB*08] Martin S., Kaufmann P., Botsch M., Wicke M., Gross M.: Polyhedral finite elements using harmonic basis functions. *Comp. Graph. Forum 27*, 5 (2008), 1521–1529. 2

[MPBM03] Matusik W., Pfister H., Brand M., McMillan L.: A data-driven reflectance model. In *ACM SIGGRAPH* (2003), pp. 759–769. 3

[MTGG11] Martin S., Thomaszewski B., Grinspun E., Gross M.: Example-based elastic materials. *ACM TOG (SIGGRAPH) 30*, 4 (2011), 72:1–72:8. 3

[MtLTM88] Magnenat-thalmann N., Laperrire R., Thalmann D., Montr£¡eal U. D.: Joint-dependent local deformations for hand animation and object grasping. In *In Proceedings on Graphics interface 88* (1988), pp. 26–33. 2

[NKJF09] Nesme M., Kry P. G., Jeráková L., Faure F.: Preserving topology and elasticity for embedded deformable models. In *ACM SIGGRAPH* (2009), pp. 52:1–52:9. 2

[NMK*05] Nealen A., Müller M., Keiser R., Boxermann E., Carlson M.: Physically Based Deformable Models in Computer Graphics. In *EG-STAR* (2005), pp. 71–94. 2

[PZB*09] Popa T., Zhou Q., Bradley D., Kraevoy V., Fu H., Sheffer A., Heidrich W.: Wrinkling captured garments using space-time data-driven deformation. *Comp. Graph. Forum (Proc. Eurographics) 28*, 2 (2009), 427–435. 3

[RPC*10] Rohmer D., Popa T., Cani M.-P., Hahmann S., Sheffer A.: Animation wrinkling: augmenting coarse cloth simulations with realistic-looking wrinkles. In *ACM SIGGRAPH Asia* (2010), pp. 157:1–157:8. 2

[THM*03] Teschner M., Heidelberger B., Müller M., Pomerantes D., Gross M.: Optimized spatial hashing for collision detection of deformable objects. In *Proc. Vision, Modeling, Visualization* (2003), pp. 47–54. 7

[WHRO10] Wang H., Hecht F., Ramamoorthi R., O'Brien J.: Example-based wrinkle synthesis for clothing animation. *ACM TOG 29*, 4 (2010), 107:1–107:8. 3

[ZWP05] Zhong H., Wachowiak M., Peters T.: A real time finite element based tissue simulation method incorporating nonlinear elastic behavior. *Computer Methods in Biomechanics and Biomedical Engineering 8*, 3 (2005), 177–189. 2, 3