An Efficient Construction of Reduced Deformable Objects

Christoph von Tycowicz[1] Christian Schulz[2] Hans-Peter Seidel[2] Klaus Hildebrandt[2]
[1]Freie Universität Berlin [2]Max-Planck-Institut für Informatik

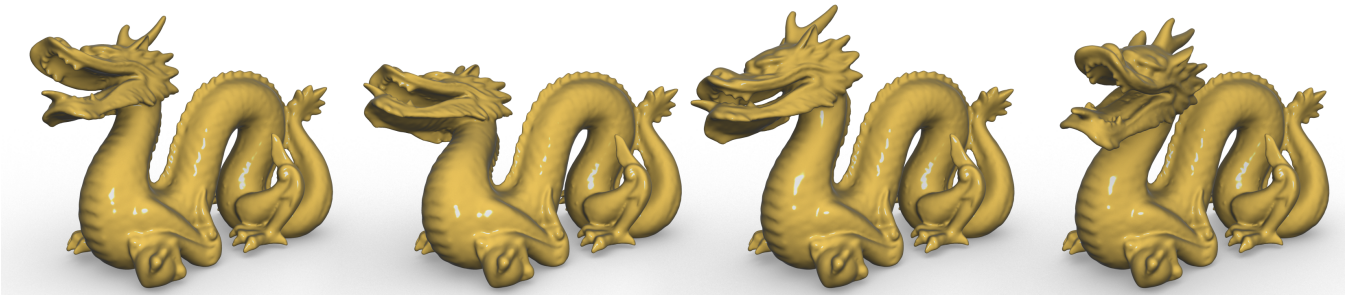

Figure 1: *Nonlinear simulation of a deformable object with 92 k tets computed at over 120 Hz after about 4 mins of preprocessing.*

Abstract

Many efficient computational methods for physical simulation are based on model reduction. We propose new model reduction techniques for the *approximation of reduced forces* and for the *construction of reduced shape spaces of deformable objects* that accelerate the construction of a reduced dynamical system, increase the accuracy of the approximation, and simplify the implementation of model reduction. Based on the techniques, we introduce schemes for real-time simulation of deformable objects and interactive deformation-based editing of triangle or tet meshes. We demonstrate the effectiveness of the new techniques in different experiments with elastic solids and shells and compare them to alternative approaches.

CR Categories: I.6.8 [Simulation and Modeling]: Types of Simulation—Animation I.3.5 [Computer Graphics]: Computation Geometry and Object Modeling—Physically-based modeling;

Keywords: real-time simulation; geometric modeling; model reduction; subset selection; modal analysis; modal derivatives

Links: DL PDF

1 Introduction

Methods for real-time simulation of deformable objects based on model reduction have received much attention in recent years. These schemes construct a low-dimensional approximation of the dynamical system underlying a simulation and thereby achieve a runtime that depends only on the complexity of the low-dimensional system. We focus on two problems of reduced nonlinear simulation: the *subspace construction* and the *efficient approximation of the reduced forces*. We propose new techniques for both problems aiming at accelerating the construction of the approximate dynamical system, increasing the accuracy of the approximation, and simplifying their implementation. Based on this, we implement schemes for real-time simulation of deformable objects and for deformation-based editing of triangular or tetrahedral meshes. Beyond these two applications, the developed techniques are potentially useful for other applications including the acceleration of large simulations and the reduction of constrained spacetime optimization problems, *e.g.* for motion design.

Approximation of reduced forces In addition to dimension reduction, the real-time simulations of deformable objects require a scheme for efficiently approximating the nonlinear reduced forces. The force approximation we consider here, follows the *optimized cubature* introduced by An et al. [2008]. Typically interior forces of a discrete deformable object can be written as a sum whose summands depend only on the deformation of a local neighborhood of the object, *e.g.* a triangle or a tet. The idea is to exploit the correlations between these summands. The dimension reduction restricts the system to a small number of degrees of freedom, which in turn strengthens the correlations. The strategy is to select a small number of summands and to approximate the reduced forces by a linear combination of these summands. The subset and weights are determined through an optimization procedure in which the approximation error on an automatically generated set of training poses is minimized. This is a constrained best subset selection problem. In this paper, we devise a new scheme for efficiently solving this problem, which is based on recent advances in the field of sparse approximation. Our strategy for solving the subset selection problem is substantially different from that used in [An et al. 2008]. They use a greedy strategy that iteratively constructs the selection set by successively adding one entity per iteration. In contrast, our scheme constructs a complete selection set in the first iteration and the whole selection set can be changed in subsequent iterations. We demonstrate in a number of examples that our scheme can produce a significantly smaller approximation error at lower computational costs and is able to achieve a given training error with a smaller selection set.

Subspace construction Subspace construction based on linear modal analysis has become standard practice for the dimension reduction of linear second-order dynamical systems. However, for

Figure 2: *Interactive deformation-based editing of a Chinese dragon model with 260 k triangles (see Table 1).*

nonlinear systems, such a basis cannot capture the effects of the nonlinearities, which for the simulation of deformable objects leads to artifacts for large deformations. We propose a simple, yet effective, technique for extending modal bases and demonstrate that the resulting subspaces can better represent the nonlinear behavior of deformable objects. The idea is to use linear transformations of $\mathbb{R}^3$ to create new basis vectors. In contrast to common usage, we do not apply the linear transformations to deform the geometry directly, but to "deform" the linear vibration modes. The resulting new basis vectors depend not only on the geometry, but also on the material properties of the deformable object. Using this strategy, modal bases are extended in such a way that the spanned subspaces better approximate large deformations. In our experiments, we found the resulting effect comparable to that achieved by adding modal derivatives to linear modal bases. Benefits of the proposed technique are that the construction is fast and simple to implement.

2 Related Work

Using subspaces constructed from linear vibration modes to accelerate the integration of linear second-order dynamical systems is a standard technique with a long tradition [Pentland and Williams 1989]. Still, the question of how this technique can be generalized to nonlinear systems is an active area of research. One strategy is to compute the vibration modes around several states and to use the span of the union of these modes. The drawback of this approach is the high computational cost for solving several eigenvalue problems. An alternative approach is to enrich the modal basis with *modal derivatives* [Idelsohn and Cardona 1985; Barbič and James 2005; Hildebrandt et al. 2011; Tiso 2011]. For any pair of modes, a modal derivative can be generated. The computation involves second derivatives of the forces. Roughly speaking, the modal derivative computed from a pair of modes describes how one mode changes when the object is deformed in the direction of the other mode. Commonly a symmetrized version of the modal derivatives, in which an order of the pair of modes is not relevant, is used. An alternative to using modal analysis is to construct reduced spaces based on a principal component analysis of a set of observations [Krysl et al. 2001].

After dimension reduction, the cost for evaluating the nonlinear interior forces of a deformable object is still high as it requires computing and projecting the unreduced forces. *Optimized cubatures* have been successfully applied for approximating the interior forces of different types of hyperelastic materials for elastic solids [An et al. 2008] and thin shells [Chadwick et al. 2009]. Recently, they were also used for reduced fluid re-simulations [Kim and Delaney 2013]. Computing the cubature points requires solving a complex optimization problem: a best subset selection problem. To solve the problem, current schemes use a greedy strategy that incrementally builds the selection set. An alternative to force approximation is the exact evaluation of the reduced forces. For linear materials, *e.g.* the St.Venant–Kirchhoff model of elastic solids, the forces are cubic polynomials on the shape space. In this case, the coefficients of the restriction of the polynomials to the reduced space can be precomputed [Barbič and James 2005]. This yields an exact representation of the forces in the reduced space and evaluation costs that depend only on the size of the subspace. However, the number of coefficients to be precomputed and evaluated at runtime grows quartically with the dimension of the reduced space.

In addition to real-time simulation, model reduction has been used to accelerate large simulations [Kim and James 2009] and for controlling the motion of deformable objects [Barbič and Popović 2008; Barbič et al. 2009; Barbič et al. 2012; Hildebrandt et al. 2012] as well as characters [Safonova et al. 2004] and fluids [Treuille et al. 2006; Wicke et al. 2009]. Moreover, in [Hahn et al. 2012] simulations in reduced spaces obtained from animators' rigs were considered with the goal of simplifying the integration of simulation into the traditional animation pipeline. Subset selection based on training poses has also been used for facial articulation and global illumination by Meyer and Anderson [2007].

Alternative approaches for real-time simulation of deformable objects in a reduced space are *modal warping* [Choi and Ko 2005] and *rotation strain coordinates* [Huang et al. 2011]. These schemes integrate a linearized system in modal coordinates and *warp* the solutions to counteract artifacts produced by the linearization.

In geometry processing, deformable objects are used for editing shapes. In such a *deformation-based editing* system, see [Botsch and Sorkine 2008; Botsch et al. 2010] and references therein, a user can select parts of a geometry as handles and translate and rotate them in space. The system automatically deforms the shape so that the handles interpolate or approximate the specified positions. This is done by computing static equilibrium states of the deformable object subject to constraints or external forces that represent the user's input. A major advantage of deformation-based editing over traditional modeling techniques, like NURBS or subdivision surfaces, is that many complex editing tasks can be described by few constraints. This allows for efficient and simple click-and-drag user interfaces. To obtain an interactive editing system for larger models, methods that accelerate the computation based on space deformation [Sumner et al. 2007; Botsch et al. 2007; Ben-Chen et al. 2009], skinning [Jacobson et al. 2012], and model reduction [Hildebrandt et al. 2011] have been proposed. Recently, reduced deformable objects were used to create a system for modeling simulation-ready plants [Zhao and Barbič 2013]. The deformation based editing system we implemented to test the proposed subspaces and force approximation follows [Hildebrandt et al. 2011].

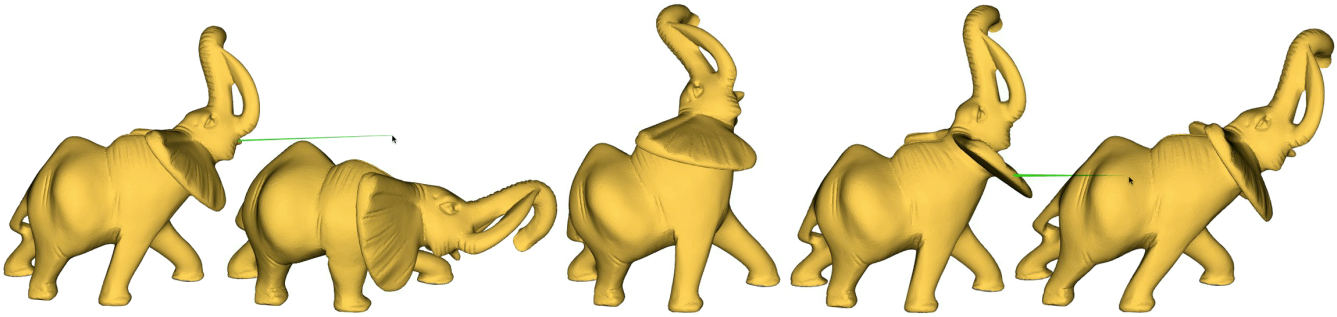

Figure 3: *User interaction with a real-time simulation of an elephant model (see Table 1).*

3 Background: Dimension Reduction

In this section, we briefly review the basics of model reduction for physical simulation and modeling of deformable objects and introduce our notation. For a recent tutorial on the subject, we refer to [Sifakis and Barbič 2012].

There are different physical models of deformable objects and various ways to discretize them. We keep our presentation general so that it covers a broad class of discrete deformable objects; the specific setting used for our experiments is treated in Section 7. We consider a discrete deformable object whose configuration space is $\mathbb{R}^{3n}$, *e.g.* a triangular or tetrahedral mesh whose states are described by listing the coordinates of all vertices. Let x_0 be a fixed state in $\mathbb{R}^{3n}$. We describe any deformation of x_0 by a displacement vector $u \in \mathbb{R}^{3n}$. The dynamics of a deformable object are described by the equations of motion

$$M\,\ddot{u}(t) = F(u(t), \dot{u}(t), t),$$

where F represents the acting forces and M is the positive definite and symmetric mass matrix. The forces F are a superposition of internal deformation forces $F^{int}(u(t))$ of the elastic shape, external forces $F^{ext}(u(t), \dot{u}(t), t)$, and damping forces $F^{damp}(u(t), \dot{u}(t))$.

Dimension reduction restricts the configuration space to a d-dimensional subspace V of $\mathbb{R}^{3n}$. The construction of such a space is treated in Section 5. Let $\{b_i\}_{i\in\{1,2,\ldots,d\}}$ be a basis of V, then the matrix $U = [b_1, b_2, ..., b_d] \in \mathbb{R}^{3n\times d}$ maps the reduced coordinates q in V onto the corresponding displacement vector $u \in \mathbb{R}^{3n}$,

$$u = U\,q.$$

The reduced equations of motion are

$$\bar{M}\,\ddot{q}(t) = \bar{F}(q(t), \dot{q}(t), t), \tag{1}$$

where

$$\bar{M} = U^T M U \text{ and } \bar{F}(q(t), \dot{q}(t), t) = U^T F(Uq(t), U\dot{q}(t), t)$$

are the reduced mass matrix and the reduced forces. In Section 4, we show how the reduced forces can be efficiently approximated.

Deformation-based editing Deformation-based editing allows a user to model a shape by first selecting arbitrary regions of the shape as handles and then translating and rotating them in space. The modeling system automatically computes shapes that follow the handles. This is realized by treating the shape as a deformable object and translating the user input into forces that act on the object. Static equilibrium states of the deformable object under the external forces are then computed. We assume that the internal forces are conservative and denote the potential (deformation energy) by $E(u)$. The external forces are Hookean zero-length springs that act on the handles and pull them towards a position indicated by the user. We denote the potential of the springs by $E^C(u)$ and set

$$\mathcal{E}(u) = E(u) + E^C(u).$$

The deformed shape is the minimizer

$$\arg\min_{u\in\mathbb{R}^{3n}} \mathcal{E}(u). \tag{2}$$

Since the geometries are often highly resolved, *e.g.* 3D-scan data, this is a high-dimensional nonlinear optimization problem. Model reduction can be used to get an interactive system for deformation-based editing, see [Hildebrandt et al. 2011]. In such system, the optimization problem (2) is replaced by the reduced problem

$$\arg\min_{q\in V} \mathcal{E}(Uq). \tag{3}$$

To get interactive response times, an efficient approximation of the energy and gradient in the reduced spaces is needed. In [Hildebrandt et al. 2011] a mesh coarsening approach was used. The force approximation proposed in next section is an alternative to this.

4 Force Approximation

Applying dimensional model reduction to complex, nonlinear systems can significantly improve the runtime performance. However, evaluating the reduced internal forces $\bar{F}^{int}$ by computing F^{int} and projecting it into the subspace becomes expensive when the mesh size increases. To make the computational costs independent of the size of the full problem, an efficient evaluation of the reduced internal forces is needed. The *optimized cubature*, proposed by An et al. [2008], constructs an approximation of the reduced forces based on a best subset selection problem. We follow this approach and set up a similar optimization problem in Equation (6). However, we propose a completely different strategy to solve the best subset selection problem in Section 4.1. A comparison of running times and accuracy of the proposed scheme and the greedy solver used in [An et al. 2008] on a set of different models and parameter settings can be found in Table 1.

We assume that the internal forces can be written as a sum

$$F^{int}(u) = \sum_{i=1}^{m} f_i^{int}(u),$$

where any f_i^{int} depends only on a local neighborhood, *e.g.* the four vertices for a tet. A wide range of discretizations, *e.g.* finite elements, represent $F^{int}(u)$ in such a form. Then, the reduced force

$\bar{F}^{int}$ satisifes

$$\bar{F}^{int}(q) = \sum_{i=1}^{m} U^T f_i^{int}(Uq) = \sum_{i=1}^{m} \bar{f}_i^{int}(q),$$

where $\bar{f}_i^{int}(q) = U^T f_i^{int}(Uq)$. The idea behind the approximation of $\bar{F}^{int}$ is to exploit the correlations between the $\bar{f}_i^{int}$s. We carefully select a subset of the $\bar{f}_i^{int}$s and assign a non-negative weight w_i to any selected $\bar{f}_i^{int}$. Then, the approximate reduced force is a linear combination of the selected $\bar{f}_i^{int}$s. Formally, we store the weights and indices of the selected $\bar{f}_i^{int}$s in a sparse m-dimensional vector w. An nonzero entry w_i in w, means that $\bar{f}_i^{int}$ is in the selection set and has the weight w_i. Then, we define

$$\bar{F}^{int}(q) \approx \tilde{F}_w^{int}(q) = \sum_{i \in \text{supp}(w)} w_i \bar{f}_i^{int}(q), \tag{4}$$

where $\text{supp}(w)$ denotes the support w, *e.g.* the set of indices with non-zero entries of w. The motivation for restricting to positive weights is that we want to preserve the structure of F^{int} and reduce the potential of overfitting the model function to the training data.

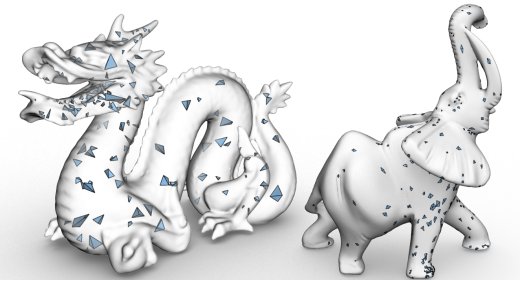

Figure 4: *Optimal set of components used for the approximation of the reduced forces. Left: tets in a solid model of the dragon; right: edge flaps in the shell model of the elephant (see Table 1).*

Determining a good subset of components and weights, is a non-trivial task for general, nonlinear materials and geometrically complex objects. We deal with this problem by using a data-driven approach that *learns* the best subset and weights from a set of T automatically generated training poses (and corresponding forces). Let $\{q_t\}_{t=1,\dots,T}$ be the set of training poses. Then, the relative fitting error of $\tilde{F}_w^{int}$ to the training forces is

$$\epsilon(w) = \sqrt{\frac{1}{T} \sum_{t=1}^{T} \frac{\epsilon_t(w)^2}{\left|\left|\bar{F}^{int}(q_t)\right|\right|^2}}, \tag{5}$$

where $\epsilon_t(w) = ||\bar{F}^{int}(q_t) - \tilde{F}_w^{int}(q_t)||$ is the absolute approximation error to $\bar{F}^{int}(q_t)$. The problem of finding a sparse approximation can now be stated in terms of an optimization problem in which we try to minimize the fitting error

$$\min_w \epsilon(w)^2 \quad \text{subject to } \; w \geq 0, |\text{supp}(w)| \leq s, \tag{6}$$

where $|\text{supp}(w)|$ denotes the cardinality of $\text{supp}(w)$. As $\epsilon(w)^2$ depends quadratically on w, (6) is a best subset selection problem with non-negativity constraints. This can be seen by reformulating the error as $\epsilon(w) = \frac{1}{\sqrt{T}}||Aw - b||$, where $A \in \mathbb{R}^{dT \times m}$ and $b \in \mathbb{R}^{dT}$ are defined by

$$A = \begin{bmatrix} \frac{\bar{f}_1^{int}(q_1)}{||\bar{F}^{int}(q_1)||} \cdots \frac{\bar{f}_m^{int}(q_1)}{||\bar{F}^{int}(q_1)||} \\ \vdots \quad \ddots \quad \vdots \\ \frac{\bar{f}_1^{int}(q_T)}{||\bar{F}^{int}(q_T)||} \cdots \frac{\bar{f}_m^{int}(q_T)}{||\bar{F}^{int}(q_T)||} \end{bmatrix}, b = \begin{bmatrix} \frac{\bar{F}^{int}(q_1)}{||\bar{F}^{int}(q_1)||} \\ \vdots \\ \frac{\bar{F}^{int}(q_T)}{||\bar{F}^{int}(q_T)||} \end{bmatrix}.$$

In the remainder of this section, we provide details of the individual steps needed to estimate an optimal subset, *i.e.*, a solution to (6). The complete optimization scheme is summarized in Algorithm 1.

4.1 Optimal subsets via sparse approximation

Best subset selection problems are ubiquitous in a variety of disciplines and much research has been devoted to their efficient solution. In particular, their presence in the field of Compressed Sensing received significant attention and triggered further advancements, one of them being the *normalized iterative hard thresholding (NIHT) algorithm* [Blumensath and Davies 2010].

Let $f(w) = T\epsilon(w)^2$ be the objective function. Then, an iteration of the NIHT algorithm is given by

$$w^{i+1} = \mathcal{H}_s(w^i - \mu^i \nabla f(w^i)), \tag{7}$$

where $\nabla f(w^i) = 2A^T(Aw^i - b)$ and $\mathcal{H}_s$ is a combinatorial projection that sets all but the s largest (in magnitude) entries of a vector to zero. Increased stability over the traditional hard thresholding algorithm is due to choosing the step length μ^i adaptively in every iteration. Before we proceed to the estimation of μ^i, we remark that the support of w^{i+1} will be necessarily contained in the $2s$-element set

$$\mathcal{S}^i = \text{supp}(w^i) \cup \text{supp}(\mathcal{H}_s(-\nabla_{\mathcal{I} \setminus \text{supp}(w^i)} f(w^i))). \tag{8}$$

Here $\nabla_{\mathcal{K}} f$ denotes the gradient of f with all entries not in the set $\mathcal{K}$ set to zero and $\mathcal{I} = \{1, \dots, m\}$. Based on S^i, the step length is computed to be

$$\mu^i = \frac{\left|\left|\nabla_{\mathcal{S}^i} f(w^i)\right|\right|^2}{||A \nabla_{\mathcal{S}^i} f(w^i)||^2}. \tag{9}$$

Despite its simplicity, under certain conditions, NIHT has a linear rate of convergence and can approximate sparse solutions with near optimal accuracy. However, NIHT can be further accelerated (see [Cevher 2011]). One of the most effective acceleration methods is the *hard thresholding pursuit (HTP)* by [Foucart 2011] which adds a second step to the NIHT algorithm. In every iteration, the projected gradient descent step in (7) is preceded by the low-dimensional minimization

$$w^{*,i+1} = \arg\min_{\{v | \text{supp}(v) \subseteq \text{supp}(w^{i+1})\}} f(v). \tag{10}$$

We propose a novel algorithm, called *non-negativity-constrained HTP* (NN-HTP) that extends HTP by an efficient treatment of non-negativity constraints. First, we define a new projection operator

Algorithm: NN-HTP
Data: f, w^0, tol
Result: x^*
$w^0 \leftarrow \mathcal{H}_s^+(w^0)$;
for $i = 1, 2, \dots$ **do**
  (Lazy) evaluate $\nabla f(w^i)$;
  if $\left|\left|\nabla f(w^i)\right|\right| \leq tol$ **then return** w^i;
  Determine $\mathcal{S}^i$ via (8) using $\mathcal{H}_s^+$;
  Determine μ^i via (9);
  $w^{i+1} \leftarrow \mathcal{H}_s^+(w^i - \mu^i \nabla_{\mathcal{S}^i} f(w^i))$;
  $\mathcal{X}^{i+1} \leftarrow \text{supp}(w^{i+1})$;
  if $\mathcal{X}^{i+1} = \mathcal{X}^i$ **then return** w^i;
  $w^{i+1} \leftarrow \arg\min_{\{v | \text{supp}(v) \subseteq \mathcal{X}^{i+1}, v \geq 0\}} f(v)$;
end
Algorithm 1: Non-negativity-constrained hard thresholding pursuit.

$\mathcal{H}_s^+(w)$ that projects w to the new feasible region, *i.e.* ignores all negative entries and keeps only the s largest ones. To satisfy the constraints, we replace the projection operator in (7) with $\mathcal{H}_s^+$. Accordingly, we account for the adjustments in (7) by applying $\mathcal{H}_s^+$ in (8) to ensure the correct support estimation.

The second step is to incorporate the non-negativity constraints in the s-dimensional minimization (10). Let $A' \in \mathbb{R}^{dT \times s}$ denote the submatrix of A containing only the columns corresponding to elements in $\text{supp}(w^i)$ and $w' \in \mathbb{R}^s$ the associated subvector of w^i. Then, we can determine the minimizer $w^{*,i+1}$ by solving the non-negative least squares (NNLS) problem $A'w' = b$ with $w' \geq 0$. In our experiments, these NNLS problems proved to be too stiff for projection-based solvers like L-BFGS-B [Morales and Nocedal 2011]. Therefore, we employ the efficient FNNLS solver by Bro and De Jong [1997], an active set solver that accelerates the solution of the normal equations by precomputing the matrix ${A'}^T A'$.

Lazy optimization In every iteration of our NN-HTP solver, at most s new components can enter the subset. Considering all remaining components as candidates comes at significant computational costs; especially since matrix A requires a storage size of $O(dTm)$ and is therefore evaluated only partially when needed.

To reduce the costs of the subset optimization, we introduce a parameter c that allows to specify the number of considered components. This parameter allows us to trade-off convergence speed against iteration costs of our NN-HTP solver. In every iteration, we randomly choose c of the remaining components and calculate only the corresponding columns of A. In fact, we found that considering only a subset can even increase the convergence of the algorithm. For our examples, we achieved the best results with $c \approx 5s$.

Automatic training pose generation Given the material and a subspace deformation model for a deformable object we can generate training poses automatically by randomly sampling points in the subspace. To acquire a material-aware sampling that roughly balances the ratio of configurations with high and low strain energy, we perform the sampling in the basis $\bar{\phi}_1, \bar{\phi}_2, \ldots, \bar{\phi}_d$ of vibrations modes in the reduced space. In particular, $q_t = \sum_i^d \text{rnd}(\bar{\omega}_i^{-1})\bar{\phi}_i$, where $\text{rnd}(\bar{\omega}_i^{-1})$ generates normally distributed random numbers with standard deviations proportional to the inverse of the frequency $\bar{\omega}_i$.

5 Subspace Construction

We propose a construction of subspaces for nonlinear simulations of deformable objects. We start with is a modal basis consisting of low-frequency vibration modes around a rest state of the elastic energy. Modal bases depend on the geometry and the material of the deformable object and are a standard tool for the model reduction of linear problems, *e.g.* if only small deformations occur. For deformable objects undergoing large deformations, these spaces cannot capture the nonlinearities. This is reflected in a large increase of strain energy (and visible artifacts) when such a state is projected onto the subspace. The effect is illustrated in Figure 9, in which editing results computed in different subspaces are shown. To compensate for these effects, we propose a strategy for enriching modal bases. The idea is to add new vectors to the basis that are obtained by applying certain linear transformations to the vectors in the modal basis. This provides the subspace with additional degrees of freedom that can compensate for artifacts which would appear in the space spanned by the modal basis. Our experiments show that the effect we achieve through this construction is comparable to that of enriching a modal basis with modal derivatives.

Modal bases Let us assume that x_0 is a static and stable equilibrium for some constant external force. If the object is excited (by a small enough stimulus), it vibrates around x_0. The vibration modes ϕ_i and frequencies ω_i can be computed by solving the sparse generalized eigenvalue problem

$$\omega_i^2 \, M \, \phi_i = K \, \phi_i,$$

where $K = \frac{\partial}{\partial u} F^{int}(0)$ is the tangential stiffness matrix. The modal bases we use as a starting point for our construction of a reduced space consists of the 15-20 eigenmodes with the lowest frequencies.

Extended modal bases In the continuous setting, an element of the modal basis is a vector field which assigns a vector in $\mathbb{R}^3$ to every point of the deformable object. Similarly, we can interpret a modal vector $\phi_i \in \mathbb{R}^{3n}$ as a discrete vector field consisting of n vectors in $\mathbb{R}^3$, *e.g.* a $\mathbb{R}^3$-vector at every vertex of a triangle or tet mesh. To construct a new vector field from a mode ϕ_i, we consider a linear map on $\mathbb{R}^3$ and apply it (the same linear map) to all n $\mathbb{R}^3$-vectors of the mode. For example, we fix an angle and an axis in $\mathbb{R}^3$ and rotate all the n vectors by the same angle about the same axis.

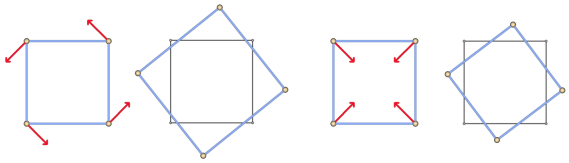

Figure 5: *Linearization of rotations (first image) introduce artifacts (second image). By rotating the vectors of the linearized rotation by 90° (third image), one obtains a vector field that can be used to compensate for the artifacts (fourth image).*

Let us look at the simple example shown in Figure 5. It illustrates that extending a basis using this construction can help to compensate for linearization artifacts. We consider a $2D$-object in the plane (a square in the figure) and a vector field pointing into the direction of a linearized rotation (around some center c). Moving the vertices of the object along the direction of the vector field, rotates the object, but at the same time produces an artifact: the volume of the object is increased. Now we construct a second vector field by rotating every vector of the linearized rotation by 90°. Then, any rotation of the object around c can be exactly described as a linear combination of the two vector fields.

Motivated by such examples, we propose using construction for extending modal bases. If a modal basis consists of r modes, then the extended basis can have at most $9r$ vectors, since there are only nine linear independent linear maps on $\mathbb{R}^3$. The scheme for extending a modal basis is outlined in Algorithm 2. In the first step, $9r$ vectors are constructed based on the r modes. Then, the Gram–Schmidt process is applied to get an orthonormal basis of the corresponding

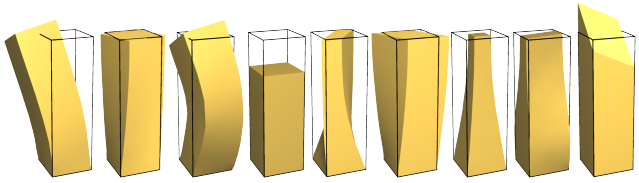

Figure 6: *Three linear modes (left) and for each of the three modes two vectors obtained through our basis extension are shown.*

reduced space. A basis in the space of linear maps on $\mathbb{R}^3$ is formed by the nine matrices B_{kl} that have only one non-vanishing entry, which takes the value 1. By applying each of the matrices B_{kl} to the n $\mathbb{R}^3$-vectors of a mode, we obtain the desired nine new $\mathbb{R}^{3n}$-vectors. The construction of all $9r$ vectors can be implemented in four nested for-loops. The first loop (index i in Algorithm 2) iterates over all modes, the second (index j) over all n $\mathbb{R}^3$-vectors of the ith mode, and the third and fourth (indices k and l) over the 9 matrices B_{kl}. The body of the inner loop multiplies the matrix B_{kl} by the jth $\mathbb{R}^3$-vector of the ith mode and inserts the result into the corresponding vector of the extended basis. This requires only one assignment since after a multiplication with B_{kl}, a $\mathbb{R}^3$-vector has only one non-zero component. The construction of the $9r$ vectors requires only $9rn$ assignments, and, therefore, is fast compared to the construction of the modal basis. In Algorithm 2, the $\mathbb{R}^{3n}$-vectors successively list the x-coordinates the n $\mathbb{R}^3$-vectors, then the n y-coordinates, and finally the z-coordinates.

Algorithm: Extend Modal Basis
Data: $\phi_1, \phi_2, \ldots, \phi_r \in \mathbb{R}^{3n}$
Result: $e_1, e_2, \ldots, e_d \in \mathbb{R}^{3n}$
allocate $b_1, b_2, \ldots, b_{9r} = 0 \in \mathbb{R}^{3n}$;
for $i = 1, 2, \ldots, r$ **do**
  for $j = 1, 2, \ldots, n$ **do**
    for $k = 1, 2, 3$ **do**
      for $l = 1, 2, 3$ **do**
        $b_{9(i-1)+3(k-1)+l}[(l-1)n+j] = \phi_i[(k-1)n+j]$;
      end
    end
  end
end
$\{e_i\} \leftarrow$ orthonormalize($\{b_i\}$);

Algorithm 2: Construction of the extended modal basis.

If the object is unconstrained, the first six eigenmodes span the translations and the linearized rotations of the object. In this case, we modify the construction. In the first step, we apply the extension strategy to the modes $\{\phi_7, \phi_8, \ldots, \phi_r\}$. In the second step, we add 12 basis vectors that span all affine transformations, *i.e.* linear transformations and translations, of the object to the basis. Together, the extended modal basis has maximum of $9r$ (constrained object) or $9(r-6)+12$ (unconstrained object) vectors. The number of elements of the basis can be smaller if the generated vectors are linearly dependent. Examples of vectors in the extended modal basis are shown in Figure 6.

There are also alternative schemes for computing a basis that spans the same reduced space. For example, one could use another basis of the space of linear maps on $\mathbb{R}^3$ to construct the $9r$ vectors. We chose to use the basis B_{kl} because the sparsity of the matrices B_{kl} reduces the number of required arithmetic operations.

6 Applications

Before a real-time simulation or interactive editing session starts, we compute a subspace and the selection set for the force approximation using our techniques.

Real-time simulation To simulate the dynamics of a reduced deformable object, the system (1) is numerically integrated over time. Due to the nonlinear forcing terms and the high-frequency components contained in our large-deformation model, explicit schemes can be hard to control as numerical stiffness causes the integrator to become unstable. Because stability is crucial for interactive applications, we opt for a second-order accurate, implicit Newmark integrator. In particular, we use the *average acceleration method* (see [Hughes 2000]), which is unconditionally stable and widely used in structural dynamics. Each timestep involves solving a d-dimensional nonlinear system of equations. In our experiments, we found a quasi-Newton solver that constructs an approximation of the inverse of the Jacobian on the fly during optimization to be very effective for this. Explicitly, we used a variant of *Broyden's method* (see [Dennis and Schnabel 1987]), which achieves superlinear convergence, but in each iteration merely requires the evaluation of the internal forces, a low-rank matrix update, and matrix-vector operations. Using the force approximation, all these operations can be performed at $O(d^2)$ cost. To achieve a warm start, we compute the inverse Jacobian at the rest state during the preprocess and use it as a preconditioner for the nonlinear system. An alternative to the quasi-Newton scheme would be to use a Newton-Raphson solver.

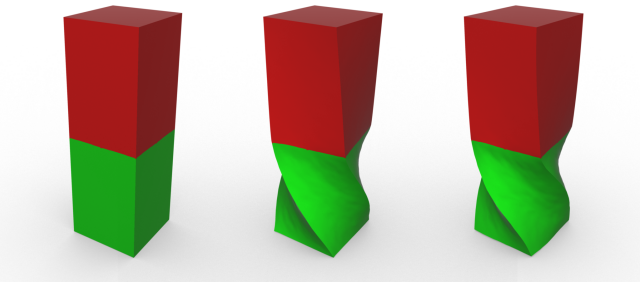

Figure 7: *Deformations of an object with inhomogeneous material. Left: initial state with color-coding of Young's modulus (red and green denote values of* 8×10^4 *and* 1×10^4*, respectively); middle: unreduced solution; right: solution in a 138-dimensional subspace.*

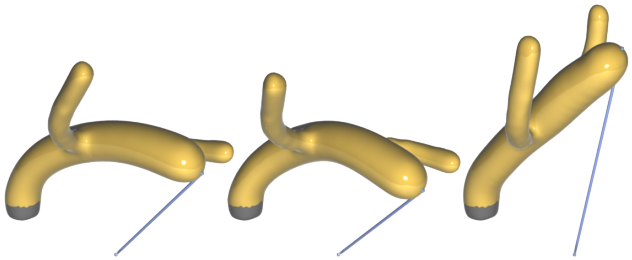

Figure 8: *Comparison of simulations of a spring pulling at a deformable object. From left to right: reference full simulation, simulation using the proposed space reduction and force approximation, results of a nonlinear simulation in a reduced space spanned by linear modes. Reduced spaces have the same dimensions.*

Deformation-based modeling In recent years, deformation-based modeling has attracted much attention due to its simple user-interface and the ability to directly model unstructured meshes, *e.g.* produced by 3D-scanners. In [Hildebrandt et al. 2011] an editing system based on a model reduction of the nonlinear optimization problem (2) was proposed. We follow this approach but replace the subspace construction and force approximation techniques.

Solving the reduced optimization problem (3) is related to finding the roots of the nonlinear system of equation arising from implicit Newmark integration. Since the Hessian of the reduced energy is the Jacobian of the d-dimensional system of equations $\frac{\partial}{\partial q}\bar{\mathcal{E}}(q) = 0$, it could be approximated using the same techniques from the previous paragraph. However, we would neglect important properties

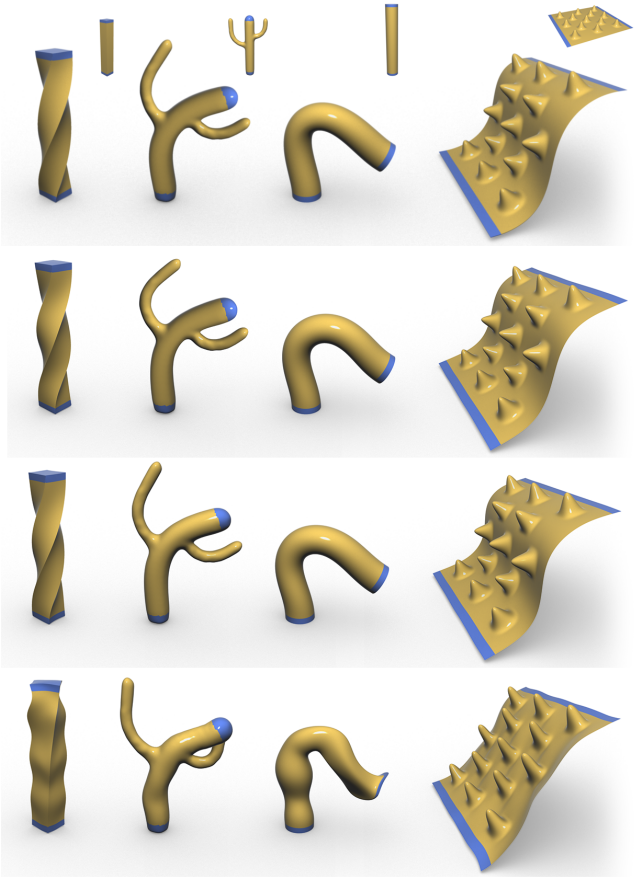

Figure 9: *Comparison of deformations found in 93-dimensional subspaces. Reference solution in unreduced space and rest states as superscripts (top row), our reduced space (second row), linear modes and modal derivatives (third row), only linear modes (bottom row).*

inherent to the minimization problem, *e.g.*, the Hessian of the energy is symmetric. An efficient quasi-Newton solver which incorporates these properties and again approximates the inverse Hessian (saving the costs of linear system solves) is the *BFGS method* (see [Dennis and Schnabel 1987]). As in Broyden's method, we can achieve a significant acceleration in convergence by providing the solver with the inverse Hessian at the rest state as a warm start.

Our modeling system allows a user to mark parts of the object as handles which, during runtime, can be translated and rotated in space by the user to prescribe modeling operations. Once the user defines those handle regions, we set up the control energy E^C. Let C_j be a randomly determined subset of vertex indices corresponding to the j-th handle. Then E^C is the quadratic energy

$$E^C(u) = \sum_j \sum_{i \in C_j} w_j (v_i - v_i')^2,$$

where w_j is a handle weight and v_i, v_i' are the current and prescribed positions of the i-th vertex. In our experiments, we chose the number of sampling points for the handle regions in the order of the cardinality of the selection set used for approximating the elastic forces.

The interior forces equal the negative of the gradient of the elastic potential E. Therefore, the force approximation can be used for fast gradient approximation. To approximate the elastic potential E, we use the same selection set and weights.

7 Results and Discussion

To test the proposed force approximation and subspace construction, we implemented the schemes for real-time simulation and deformation-based editing discussed in Section 6. We experimented with elastic solids and shells; in particular, we used finite elements discretizations of the St.Venant–Kirchhoff and Mooney–Rivlin material models of elastic solids and *Discrete Shells* [Grinspun et al. 2003]. For the simulations, we used (linear) Rayleigh damping.

The reduced internal forces of a single component (tet or edge flap) can be evaluated at $O(d)$ cost. As in [An et al. 2008], we observed that the number of components needed to achieve a given error tolerance grows linearly with the subspace dimension. Thus, our reduction achieves force approximation independently of the full shape space at $O(d^2)$ cost. Runtime statistics of reduced simulations for various geometries and parameter settings are listed in Table 1. This includes timings for the evaluation of the reduced forces and one step of an implicit Newmark integration. In addition, the resulting time-stepping rate of subspace integration (as average measured over 1k timesteps with constant external loads) is listed. Depending on the parameter settings, the reduced simulations achieve rates of 50-5000 implicit Newmark integration steps per second.

The total time needed to set up the reduced deformable models is dominated by the subspace construction, training force evaluation and subset optimization. Table 1 provides the times for each of the steps. Figure 4 visualizes optimal sets of components, *i.e.*, tets and edge flaps used to approximate the St.Vernant–Kirchhoff material and the discrete shells energy, respectively. Clearly, component selection is not based on purely geometric criteria. In particular, components tend to cluster in certain regions of the objects, *e.g.*, the jaw of the dragon and the right fore leg of the elephant; both regions proved to exhibit more versatile deformation behavior as compared with regions having fewer components.

Most of our experiments consider homogeneous materials. Still, since our construction of a reduced space includes a modal basis, it is material-aware. Figure 7 shows an example of a deformation of an inhomogeneous block made from two different materials. The reduced space adapts to the inhomogeneous material. For comparison, we show the unreduced (reference) solution.

To evaluate the fidelity of the reduced simulation, we measured the deviation of the reduced from the reference-full simulation for a dragon model with $92\,k$ tets. The average L^2-distance over all

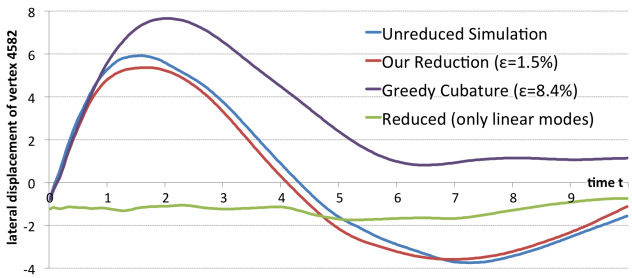

Figure 10: ***Runtime accuracy.*** *Lateral displacement of a vertex at the head of the dragon model (see Fig. 1 and Table 1) for the full and different reduced simulations.*

	Model	#Cmps	$d\,(r)$	U	T	b	s	$\tilde{F}_w^{int}$	Step	Steps/s	NN-HTP error	NN-HTP time	Greedy Cubature error	Greedy Cubature time
shell	Bunny	104310	135 (15)	26s	2k	125s	800	3.7ms	10.7ms	93	2.9%	593s	12.2%	2.2h
	Chinese dragon	389994	138 (20)	110s	1k	243s	250	1.5ms	4.9ms	203	7.4%	69s	17.4%	266s
	Dinosaur	168576	135 (15)	41s	1k	101s	500	3.0ms	7.9ms	126	2.3%	223s	11.0%	1293s
	Elephant	75000	135 (15)	18s	1k	46s	500	2.8ms	10.1ms	99	3.7%	205s	11.0%	1356s
	VW Beetle	52645	300 (300)	67s	2k	227s	800	9.7ms	20.6ms	49	2.8%	1473s	12.1%	2.7h
solid	Aorta*	35551	138 (20)	6s	1k	73s	200	1.0ms	4.4ms	245	2.1%	22s	10.9%	182s
	Dragon*	92386	135 (15)	10s	1k	182s	150	0.9ms	3.7ms	272	1.5%	63s	8.4%	130s
	Menger†	393216	135 (15)	36s	1k	332s	100	0.5ms	2.0ms	509	2.7%	8s	7.0%	116s
	Neptune†	691434	135 (15)	55s	1k	396s	200	0.8ms	2.4ms	424	2.8%	41s	25.6%	148s
	Skull*	156157	300 (300)	186s	1k	470s	250	3.5ms	9.6ms	104	2.3%	66s	8.4%	606s
	Vertebrae*	70447	15 (15)	7s	1k	117s	30	0.1ms	0.2ms	5059	1.5%	1s	7.6%	6s

Table 1: *Statistics measured on a 2012 MacBook Pro Retina 2.6GHz. From left to right: number of components, dimension of subspace (number of linear modes used), time for subspace construction, number of training poses, time for computation of training forces, posed cardinality constraints, time to evaluate force approximation, time for one step of implicit Newmark integration, time-stepping rate of subspace integration, achieved subset optimization error and optimization time for our scheme and the scheme from [An et al. 2008]. Experiments are based on Discrete Shells [Grinspun et al. 2003] and finite elements discretizations of the St.Venant–Kirchhoff (∗) and Mooney–Rivlin (†) material models. (3% error and max. 10 iterations for NN-HTP as stopping criteria for all runs except the example of the dragon, which converged to 1.5% after 18 iterations.)*

frames is listed in Table 2. Additionally, we show both simulations in the accompanying video and plot the lateral displacement of the mesh vertex with the largest initial velocity (a vertex at the side of the head) in Figure 10. The example illustrates that the proposed scheme is able to closely match the unreduced trajectory while featuring superior runtime performance—a more than 6000-fold increase in simulation speed after only about four minutes of preprocessing.

Comparison to Greedy Cubature An advantage of the proposed solver compared to the greedy strategy of An et al. [2008] is that the NN-HTP scheme reduced the number of iterations required to construct a selection set. The greedy strategy adds one component to the selection set in every iteration, hence, requires as many iterations as the cardinality of the selection set. In contrast, the NN-HTP builds a complete selection set in the first iteration and then updates this set at every iteration. In our experiments, we typically needed ten or less iterations to construct the selection set. As a consequence the number of NNLS problems that need to be solved to construct a selection set decreases. In addition, we use the FNNLS routine to solve the NNLS problems, which we found more effective than the Lawson–Hanson algorithm, see [Lawson and Hanson 1974], used by An et al. [2008]. We want to remark that, independent from our work, Kim and Delaney [2013] proposed a modified greedy cubature construction that reduces the number of iterations and uses FNNLS.

Table 1 provides a comparison of our optimization algorithm (NN-HTP) to the greedy optimization approach by [An et al. 2008] (using the implementation by Steven An with enabled parallelization). For the comparisons we used St.Venant–Kirchhoff and Mooney–Rivlin materials and Discrete Shells. The results illustrate the advantages of NN-HTP: it produces a significantly smaller approximation error in considerably less time. Furthermore, it is able to achieve a given training error with a smaller selection set yielding force approximations with considerably higher runtime performance. In particular, Figure 11 extends the elephant experiment from Table 1 by showing approximation errors as a function of the sparsity constraints. Greedy cubature reaches 3.7% training error with 1474 sparsity constraints (almost three times as many as needed by NN-HTP). Tests of the greedy algorithm were performed using runtime-favoring configurations as reported in [An et al. 2008] ($|\mathcal{C}| = 100, T_s = 10$). However, as shown in Figure 12,

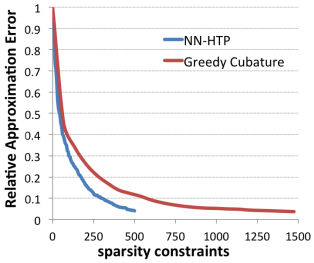

Figure 11: *Achieved subset optimization error with increasing sparsity constraints for NN-HTP and the Greedy Cubature by [An et al. 2008]. This plot shows optimization results for the elephant shell model in a 135 dim. subspace* (cf. *Table 1 for* $s = 500$).

disabling the *subset training* only slightly improves the approximation error from 11.0% to 8.5% but significantly increases runtime. On the other hand, NN-HTP is able to find a solution with 2.3% error in only ten iterations. The benefit of a more accurate force approximation for simulation is illustrated in the accompanying video and Figure 10. The video contains a sequence that compares the full simulation of the dragon model with reduced simulations using force approximations computed by our method and by greedy cubature with the same cardinality constraints (for details see Table 1). Furthermore, the figure shows the lateral displacement of the mesh vertex with the largest initial velocity for the different simulations.

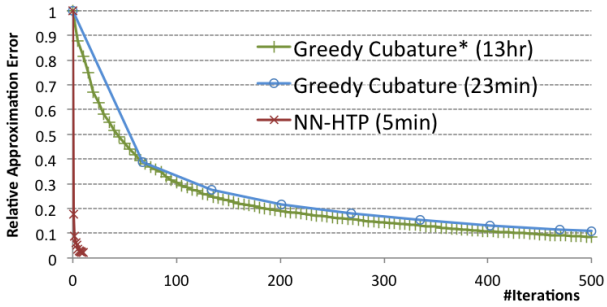

Figure 12: *Comparison between our NN-HTP and Greedy Cubature [An et al. 2008] for the dinosaur model (see Table 1). (∗) Disabling subset training in greedy solver increases runtime immensely with only marginal improvements in convergence.*

Comparison to modal derivatives To evaluate how well our extended modal bases can represent large deformations, we compare simulation and editing results obtained different subspaces: modal bases, extended modal bases, and modal bases augmented with modal derivatives. For the examples of static states shown in Figure 9 and the simulation of a dragon models shown in Figure 1, Table 2 lists the relative L^2-error, *i.e.* the L^2-norm of the difference of the reduced and the unreduced reference solutions divided by the area/volume of the rest state. The configurations that define the four static states are taken from [Botsch and Sorkine 2008], where they were tested using different linear editing schemes, *e.g.* based on linearized thin shells. None of the linear schemes could deal with all four poses without developing visible artifacts.

Model	#v	d	Our	M. D.	Linear
Bar	6k	93	0.0109	0.0113	0.0851
Cactus	5k	93	0.0131	0.0057	0.0933
Cylinder	5k	93	0.0089	0.0080	0.0565
Plate	40k	93	0.0109	0.0136	0.0266
Dragon	26k	135	0.0170	0.0225	0.0440

Table 2: ***Subspace Fidelity***. *From left to right: number of vertices, subspace dimension, L^2-norms of the differences between the unreduced solution to those obtained in subspaces from our construction, modal derivatives, and linear modes only. See Fig. 9 for shell examples and Fig. 1 for snapshots from the simulation of the dragon.*

For the comparisons, we used subspaces of the same dimension. In most cases, we used 15 eigenmodes to construct the extended modal basis. This yields 93- or 135-dimensional spaces depending on whether the object is constrained or unconstrained, see Section 5. For some unconstrained objects, we used 20 eigenmodes, resulting in 138-dimensional subspaces. To construct subspaces of equal dimension using modal derivatives, we start with the same number r of linear modes, compute all modal derivatives, and choose the required number of basis vectors from the span of the computed derivatives. For $r = 15$ and an unconstrained object, all modal derivatives are used.

For all examples, our construction and spaces enriched with modal derivatives yield better approximations of the unreduced reference solution than the spaces constructed only from linear modes. The results obtained with our construction and with modal derivatives are of comparable approximation quality. We want to remark that this comparison is limited to hyperelastic, isotropic, and homogeneous materials and the ratio of linear modes and modal derivatives described above.

Model	#v	d (r)	Φ	Our	M. D.
Chinese Dragon	130k	138 (20)	102s	8s	407s
Dinosaur	56k	135 (15)	38s	3s	169s
Elephant	25k	135 (15)	17s	1s	81s
Dragon	26k	135 (15)	9s	1s	28s
Vertebrae	16k	135 (15)	7s	1s	21s

Table 3: ***Subspace construction times***. *From left to right: number of vertices, subspace dimension (number of used vibration modes), time for computation of vibration modes, our extended modes and modal derivatives.*

We found the results promising, since compared to modal derivatives, the proposed basis extension is easier to implement and faster to execute. To illustrate the lower computational costs, we list computation times for the construction of the linear modes, our extended modes, and the modal derivatives in Table 3. Of course, these times may vary depending on what solvers are used for the computation of the eigenmodes and the modal derivatives. To compute the eigenmodes, we use our implementation of the shift-and-invert Lanczos method. Since the computation of the modal derivatives involves solving a number of linear systems with the same matrix, we use a sparse direct solver for the systems (reusing the factorization). Additionally, we set up the right-hand sides of the systems in parallel.

Deformation-based editing The deformation-based editing system we implemented follows [Hildebrandt et al. 2011]. The techniques for subspace generation and reduced force approximation used therein, can be directly replaced by the techniques proposed here. In [Hildebrandt et al. 2011], modal derivatives are used for the construction of the subspace and the preprocessing costs are dominated by the subspace construction. Hence, using the extended modal bases instead, reduces the preprocessing time of the method (as discussed and analyzed above).

The editing method focuses on global edits. Local edits are only supported to a certain degree, depending on what can be represented in the subspace. Creating a method that integrates local and global edits, *e.g.* by extending the subspace basis on the fly, is still an open problem. For reduced simulations, a method that extends a reduced space during runtime, *e.g.* to represent collisions more accurately, was recently proposed by Harmon and Zorin [2013]. For interactive shape editing, there are many alternative schemes to [Hildebrandt et al. 2011] (see Section 2). A through comparison of the different methods, however, lies beyond the scope of this paper.

8 Conclusion and Future Work

We propose model reduction techniques for the approximation of the reduced forces and for the generation of subspaces for deformable objects. We demonstrate their effectiveness in real-time simulations and deformation-based editing and compare them to alternative techniques.

Beyond the discussed applications, we will use our model reduction techniques to accelerate the integration of large simulations. Such a scheme must satisfy an error tolerance and therefore needs to modify the reduced space and force approximation if the tolerance cannot be achieved with the current reduced model. We expect benefits from both techniques presented in this paper for this application. Our construction of a reduced basis, which is fast and supports large deformations, could be used to accelerate adjusting or recomputing the reduced space. Furthermore, our NN-HTP solver could incrementally update the force approximation using the previous approximation as a starting point.

Another direction for future work is to develop fast solvers for other physical systems, or, more generally, to other problems that are modeled with complex systems of equations based on the proposed techniques.

Acknowledgements

We would like to thank Steven S. An for sharing his *cubature* source code and the anonymous reviewers for their comments and suggestions. This work was supported by the Max Planck Center for Visual Computing and Communication (MPC-VCC) and the DFG Research Center MATHEON "Mathematics for Key Technologies".

References

AN, S. S., KIM, T., AND JAMES, D. L. 2008. Optimizing cubature for efficient integration of subspace deformations. *ACM Trans. Graph. 27*, 5 (Dec.), 165:1–165:10.

BARBIČ, J., AND JAMES, D. L. 2005. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. Graph. 24*, 3 (July), 982–990.

BARBIČ, J., AND POPOVIĆ, J. 2008. Real-time control of physically based simulations using gentle forces. *ACM Trans. Graph. 27*, 5 (Dec.), 163:1–163:10.

BARBIČ, J., DA SILVA, M., AND POPOVIĆ, J. 2009. Deformable object animation using reduced optimal control. *ACM Trans. Graph. 28*, 53:1–53:9.

BARBIČ, J., SIN, F., AND GRINSPUN, E. 2012. Interactive editing of deformable simulations. *ACM Trans. Graph. 31*, 4, 70:1–70:8.

BEN-CHEN, M., WEBER, O., AND GOTSMAN, C. 2009. Variational harmonic maps for space deformation. *ACM Trans. Graph. 28*, 3 (July), 34:1–34:11.

BLUMENSATH, T., AND DAVIES, M. 2010. Normalized iterative hard thresholding: Guaranteed stability and performance. *IEEE J. Sel. Topics Signal Process. 4*, 2 (April), 298 –309.

BOTSCH, M., AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Trans. Vis. Comput. Graphics 14*, 1 (Jan.), 213–230.

BOTSCH, M., PAULY, M., WICKE, M., AND GROSS, M. 2007. Adaptive space deformations based on rigid cells. *Computer Graphics Forum 26*, 3, 339–347.

BOTSCH, M., KOBBELT, L., PAULY, M., ALLIEZ, P., AND LEVY, B. 2010. *Polygon Mesh Processing*. AK Peters.

BRO, R., AND DE JONG, S. 1997. A fast non-negativity-constrained least squares algorithm. *Journal of Chemometrics 11*, 5, 393–401.

CEVHER, V. 2011. On Accelerated Hard Thresholding Methods for Sparse Approximation. In *Wavelets and Sparsity XIV*, vol. 8138.

CHADWICK, J. N., AN, S. S., AND JAMES, D. L. 2009. Harmonic shells: a practical nonlinear sound model for near-rigid thin shells. *ACM Trans. Graph. 28*, 5 (Dec.), 119:1–119:10.

CHOI, M. G., AND KO, H.-S. 2005. Modal warping: Real-time simulation of large rotational deformation and manipulation. *IEEE Trans. Vis. Comput. Graphics 11*, 1 (Jan.), 91–101.

DENNIS, J., AND SCHNABEL, R. 1987. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Classics in Applied Mathematics. SIAM.

FOUCART, S. 2011. Hard thresholding pursuit: An algorithm for compressive sensing. *SIAM J. Numer. Anal. 49*, 6, 2543–2563.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete Shells. *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 62–67.

HAHN, F., MARTIN, S., THOMASZEWSKI, B., SUMNER, R., COROS, S., AND GROSS, M. 2012. Rig-space physics. *ACM Trans. Graph. 31*, 4 (July), 72:1–72:8.

HARMON, D., AND ZORIN, D. 2013. Subspace integration with local deformations. *ACM Trans. Graph. 32*, 4, 107:1–107:10.

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2011. Interactive surface modeling using modal analysis. *ACM Trans. Graph. 30* (October), 119:1–119:11.

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2012. Interactive spacetime control of deformable objects. *ACM Trans. Graph. 31*, 4 (July), 71:1–71:8.

HUANG, J., TONG, Y., ZHOU, K., BAO, H., AND DESBRUN, M. 2011. Interactive shape interpolation through controllable dynamic deformation. *Trans. Vis. Comput. Graph. 17*, 7, 983–992.

HUGHES, T. 2000. *The finite element method: linear static and dynamic finite element analysis*. Dover Publications.

IDELSOHN, S. R., AND CARDONA, A. 1985. A reduction method for nonlinear structural dynamic analysis. *Computer Methods in Applied Mechanics and Engineering 49*, 3, 253 – 279.

JACOBSON, A., BARAN, I., KAVAN, L., POPOVIĆ, J., AND SORKINE, O. 2012. Fast automatic skinning transformations. *ACM Trans. Graph. 31*, 4 (July), 77:1–77:10.

KIM, T., AND DELANEY, J. 2013. Subspace fluid re-simulation. *ACM Trans. Graph. 32*, 4 (July), 62:1–62:9.

KIM, T., AND JAMES, D. L. 2009. Skipping steps in deformable simulation with online model reduction. *ACM Trans. Graph. 28*, 123:1–123:9.

KRYSL, P., LALL, S., AND MARSDEN, J. E. 2001. Dimensional model reduction in non-linear finite element dynamics of solids and structures. *Int. J. Numer. Meth. Eng. 51*, 4, 479–504.

LAWSON, C. L., AND HANSON, R. J. 1974. *Solving Least Square Problems*. Prentice Hall.

MEYER, M., AND ANDERSON, J. 2007. Key point subspace acceleration and soft caching. *ACM Trans. Graph. 26*, 3 (July).

MORALES, J. L., AND NOCEDAL, J. 2011. Remark on "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization". *ACM Trans. Math. Softw. 38*, 1, 7:1–7:4.

PENTLAND, A., AND WILLIAMS, J. 1989. Good vibrations: modal dynamics for graphics and animation. *SIGGRAPH Comput. Graph. 23*, 3 (July), 207–214.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. Graph. 23*, 514–521.

SIFAKIS, E., AND BARBIČ, J. 2012. FEM simulation of 3D deformable solids: A practitioner's guide to theory, discretization and model reduction. In *SIGGRAPH Courses*, 20:1–20:50.

SUMNER, R. W., SCHMID, J., AND PAULY, M. 2007. Embedded deformation for shape manipulation. *ACM Trans. Graph. 26*, 3.

TISO, P. 2011. Optimal second order reduction basis selection for nonlinear transient analysis. In *Modal Analysis Topics*, vol. 6. Springer, 27–39.

TREUILLE, A., LEWIS, A., AND POPOVIĆ, Z. 2006. Model reduction for real-time fluids. *ACM Trans. Graph. 25*, 826–834.

WICKE, M., STANTON, M., AND TREUILLE, A. 2009. Modular bases for fluid dynamics. *ACM Trans. Graph. 28*, 39:1–39:8.

ZHAO, Y., AND BARBIČ, J. 2013. Interactive authoring of simulation-ready plants. *ACM Trans. Graph. 32*, 4, 84:1–84:12.