Interactive Surface Modeling using Modal Analysis

Klaus Hildebrandt, Christian Schulz, Christoph von Tycowicz, and Konrad Polthier
Freie Universität Berlin

We propose a framework for deformation-based surface modeling that is interactive, robust and intuitive to use. The deformations are described by a non-linear optimization problem that models static states of elastic shapes under external forces which implement the user input. Interactive response is achieved by a combination of model reduction, a robust energy approximation, and an efficient quasi-Newton solver. Motivated by the observation that a typical modeling session requires only a fraction of the full shape space of the underlying model, we use second and third derivatives of a deformation energy to construct a low-dimensional shape space that forms the feasible set for the optimization. Based on mesh coarsening, we propose an energy approximation scheme with adjustable approximation quality. The quasi-Newton solver guarantees superlinear convergence without the need of costly Hessian evaluations during modeling. We demonstrate the effectiveness of the approach on different examples including the test suite introduced in [Botsch and Sorkine 2008].

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Physically based modeling*

Additional Key Words and Phrases: geometric modeling, deformation-based modeling, interactive modeling, surface editing, geometric optimization, model reduction, modal derivatives.

1. INTRODUCTION

In recent years, a special focus in geometric modeling has been on schemes for deformation-based surface editing. A major advantage of such schemes over traditional modeling techniques like NURBS or subdivision surfaces is that typical modeling operations can be described by few constraints. This allows for efficient and simple click-and-drag user interfaces. To provide intuitive usability, the computed deformations must be physically meaningful to match the user's intuition and experience on how shapes deform. This leads to non-linear optimization problems that, to achieve interactivity, have to be solved within fractions of a second. The surfaces to be edited are often rich in detail and thus of high resolution. We distinguish between local and global deformations. Local deformations are restricted to a small area with the rest of the surface fixed. The resulting optimization problems are of small scale, still, it is challenging to solve them at interactive rates. The focus of this work is on global shape deformations which lead to optimization problems that, due to their size, cannot be solved at interactive rates. Since interactivity is indispensable, the challenge is to design methods that find as-good-as-possible approximations at low computational cost.

Instead of manipulating the surface directly, recent schemes for global interactive deformation-based editing manipulate a part of the ambient space that surrounds the surface and therefore implicitly edit the surface. The deformations of the space are often induced by a cage, which in turn is manipulated by a deformation-based editing scheme. The advantage of this concept is that the size of the optimization problem now depends on the resolution of the cage and is independent of the resolution of the surface.

Contributions. The contribution of this paper is a framework for deformation-based surface modeling that is interactive, robust, intuitive to use, and works with various surface deformation energies. The main idea is to reduce the complexity of the optimization problem in two ways: by using a low-dimensional shape space and by approximating the energy and its gradient and Hessian. The motivation for the space reduction is the observation that a modeling session typically requires only a fraction of the full shape space of the underlying model. We construct a reduced shape space as the linear span of two sets V_1 and V_2 of vectors. The set V_1 comprises the eigenvectors of the Hessian of the deformation energy that correspond to the lowest eigenvalues. The span of these vectors contains the deformations that locally cause the least increase of energy and is therefore well-suited to generate small deformations. However, large deformations in span(V_1) often develop artifacts and thus have high energy values. To improve the representation of large deformations in the reduced space, we collect in the set V_2 vectors that at points in span(V_1) point into energy descent directions. These directions are constructed using the third order term of a Taylor series of the Newton descent direction of the deformation energy. For the approximation of the energy and its derivatives, we propose a scheme based on a second reduced shape space for a simplified mesh. By construction, the two reduced shape spaces are isomorphic and we can use the isomorphism to pull the energy from the shape space of the simplified mesh to the shape space of the full mesh. Altogether, the resulting reduced problem is independent of the resolution of the mesh and our experiments show typical modeling operations, including large deformations, that are reasonably well approximated. To solve the reduced optimization problem, we use a quasi-Newton method that maintains an approximation of the inverse of the Hessian and generates descent directions at the cost of gradient evaluations while still producing superlinear convergence.

Our approach is an alternative to space deformation schemes. Space deformations are controlled by some object, *e. g.*, a cage around the surface (other objects like a volumetric mesh or a skeleton have been used as well). Then, the set of possible deformations of the surface depends on the cage and a space warping scheme. In contrast, our method does not depend on an artificial cage and a space warping scheme, but the subspaces we consider depend on geometric properties of the surface. A resulting advantage of our approach is that we do not need to deal with interpolation artifacts that many space warping schemes create. Furthermore, our scheme does not need to construct a cage, which often is a manual process. Instead the preprocess of our scheme is automatic and the computed basis can be stored on a hard disc with the surface. The subspaces our method produces are effective: we demonstrate that even 67-dimensional spaces can produce good approximations for typical modeling operations. In contrast, the coarsest cages that are used for space deformation have 200-500 vertices, hence generate a 600-1500 dimensional shape space. In addition, our approach is flexible. We can use the same energies as in the unreduced case, like PriMo [Botsch et al. 2006] or as-rigid-as-possible [Sorkine and Alexa 2007]. The approximation quality of the energy is adjustable and the size of the reduced space can be increased. Increasing both parameters will lead to the exact solution of the unreduced prob-

Fig. 1. Linear vibration modes and modal derivatives of the discrete shells energy on the dragon model. Figure shows: the rest state (top left), two linear modes (left), and two modal derivatives (right).

lem. Both parameters, quality of energy approximation and size of reduced space are independent.

The reduced bases we consider allow for a physical interpretation. If we regard the deformation energy as a potential energy of elastic deformations of the surface away from a rest shape, then the eigenvectors of the Hessian are the linear free vibration modes of the rest shape. Furthermore, the additional basis vectors, which form the set V_2, can be interpreted as simplified modal derivatives. Therefore, our method is linked to techniques from real-time physical simulations, especially to the work of Barbič and James [2005], in which a reduced basis consisting of vibration modes and modal derivatives is used for real-time simulation of the dynamics of St. Vernant-Kirchhoff deformable bodies. However, there are fundamental differences between the schemes. Whereas dynamic simulations require the integration of systems of ODEs, we solve an optimization problem and therefore need a completely different solver. In real-time physical simulations the approximation of forces is done using a cubic polynomial for each component of the force in the reduced space. This means that the number of coefficients required to represent the forces grows with $\mathcal{O}(r^4)$, where r is the dimension of the reduced space. In contrast, our energy approximation technique is independent of the size of the reduced space. In addition, we consider deformation energies defined for surface meshes (*e. g.* elastic thin shells), whereas [Barbič and James 2005] simulate elastic bodies using tetrahedral meshes.

2. RELATED WORK

Deformation-based modeling of surfaces describes shape editing operations relative to an initial surface, *e. g.*, a surface generated by a 3d-scanner. Such methods are driven by a deformation energy, *i. e.*, a function on a shape space of surfaces that, for every surface in the shape space, provides a quantitative assessment of the magnitude of the deformation of the surface from the initial surface. Methods for deformation-based modeling can be classified in three categories: linear, non-linear, and space deformation schemes. In addition, our work is linked to dimension reduction in physical simulations and modal analysis in geometry processing.

Linear surface modeling. Linear methods for surface modeling employ a quadratic deformation energy. Such energies are based on linearized thin shells, or, alternatively, on Laplacian coordinates or differential coordinates. Energies based on Laplacian coordinates assess the magnitude of a deformation of a mesh from an initial mesh by summing up the (squared norms of the) deviations of the local Laplace coordinates (which can be seen as discrete mean curvature vectors) at all vertices. The recent survey [Botsch and Sorkine 2008] provides a detailed overview of linear approaches and includes a comparison of various schemes. The main advantage of linear methods is that the minimization problem to be solved is comparably simple. For example, if the constraints are also modeled as a quadratic energy, as *e. g.* in [Lipman et al. 2004; Sorkine et al. 2004; Nealen et al. 2005], the deformed surface can be computed by solving a sparse linear system. These methods are designed for small deformations around the initial surface and often produce unintuitive results for large deformaions.

Non-linear surface modeling. Physical models of elastic shells are strongly non-linear and discretizations yield stiff discrete energies. The resulting optimization problems are challenging, especially if real-time solvers are desired. The non-linear PriMo energy [Botsch et al. 2006; Botsch et al. 2007] aims at numerical robustness and physically plausible solutions. The idea is to extended the triangles of a mesh to volumetric prisms, which are coupled through elastic forces. During deformations of the mesh the prisms are transformed only rigidly, which increases the robustness of the energy since the prisms cannot degenerate. As an alternative to elastic shells, non-linear methods based on Laplacian coordinates have been proposed. One idea, which can be found in several approaches, is to measure the change of the length of the Laplacian coordinates instead of measuring the change of the full vector. Dual Laplacian editing [Au et al. 2006] iteratively solves quadratic problems and after each iteration rotates the prescribed Laplacian coordinates to match with the surface normal directions of the current iterate. Huang et al. [2006] describe the prescribed Laplacian coordinates at each vertex in a local coordinate system that is used to update the direction of the prescribed coordinates. Pyramid coordinates [Kraevoy and Sheffer 2006] can also be seen as non-linear rotation-invariant Laplacian coordinates. For any vertex v there is a rotation that minimizes, in a least squares sense, the distance between the 1-ring of v on the initial and on the actual surface. The *as-rigid-as-possible* energy [Sorkine and Alexa 2007] is a weighted sum of these minima over all vertices. Recently, Chao et al. [2010] proposed to use the *distance between the differential of a deformation and the rotation group* as a principle for a geometric model for elasticity. This model includes a material model with standard elastic moduli (Lamé parameters) and is connected to the Biot strain of mechanics. The connection of this model of elasticity to energies used in geometric modeling, like the as-rigid-as-possible energy, opens the door to an analysis of the link of these energies and the Biot strain. The drawback of using non-linear energies for surface modeling is that directly solving the resulting minimization problem is costly, thus interactive performance is limited by the size of the meshes.

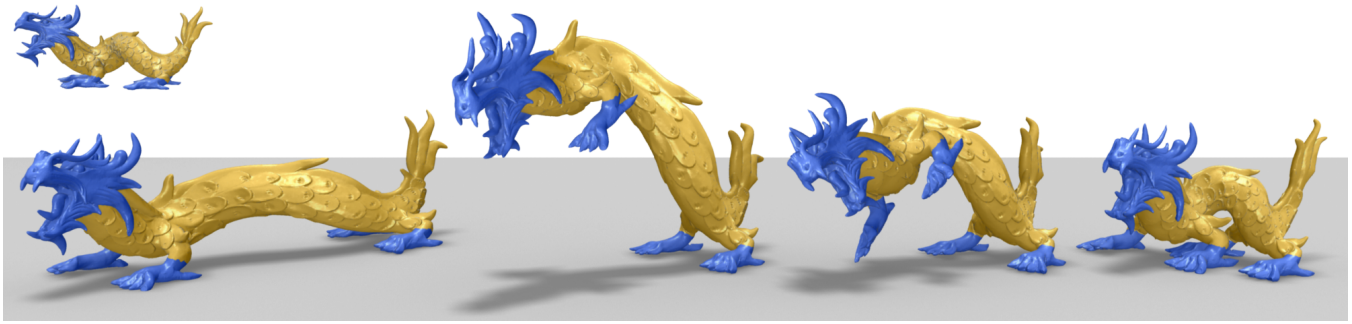


Fig. 2. Large deformations of the dragon model (130k vertices) computed by our modeling framework in a 130-dimensional shape space using a 1k ghost.

Space deformation. Instead of deforming the surface directly, the idea of space deformation methods is to deform the ambient space around the surface and therefore implicitly also the surface. To control the space deformations, a cage is built around the surface. The interior of the cage is described by a boundary representation, *i. e.*, every point in the interior of the volume is represented by coordinates that relate to the vertices of the cage. When the cage is deformed, the coordinates assign a new location to every point inside the cage. Some of the different boundary representations that have been proposed for this purpose are: mean value coordinates [Ju et al. 2005], harmonic coordinates [Joshi et al. 2007], and Green coordinates [Lipman et al. 2008; Ben-Chen et al. 2009]. The advantage of space deformations is that the complexity of the cage is independent of the resolution of the surface. Since it is inconvenient to model the cage directly, the cage can be modeled by a surface deformation scheme [Huang et al. 2006]. Alternatively, space deformations can be induced by a skeleton [Shi et al. 2007], a volumetric mesh [Botsch et al. 2007], or a graph structure [Sumner et al. 2007; Adams et al. 2008].

Dimension reduction in physical simulations. Closely related to deformation-based modeling is the physical simulation of elastic bodies; we discuss aspects of the interrelation of the two topics at the end of Section 3. Dimension reduction is an established technique in physical simulation [Pentland and Williams 1989; Krysl et al. 2001], that reduces the computational cost of a simulation. Linear modal analysis [Pentland and Williams 1989; Hauser et al. 2003; Choi and Ko 2005] can be used to automatically generate reduced coordinates that are well-suited to approximate small deformations away from the rest pose. To improve the approximation quality for larger deformations Barbič and James [2005] extend the basis of linear modes by simplified modal derivatives. In order to get real-time rates for simulations, in addition to the dimension reduction, the cost for evaluation of the forces has to be reduced. This can be achieved by representing or approximating (depending on the system to be simulated) the reduced forces by cubic polynomials on the reduced space [Barbič and James 2005; An et al. 2008; Barbič and Popović 2008]. The coefficients of this polynomial can be precomputed, which makes the cost for evaluating the forces independent of the size of the full system. All above-cited papers simulate solid bodies and therefore use volumetric meshes. Surface models are, for example, used for cloth simulation, where thin shells are applied to model cloth with folds and wrinkles. Common discrete models [Baraff and Witkin 1998; Bridson et al. 2003; Grinspun et al. 2003; Garg et al. 2007] measure the bending of the surface at the edges of the mesh. Such a discrete energy is given as a sum of contributions from stencils that consist of only two triangles sharing an edge. This reduces the complexity of the expressions for the energy and its derivatives, which in turn accelerates the evaluation and simplifies the implementation.

Modal analysis in geometry processing. The spectrum and the modes of the Laplace-Beltrami operator of a surface proved to be useful for various applications in geometry processing and computer graphics. An overview of this development can be found in the recent survey by Zhang et al. [2010] and in the course notes of a *Siggraph Asia 2009* course held by Lévy and Zhang [2009]. In addition, the spectrum of the Hessian of surface deformation energies has been investigated: Huang et al. [2009] use eigenmodes of the Hessian of the as-rigid-as-possible energy to construct physically meaningful segmentations of surfaces, and Hildebrandt et al. [2010] design surface signatures based on eigenmodes of the Hessian of the discrete shells energy. In general, the vibration modes of a curved surface differ significantly from eigenmodes of the Laplacian. In our experiements, low-frequency vibration modes of many of our test models tend to focus on some area, *e. g.*, the dragon model has vibration modes that move fore- or hind legs and keep the rest of the body almost fixed; in contrast, Laplace modes tend to distribute the vibration equally over the whole surface.

3. DEFORMATION-BASED MODELING

In this section, we describe the basis of our modeling framework and derive the full non-linear optimization problem that defines the deformation. At the end of the section, we discuss the connection of our framework to physical simulations of elastic shapes. A major ingredient to deformation-based modeling schemes is the deformation energy. Our method can work with any energy that is defined for a shape space of meshes and has continuous third derivatives at the reference mesh. However, the quality of our energy approximation scheme is directly connected to the insensitivity of the energy against simplification of the mesh. In our experiments, we used two different energies: the discrete shells and the as-rigid-as-possible energy. We start the section with a brief review of thin shell energies and of discrete shells.

Thin shell energies. We consider a homogeneous and isotropic thin shell that has a constant thickness over a surface $\bar{\Sigma}$, the so-called middle surface. Under certain assumptions, including the Kirchhoff-Love assumption, elastic deformations of the shell can be described by geometric properties of the middle surface, cf. [Terzopoulos et al. 1987; Ciarlet 2000]. The energy of such a shell is

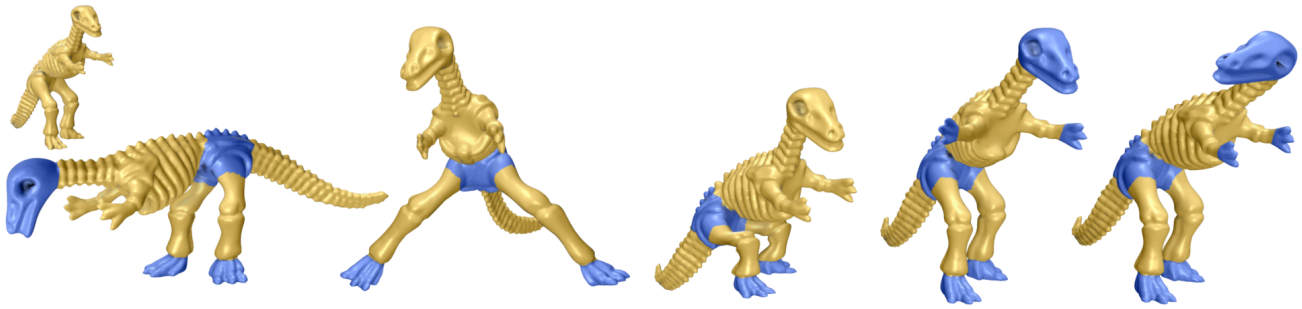


Fig. 3. Large deformations of the dino model (56k vertices) computed by our method using the as-rigid-as-possible energy as the objective functional. We use a 130-dimensional shape space (20 linear modes and 110 modal derivatives) and a ghost with 1k vertices.

given by

$$E(\Sigma) = \frac{1}{2}\int_{\bar{\Sigma}} \left(\|\mathrm{g} - \bar{\mathrm{g}}\|_\alpha^2 + \|\mathrm{h} - \bar{\mathrm{h}}\|_\beta^2 \right) \mathrm{d}\bar{A},$$

where g,$\bar{\mathrm{g}}$ and h,$\bar{\mathrm{h}}$ are the metric tensor and the shape operator of the initial and the deformed middle surfaces Σ and $\bar{\Sigma}$, and $\|\ \|_\alpha$ and $\|\ \|_\beta$ are certain matrix norms that encode the thickness and material properties of the shell. The first term, the membrane energy, measures stretching and shearing of the surface, and the second term, the flexural energy, measures bending of the surface.

Discrete shells. In the discrete setting, we consider a shape space of surface meshes in $\mathbb{R}^3$. Let x be such a mesh, then we consider the linear shape space X generated by varying the positions of the vertices of x while leaving the mesh connectivity unchanged. We consider the discrete shells energy [Grinspun et al. 2003; Garg et al. 2007]. Analogous to the continuous case, the energy that governs this model of thin shells is a weighted sum of a flexural energy and a membrane energy,

$$E(x) = \alpha\, E^F(x) + \beta\, (E^L(x) + E^A(x)). \tag{1}$$

The weights α and β reflect the thickness of the shell and properties of the material to be simulated, *e. g.*, in cloth simulation the membrane energy usually gets a high weight due to the stretch resistance of cloth. The discrete flexural energy is given as a summation over the edges of the mesh:

$$E^F = \frac{1}{2}\sum_i \frac{3\,\|\bar{e}_i\|^2}{\bar{A}_{e_i}} \left(2\sin\frac{\theta_{e_i} - \bar{\theta}_{e_i}}{2} \right)^2, \tag{2}$$

where θ_{e_i} is the dihedral angle at the edge e_i, A_{e_i} is the combined area of the two triangles incident to e_i and $\|e_i\|$ is the length of the edge. The quantities $\|\bar{e}_i\|$, $\bar{A}_{e_i}$, and $\bar{\theta}_{e_i}$ are measured on the reference mesh $\bar{x}$. The membrane energy consists of two terms: one measuring stretching of the edges,

$$E^L = \frac{1}{2}\sum_i \frac{1}{\|\bar{e}_i\|}(\|e_i\| - \|\bar{e}_i\|)^2, \tag{3}$$

and one measuring the change of the triangle areas A_i,

$$E^A = \frac{1}{2}\sum_i \frac{1}{\bar{A}_i}(A_i - \bar{A}_i)^2. \tag{4}$$

Here, the last sum runs over the triangles of the mesh.

Surface modeling. In addition to a deformation energy E, we consider an energy E^C, that provides the user with control over the deformation. Parts of the surface are marked as handles and each of the handles can be translated and rotated in space to a desired position. Let v_i be the selected vertices and v_i' the prescribed positions of the vertices. Then, E^C is the quadratic energy

$$E^C(x) = \sum_i m_i (v_i - v_i')^2, \tag{5}$$

where m_i is the mass of v_i, *i. e.*, a third of the combined area of all triangles adjacent to v_i. For given constraints, the resulting surface deformation is the solution of the optimization problem

$$\arg\min_{x \in X} \mathcal{E}(x), \tag{6}$$

where $\mathcal{E}$ is given by

$$\mathcal{E}(x) = E(x) + \mu\, E^C(x). \tag{7}$$

For small μ the constraints are soft and allow some flexibility, and for larger values of μ the constraints tend towards equality constraints. Further energies, such as one that counteracts changes of the enclosed volume, can be added to E^C.

Connection to physical simulation. This framework for surface modeling is linked to physical simulation of elastic shapes. The dynamics of a time-dependent mesh $x(t)$, which represents an elastic shape, is described by a system of second order ODE's of the form

$$M\ddot{x}(t) = F(t, x(t), \dot{x}(t)) \tag{8}$$

where F represents the acting forces and M is the mass matrix of $x(t)$, see [Baraff and Witkin 1998; Bridson et al. 2003; Grinspun et al. 2003]. The forces F are a superposition of internal deformation forces $F^{int}(x(t))$ of the elastic shape, external forces $F^{ext}(t, x(t), \dot{x}(t))$, and damping forces $F^{damp}(\dot{x}(t))$. In our setting, $x(t)$ is the middle surface of an elastic thin shell and the deformation energy E is the potential energy of the internal forces of the shell: $F^{int}(x(t)) = -\nabla E_{x(t)}$. In addition, we consider external forces that equal the negative gradient of the energy μE^C, $F^{ext}(t, x(t), \dot{x}(t)) = -\mu \nabla E^C_{x(t)}$. The physical energy of the system is the sum of the kinetic energy $T(\dot{x}(t))$ and the potential energy $V(x(t))$, where V in our case equals the energy $\mathcal{E}$ defined in eq. (7). Due to damping, the system dissipates energy:

$$\frac{d}{dt}(T + V) \leq 0. \tag{9}$$

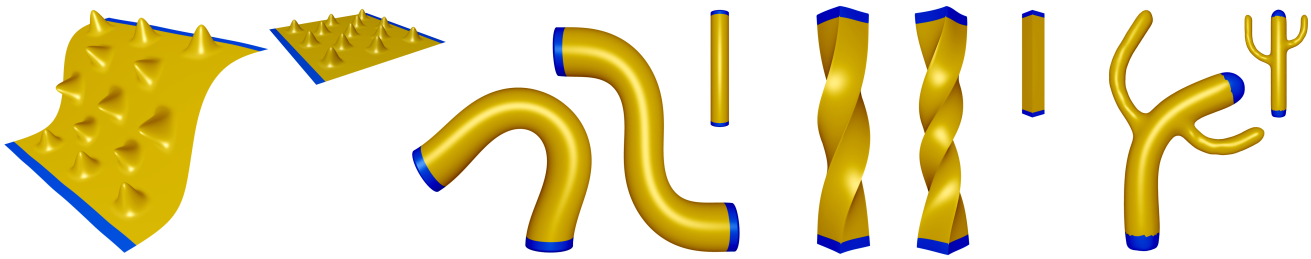


Fig. 4. Approximation quality of the reduced space with 67 dimensions in all images is demonstrated on the test suite of models and poses introduced in [Botsch and Sorkine 2008]. Two even larger deformations have been added.

This inequality is strict for all t such that $\|\dot{x}(t)\| \neq 0$ holds. For $t \to \infty$ the kinetic energy vanishes and $x(t)$ takes the position of a (local) minimum of $\mathcal{E}$. This means that the deformations our scheme computes are quasi-static limits, for $t \to \infty$, of dynamic simulations of elastic thin shells, where the user-defined constraints correspond to external forces in the simulation.

4. DIMENSION REDUCTION

For general meshes, solving the optimization problem, eq. (6), is too costly to achieve interactive rates. Even for coarse meshes with 5-10k vertices, it is challenging to get rates of 1fps. To reduce the complexity of the problem, we restrict the optimization to an affine subspace of X of the form $V_{\bar{x}} = \bar{x}+V$ where V is a subspace of X and $\bar{x}$ is the initial mesh. Then, the reduced optimization problem is

$$\arg\min_{x \in V_{\bar{x}}} \mathcal{E}(x). \tag{10}$$

An adequate space $V_{\bar{x}}$ should, on the one hand, be as small as possible, and, on the other hand, contain reasonable approximations of the desired deformations.

In our scheme, the generation of the reduced space is part of a preprocess, before the actual modeling session. To obtain a subspace that is suitable for various types of user interaction, we exclude the energy E^C from this process. As a consequence, the constraints, *e. g.*, the positions of handles, can be modified without forcing a recomputation or adjustment of the subspace. We generate V as the linear span of the union of two sets V_1 and V_2 of vectors. The set V_1 contains low-frequency eigenmodes of the Hessian of the deformation energy at $\bar{x}$. The motivation to use these directions is that we search for points in a shape space that minimize energy, and, since the gradient of the energy vanishes at the initial mesh $\bar{x}$, the low-frequency eigenmodes of the Hessian point into the directions in a shape space that (locally) cause the least increase of energy. From a physical point of view, these eigenmodes are (linearized) eigenvibrations of the surface around the rest state $\bar{x}$. The modes ϕ_i and eigenvalues λ_i are the solutions of the generalized eigenvalue problem

$$\nabla^2 E_{\bar{x}} \phi_i = \lambda_i \, M \phi_i, \tag{11}$$

where $\nabla^2 E_{\bar{x}}$ is the matrix containing the second partial derivatives of E at the initial mesh $\bar{x}$ and M is the mass matrix of $\bar{x}$, see [Hildebrandt et al. 2010]. The matrix $\nabla^2 E_{\bar{x}}$ is symmetric and at least positive semi-definite ($\bar{x}$ is a minimum of E) and M is symmetric and positive definite. Hence, the structure of this problem is similar to the generalized eigenvalue problem arising in manifold harmonics [Lévy and Zhang 2009; Zhang et al. 2010]. A fast solver that computes the modes of a lower part of the spectrum even for large meshes is presented in [Vallet and Lévy 2008]. For our purposes, we need only a small fraction of the lower part of the spectrum. For all figures, we used a set V_1 consisting of the modes ϕ_i corresponding to the lowest 15-20 eigenvalues.

Reduced shape spaces constructed only from linear modes can approximate small deformations well, but, as demonstrated in Fig. 5, the approximation of larger deformations in such spaces is often unsatisfactory. This is reflected in a large difference of energy between the minimum of the reduced problem and the minimum of the full problem. To extend the reduced space, we collect vectors that point into energy descent directions at points in $\bar{x}+\text{Span}(V_1)$ in the set V_2. Assume we are at some point x in $\bar{x}+\text{Span}(V_1)$. Then, an effective descent direction in the full space X would be the Newton direction $-\nabla^2 E_x^{-1} \nabla E_x$ at x, which is the direction in which a Newton solver would do a line search. The Taylor expansion around $\bar{x}$ of the Newton direction at x is

$$-\nabla^2 E_x^{-1} \nabla E_x = -u + \frac{1}{2} \nabla^2 E_{\bar{x}}^{-1} \nabla^3 E_{\bar{x}}(u,u) + \mathcal{O}(\|u\|^3) \tag{12}$$

where $u = x - \bar{x}$, $\nabla^3 E_{\bar{x}}(\cdot,\cdot,\cdot)$ is the third-rank tensor containing the third partial derivatives of E at $\bar{x}$, and $\nabla^3 E_{\bar{x}}(\phi_i,\phi_j)$ stands for the vector we get when we plug in ϕ_i and ϕ_j into $\nabla^3 E_{\bar{x}}(\cdot,\cdot,\cdot)$ (and transpose the resulting linear form). A derivation of (12) is provided in the appendix. We define vectors ψ_{ij} as the solutions of the equation

$$\nabla^2 E_{\bar{x}} \psi_{ij} = \nabla^3 E_{\bar{x}}(\phi_i,\phi_j). \tag{13}$$

Due to the symmetry of $\nabla^3 E_{\bar{x}}(\cdot,\cdot,\cdot)$, the vectors ψ_{ij} satisfy $\psi_{ij} = \psi_{ji}$. Furthermore, the first six linear modes span the (linearized) rigid motions, and the translations have vanishing derivatives. Hence, from the first k linear modes, we can at most construct $((k-3)^2 - (k-3))/2$ linear independent vectors ψ_{ij} by solving eq. (13) for all pairs ϕ_i,ϕ_j. We collect all the ψ_{ij} obtained from pairs of linear modes from V_1 in the set V_2. Then, at every point $x \in \bar{x}+\text{Span}(V_1)$ the vector $\nabla^2 E_{\bar{x}}^{-1} \nabla^3 E_{\bar{x}}(u,u)$, which appears in eq. (12), is in $\text{Span}(V_2)$. It follows that the affine space spanned by V_1 and V_2 contains an approximation up to the third order in $\|x-\bar{x}\|$ of the Newton direction (12) at every point $x \in \bar{x}+\text{Span}(V_1)$. In our experiments, we often did not use all ψ_{ij}, but found that using half of the number of possible ψ_{ij} is a good tradeoff between approximation quality and size of the reduced space. For example, the 67-dimensional basis that we used for many figures includes 52 linearly independent ψ_{ij}, computed in two *for*-loops over i and j, thus favoring ϕ_is with lower eigenvalues. For completeness, we would like to mention that eqs. (12) and (13) are only defined up to the kernel of $\nabla^2 E_{\bar{x}}$, which for most deformation energies are the linearized rigid motions. However, since the kernel of $\nabla^2 E_{\bar{x}}$ is contained in $\text{span}(V_1)$, the constructed re-

duced space is independent of the choice of vectors ψ_{ij} that solve eq. (13).

Furthermore, we would like to remark that the construction of the vectors ψ_{ij} is analogous to the simplified modal derivatives introduced by Barbič und James [2005], though their formulation does not use a potential energy and the simplified modal derivatives are defined and computed for the St. Venant-Kirchhoff model of three-dimensional elastic bodies. For this reason we call the ψ_{ij} the modal derivatives. Examples of vibration modes and modal derivatives of a simple shape, the bar, and of a complex shape, the dragon, are shown in Figs. 1 and 6. A physical interpretation of the modal derivatives is that ψ_{ij} describes how the mode ϕ_i changes (in first order) when $\bar{x}$ is deformed in direction ϕ_j. For our purposes, a benefit of the modal derivatives is that they help to compensate artifacts introduced by the linear modes.

To calculate the derivatives of the energies E^F, E^L, and E^A, we use the automatic-differentiation library *ADOL-C*, cf. [Griewank et al. 1996]. We do not need to compute the full tensor of third derivatives, but only the restriction of this tensor to span(V_1). Each of the three energies is a sum over contributions of the edges or the triangles of the mesh, and, to save main memory, we directly reduce the third derivatives of the individual summands. To compute the modal derivatives, we need to solve the linear equation (13) several times. Since the matrix $\nabla^2 E_{\bar{x}}$ is the same in all these linear systems, it is efficient to compute a sparse factorization of the matrix once and to use it to solve all the systems.

5. EFFICIENT SOLVER

The reduced optimization problem, eq. 10, is low dimensional, but evaluations of the energy and its derivatives are expensive and the reduced Hessian is a dense matrix. An efficient solver for this problem is the *BFGS* method, a quasi-Newton method cf. [Nocedal and Wright 2006]. This scheme, like steepest descent, requires only evaluations of the gradient at each iterate, but, unlike steepest descent, produces superlinear convergence.

The BFGS method uses an approximation of the inverse of the Hessian to generate a descent direction and therefore does not need

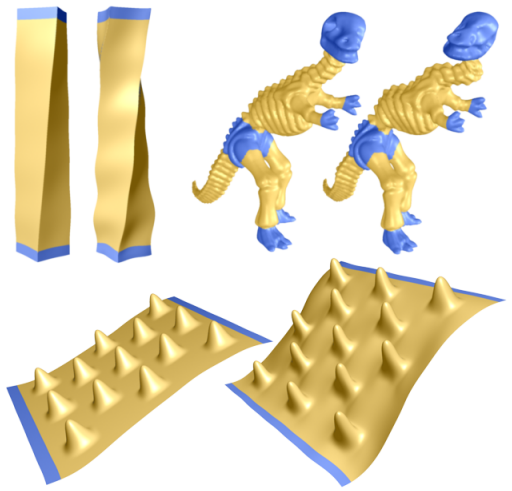


Fig. 5. Deformation results produced in reduced spaces spanned by the first 130 *linear modes*. For each model a small and a large deformation is shown. The larger deformation is produced with the same constraints as used for Figs. 3 and 4.

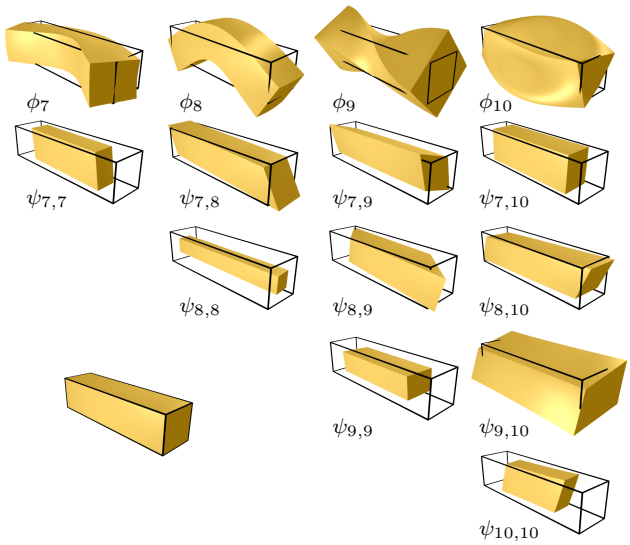


Fig. 6. Vibration modes and modal derivatives of the bar are shown: vibration modes in the top row and corresponding modal derivatives below. We leave out the first six modes because these span the linearized rigid motions.

to evaluate the Hessian or its inverse directly. Instead of computing the approximate inverse Hessian from scratch at each iteration, the change of the gradients at the recent step is used to update the matrix. Explicitly, the inverse Hessian update is given by

$$B_{k+1} = (I - \rho_k s_k y_k^T) B_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T,$$

where B_k is the approximate inverse Hessian at iteration k, $y_k = \nabla \mathcal{E}_{k+1} - \nabla \mathcal{E}_k$ the change of gradients, $s_k = x_{k+1} - x_k$ the change of position, and $\rho_k = 1/y_k^T s_k$. The classic BFGS method uses the identity matrix as the initial matrix B_1; this means it starts as a gradient descent and becomes more Newton-like during run time. To achieve a warm start of our solver, we compute the inverse of the reduced Hessian at the initial mesh once during the preprocess and use this matrix as the initial approximate inverse Hessian B_1 in the interactive phase. In our experiments, the BFGS solver (with warm start) requires a similar number of iterations to reach a local minimum as a Newton solver, which computes the full Hessian in each iteration, see Fig. 7.

Energy and gradient approximation. Since the solver maintains an approximate inverse Hessian, it does not need to solve a linear system to compute the descent direction. Then, the most expensive operations in each iteration are the evaluations of the energy and its gradient. To make these calls independent of the size of the input mesh $\bar{x}$, we need an approximation of the energy and the gradient

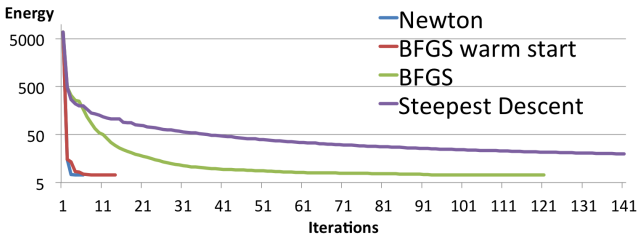


Fig. 7. Comparison of the performance of different optimization schemes: Newton's method, BFGS with and without warm start, and steepest descent.

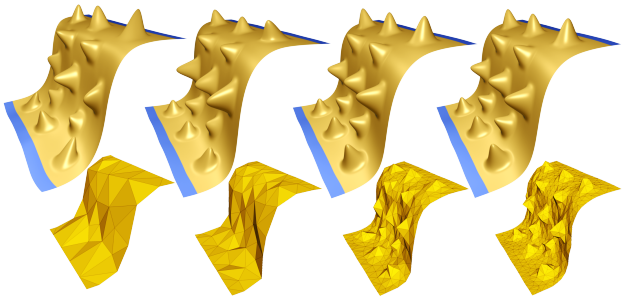


Fig. 8. Results for different ghost sizes. Number of vertices (faces) from left to right: 37 (50), 67 (100), 283 (500), and 544 (1000). The ghosts are shown in the bottom row.

for all x in the reduced shape space $V_{\bar{x}}$. For this, we build a low resolution version $\bar{x}^s$ of the initial mesh $\bar{x}$ with 500-5000 vertices. In addition, we simplify the subspace basis $\{b_i\}$. Each b_i is a vector field on $\bar{x}$, we compute corresponding vector fields b_i^s on the simplified mesh $\bar{x}^s$ and define the reduced shape space of the simplified mesh as $V_{\bar{x}^s}^s = \bar{x}^s + \text{Span}\{b_i^s\}$. Every $x \in V_{\bar{x}}$ has a unique representation in the basis $\{b_i\}$, $x = \bar{x} + \sum_i \alpha_i b_i$, and the linear map given by $\bar{x} + \sum_i \alpha_i b_i \rightarrow \bar{x}^s + \sum_i \alpha_i b_i^s$ is an isomorphism of the shape spaces $V_{\bar{x}}$ and $V_{\bar{x}^s}^s$. To approximate the energy of the surface $\bar{x} + \sum_i \alpha_i b_i \in V_{\bar{x}}$, we compute the energy of the coarse mesh $\bar{x}^s + \sum_i \alpha_i b_i^s$ and we proceed analogously to compute the gradient.

Explicitly, we use an edge-collapse scheme to generate the coarse mesh. Edge-collapse schemes implicitly generate a map from the vertices of the fine to the vertices of the coarse mesh: for every vertex of the fine mesh there is exactly one vertex on the coarse mesh to which it has been collapsed. Let v^s be a vertex of the coarse mesh and let $\{v_1, v_2, ..., v_n\}$ be the set of vertices on the fine mesh that are collapsed to v^s. We construct the simplified basis vectors b_i^s by setting $b_i^s(v^s) = \frac{1}{n}\sum_k b_i(v_k)$. If the initial mesh is strongly irregular, it is reasonable to include the masses of the vertices into this averaging process.

We would like to emphasize that though we compute the energy and gradient on a coarse mesh, the fine mesh varies in a space spanned by nice and smooth modes and modal derivatives that are computed from the fine mesh. Actually, the simplified mesh is never shown or handed to the user, therefore, we call it the *ghost*.

In real-time physical simulations, a different approach to speed up the evaluation of the forces is used, see [Barbič and James 2005]. They use cubic polynomials on the reduced space to approximate each coordinate of the forces. Transfered to our setting, this would mean to approximate the energy by a fourth-order polynomial in the reduced space. Since this polynomial in general is dense, it has $\mathcal{O}(r^4)$ coefficients, where r denotes the dimension of the reduced space. This would impose strict bounds on the maximum dimension that would still allow real-time performance of the method. In contrast, our technique to approximate the energy using a simplified mesh is independent of the dimension of the reduced space, and, therefore, allows for larger reduced spaces. For the 67- and 130- dimensional spaces we use in our experiments, the evaluation of the approximate energy based on a simplified mesh is orders of magnitude faster than the evaluation of a dense quartic polynomial on the reduced space. Furthermore, our scheme has a parameter, the number of vertices of the coarse mesh, that allows us to improve the approximation quality, if necessary.

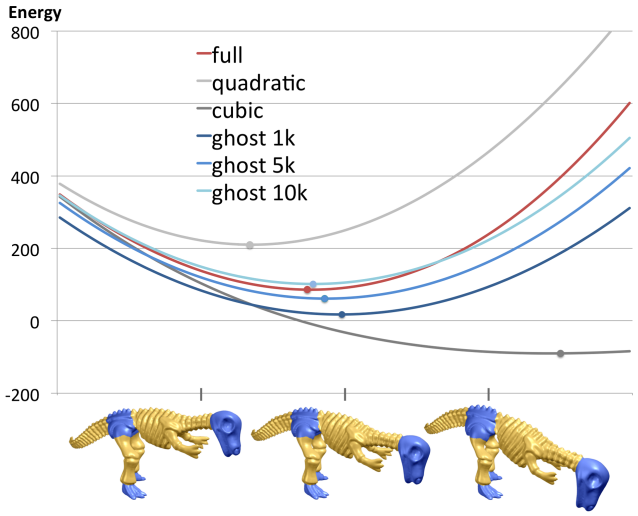
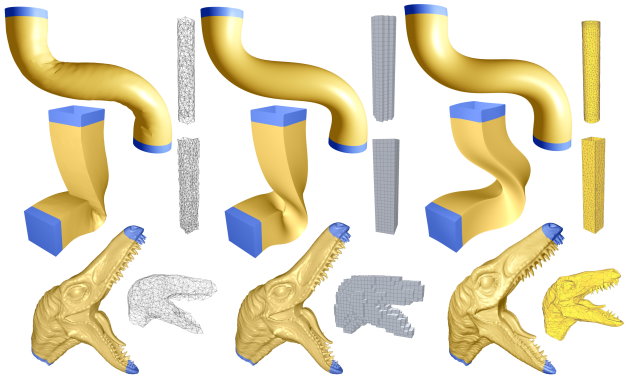


Fig. 9. Different approximations of the energy $\mathcal{E}$ in a one-dimensional affine subspace of the shape space are shown as graphs. The minima of the energies are indicated by dots. Graphs of the full energy, a second-order and a third-order Taylor series of $\mathcal{E}$, and approximations using ghosts with 1k, 5k, and 10k vertices are shown.

6. RESULTS AND DISCUSSION

Eigenvibrations of an elastic shape with small amplitude often look like natural deformations of the shape, as illustrated in Fig. 1, and shape spaces constructed from linear modes are well-suited to approximate small deformations. But for larger deformations, approximations in such spaces often develop distortions. This is illustrated in Fig. 5, which shows results obtained in spaces created only by linear modes for small and larger deformations. Our concept to extend the shape space by adding the modal derivatives ψ_{ij} largely improves the quality of the results. The large deformations shown in Fig. 5 can be compared to the results shown in Figs. 3 and 4 that are produced with the same poses but in spaces constructed from linear modes and modal derivatives. Examples of vibration modes and modal derivative are shown in Figs. 1 and 6. Results that our method produces in a reduced space with 67 dimensions (15 linear modes and 52 modal derivatives) on a set of typical test deformations are shown in Fig. 4. Considering the small size of the reduced space, even the large deformations are astonishingly well approximated. The results shown in Figure 4 can be compared with (unreduced) results of various schemes, including PriMo, shown in a comparison table in [Botsch and Sorkine 2008]. Our scheme is not limited to the discrete shells energy, but works with other shape deformation energies as well. To use it with other energies, it suffices to exchange the objective functional used for the optimization; if desired, the computation of the modes and modal derivatives can be done with other energies as well. Figure 3 shows results produced with the as-rigid-as-possible energy as objective functional.

In our experiments, the modeling framework runs robustly on various models, for small and large deformations, and with different parameter settings, like the dimension of the reduced space and the resolution of the coarse mesh. Our model reduction has an enormous effect in increasing the stability and reducing the stiffness of the optimization problem. Reasons for this effect are that the reduced shape spaces are low-dimensional and spanned by smooth vector fields that point into directions in which the energy increases



(Emb. Deforms.) (Rigid Cells) (Our method)

Fig. 10. Comparison of results of the Embedded Deformations and the Rigid Cells scheme with our method.

slowly. To demonstrate the stabilizing effect of our model reduction, we choose the discrete shell energy for most of our experiments, instead of the numerically more stable PriMo energy or as-rigid-as-possible energy. All figures are produced with both material parameters of the discrete shells energy, α and β in eq. (1), equal to 1 and we scaled each surface such that the longest edge of the bounding box has length 10. Fig. 11 shows the ghosts used to produce Figs. 2, 3, and 4. All ghosts are irregular and coarse meshes. We show experiments with various sizes of the reduced shape spaces and the ghosts in Figs. 8 and 12.

Energy approximation. Figure 9 shows a comparison of different approximations of the energy $\mathcal{E}$, where the discrete shells energy is used as a deformation energy. The full energy, a second-order (quadratic) and a third-order (cubic) Taylor series of $\mathcal{E}$ around $\bar{x}$, and approximations using ghosts with 1k, 5k, and 10k vertices are shown as graphs over a one-dimensional affine subspace of the shape space. To illustrate which subspace was used, we attach images that show shapes in the subspace to the x-axis of the image. The results of our method depend on the location of the minima rather than on the values of the energy, therefore, we added dots to the graphs that indicate the location of the minima. The figure illustrates the experimental observation that our technique to approximate the energy using a ghost mesh produces a smaller approximation error than a Taylor expansion up to second or third order. Furthermore, it demonstrates that the approximation error reduces with increasing size of the ghost mesh.

Running times. Table 1 shows running times of the configurations we used to produce the figures and additionally times of configurations for modeling the dragon model with varying parameters. This demonstrates that our framework produces interactive rates even for large meshes with 100k+ vertices. The time needed to solve the optimization problem mainly depends on the size of the reduced space and on the resolution of the coarse mesh. In our experiments, the time needed for one Newton iteration was between 4 and 33ms. During the interactive-modeling phase the constraints, which implement the user input, vary continuously. Therefore, we do not completely solve each optimization problem, but we update the constraints after either a fixed number of iterations is exceeded or an optimality criterion is satisfied. We use the optimality criterion discussed in [Gill et al. 1982], which, for a given ϵ, checks conditions on the change in energy $\mathcal{E}$, the convergence of the sequence $\{x_k\}$, and the magnitude of the gradient:

$$\begin{aligned} \mathcal{E}_{k-1} - \mathcal{E}_k &< \epsilon(1 + |\mathcal{E}_k|) \\ \|x_{k-1} - x_k\|_\infty &< \sqrt{\epsilon}(1 + \|x_k\|_\infty) \\ \|\nabla \mathcal{E}_k\|_\infty &< \sqrt[3]{\epsilon}(1 + |\mathcal{E}(x_k)|). \end{aligned} \tag{14}$$

In our experiments, we choose a maximum number of 5-10 iterations between updates of the constraints, which yields frame rates of 10+fps. After 5-10 iterations the optimality criterion is usually satisfied with ϵ between 10^{-3} and 10^{-4}. Still, we set $\epsilon = 10^{-6}$ to allow for further iterations if the constraints are not modified. This criterion is usually satisfied after about 15 Newton iterations. The reason that the running time for the as-rigid-as-possible energy (last row of Table 1) is much longer than the others is that our current implementation of the energy and gradient evaluation of this energy is very inefficient. Fig. 7 demonstrates the performance of different optimization schemes. It illustrates that our solver, BFGS with warm start, needs a similar number of iterations to converge as a Newton solver and shows that a BFGS without warm start still converges much faster than steepest descent. Steepest descent required 5806 iterations to converge and still did not reach the same energy level as the Newton or the BFGS solver. The drawback of our approach is that we need to generate the reduced shape space in a preprocess before the actual modeling session can start. In our prototype implementation, which leaves much room for optimization, the preprocess for the 40k bumpy plane took $5\frac{1}{2}$ minutes and for the dragon model with 130k vertices it took almost 20 minutes. Most of the time is spent on calculating the modal derivatives. But, for every model we only need to compute the reduced basis once and it can be stored with the mesh. Then, after loading, the modeling session can start almost immediately. Also, choosing new han-

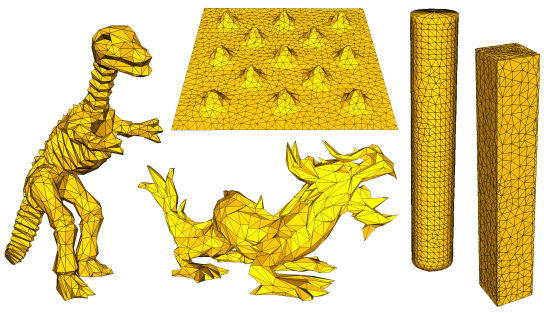


Fig. 11. The ghosts that were used to produce Figs. 2, 3, and 4 are shown.

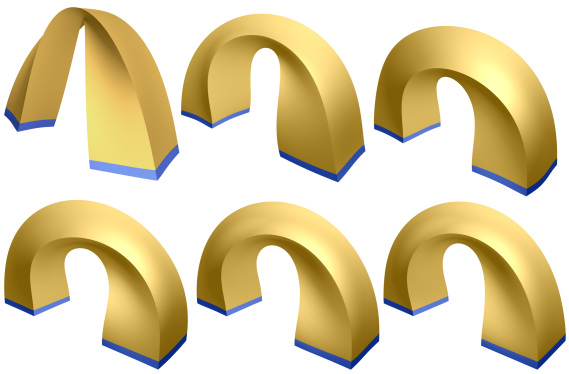


Fig. 12. A comparison of the results produced with reduced spaces with varying size is shown. Number of linear modes and modal derivatives from left to right: (8,6), (10,20), (15,52), (20,110), (30,270), full space.

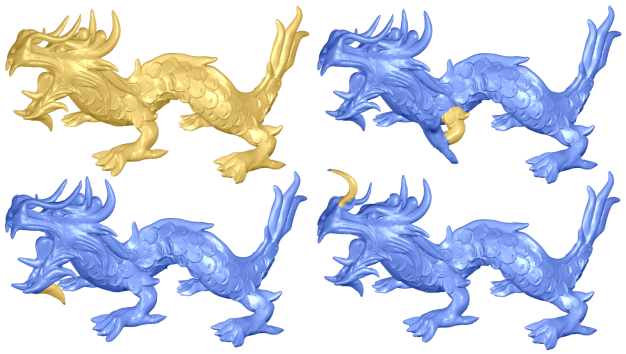


Fig. 13. Local deformations of dragon in a 130-dimensional shape space. The reference model is shown on the left of the top row.

dles or changing the resolution of the coarse mesh does not require recomputation of the basis.

Comparison to previous work. We compare the results of our method with two state-of-the-art deformation schemes: *Embedded Deformations* [Sumner et al. 2007] and *Rigid Cells* [Botsch et al. 2007]. The implementations of the methods were kindly provided by their respective authors. Fig. 10 shows deformations of the cylinder, the bar, and the head of the raptor model. The graph of Embedded Deformations, the cell complex of Rigid Cells, and the ghost of our method are shown on the right of every deformed model. The left column shows results of Embedded Deformations (with a graph of 200 vertices for the cylinder and raptor and 400 for the bar), the middle column of Rigid Cells (with 650 cells for the cylinder, 576 cells for the bar, and 1318 cells for the raptor), the right column of our method (with a 67-dim. shape space for the cylinder and bar, a 130-dim. space for the raptor, and a ghost of 1000 vertices for all). Compared to Embedded Deformations, the results our method produces are visually more appealing since Embedded Deformations produces some noise artifacts. The results of Rigid Cells are comparable to those of our method, however our method is considerably faster (see Table 1). There are three reasons for this: first, Rigid Cells requires an expensive interpolation scheme (using radial basis functions) to avoid noise artifacts, which we do not need; second, our method decouples the approximation quality of the energy (the size of the ghost) from the size of the reduced space, which allows us to use smaller reduced spaces while keeping a reasonable approximation quality of the energy; and, third, Rigid Cells is using volume meshes, which are typically larger than surface meshes.

Model	#Vert.	Dim.	Ghost	Solve	Df.	Total	Prep.	Fig.
Bumpy plane	40k	67	1k	5	3	53(10)	325	4
Cylinder	5k	67	1k	5	1	51(10)	38	4
Bar	6k	67	1k	5	1	51(10)	48	4
Cactus	5k	67	5k	32	1	161(5)	44	4
Dragon	130k	67	0.5k	4	8	48(10)	1067	
Dragon	130k	67	1k	5	8	58(10)	1067	
Dragon	130k	67	2.5k	14	8	78(5)	1066	
Dragon	130k	67	5k	33	8	173(5)	1064	
Dragon	130k	130	0.5k	7	13	69(8)	1185	
Dragon	130k	130	1k	10	13	93(8)	1185	2
Dino	56k	130	1k	10	6	86(8)	511	
Dino (ARAP)	56k	130	1k	72	6	366(5)	511	3

Table 1. Performance measured on a custom Macbook with a 2.66GHz CPU. From left to right: number of vertices, dimension of reduced space, number of vertices of the ghost, time in milliseconds for one BFGS iteration, time for mapping reduced solution into full shape space, time for full optimization (/w maximum number of iterations), time in seconds for the preprocess, and figure that shows the configuration.

Local deformations. A general problem of model reduction schemes for shape modeling is that detail editing is either impossible or requires special treatment. Fig. 13 and the two rightmost images of Fig. 3 demonstrate that a certain degree of locality is possible with our scheme, *e. g.*, the head of the dino can turn around or the arm can move without affecting other parts of the body. But Fig. 13 also shows (left image of the bottom row) that local deformations can introduce artifacts, in the shown example the mouth opens and lower jaw increases in size when the horn below the mouth is edited. In order to seamlessly switch between modeling of details, like moving a finger of the dino, and global modeling that preserves these detail edits, an integration of our scheme with a local editing method, *e. g.*, PriMo or as-rigid-as-possible, is needed. A benefit of our method for such an integration is that the same energy can be used for both local and global editing.

Future work. The framework for deformation-based modeling we present in this paper provides an efficient way to represent approximations of a large variety of shapes. We think that this technique can be useful for various problems in computer graphics that need to operate in a shape space of surfaces. We plan to extend our approach to shape matching and shape interpolation.

Our next steps to improve the presented method aim at speeding up the preprocess. We are working on an efficient representation of the tensor that contains the third derivatives at $\bar{x}$ for a certain class of deformation energies, where we exploit the property that the energy and the gradient vanishes at $\bar{x}$. In addition, we want to experiment with computing the modes and modal derivatives on a coarse mesh and then using an adequate interpolation scheme to generate smooth basis vector fields on the fine mesh.

Acknowledgements. We would like to thank Mario Botsch and Robert Sumner for sharing their software implementations and the anonymous reviewers for their comments and suggestions. This work was supported by the DFG Research Center MATHEON "Mathematics for Key Technologies" in Berlin.

REFERENCES

ADAMS, B., OVSJANIKOV, M., WAND, M., SEIDEL, H.-P., AND GUIBAS, L. J. 2008. Meshless modeling of deformable shapes and their motion. In *Proc. of Symposium on Computer Animation*. 77–86.

AN, S. S., KIM, T., AND JAMES, D. L. 2008. Optimizing cubature for efficient integration of subspace deformations. *Transactions on Graphics 27,* 5, 1–10.

AU, O. K.-C., TAI, C.-L., LIU, L., AND FU, H. 2006. Dual Laplacian editing for meshes. *IEEE TVCG 12*, 386–395.

BARAFF, D. AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH*. 43–54.

BARBIČ, J. AND JAMES, D. L. 2005. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Transactions on Graphics 24,* 3, 982–990.

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

BEN-CHEN, M., WEBER, O., AND GOTSMAN, C. 2009. Variational harmonic maps for space deformation. *Transactions on Graphics 28,* 3.

BOTSCH, M., PAULY, M., GROSS, M., AND KOBBELT, L. 2006. PriMo: Coupled prisms for intuitive surface modeling. In *Proceedings of Eurographics/Siggraph Symposium on Geometry Processing*. 11–20.

BOTSCH, M., PAULY, M., WICKE, M., AND GROSS, M. 2007. Adaptive space deformations based on rigid cells. In *Computer Graphics Forum*. Vol. 26(3). 339–347. Proceedings of Eurographics.

BOTSCH, M. AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14,* 1, 213–230.

BRIDSON, R., MARINO, S., AND FEDKIW, R. 2003. Simulation of clothing with folds and wrinkles. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 28–36.

CHAO, I., PINKALL, U., SANAN, P., AND SCHRÖDER, P. 2010. A simple geometric model for elastic deformations. *ACM Trans. Graph. 29*, 38:1–38:6.

CHOI, M. G. AND KO, H.-S. 2005. Modal warping: Real-time simulation of large rotational deformation and manipulation. *IEEE Transactions on Visualization and Computer Graphics 11,* 1, 91–101.

CIARLET, P. G. 2000. *Mathematical Elasticity - Volume III: Theory of Shells*. Studies in Mathematics and Its Applications, vol. 29. North Holland.

GARG, A., GRINSPUN, E., WARDETZKY, M., AND ZORIN, D. 2007. Cubic Shells. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 91–98.

GILL, P. E., MURRAY, W., AND WRIGHT, M. H. 1982. *Practical Optimization*. Academic Press.

GRIEWANK, A., JUEDES, D., AND UTKE, J. 1996. Algorithm 755: ADOL-C: a package for the automatic differentiation of algorithms written in C/C++. *ACM Trans. Math. Softw. 22,* 2, 131–167.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. In *Symposium on Computer Animation*. 62–67.

HAUSER, K. K., SHEN, C., AND O'BRIEN, J. F. 2003. Interactive deformation using modal analysis with constraints. In *Graphics Interface*. 247–256.

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2010. Eigenmodes of surface energies for shape analysis. In *Proceedings of Geometric Modeling and Processing*. 296–314.

HUANG, J., SHI, X., LIU, X., ZHOU, K., WEI, L.-Y., TENG, S.-H., BAO, H., GUO, B., AND SHUM, H.-Y. 2006. Subspace gradient domain mesh deformation. *ACM Transactions on Graphics 25,* 3.

HUANG, Q., WICKE, M., ADAMS, B., AND GUIBAS, L. 2009. Shape decomposition using modal analysis. *Comp. Graph. Forum 28,* 2, 407–416.

JOSHI, P., MEYER, M., DEROSE, T., GREEN, B., AND SANOCKI, T. 2007. Harmonic coordinates for character articulation. *ACM Transactions on Graphics 26,* 3.

JU, T., SCHAEFER, S., AND WARREN, J. 2005. Mean value coordinates for closed triangular meshes. In *ACM Transaction on Graphics*. 561–566.

KRAEVOY, V. AND SHEFFER, A. 2006. Mean-value geometry encoding. *International Journal of Shape Modeling 12,* 1, 29–46.

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*, 479–504.

LÉVY, B. AND ZHANG, H. 2009. Spectral mesh processing. In *ACM SIGGRAPH ASIA Courses*. 1–47.

LIPMAN, Y., LEVIN, D., AND COHEN-OR, D. 2008. Green coordinates. *ACM Trans. Graph. 27,* 3, 1–10.

LIPMAN, Y., SORKINE, O., COHEN-OR, D., LEVIN, D., RÖSSL, C., AND PETER SEIDEL, H. 2004. Differential coordinates for interactive mesh editing. In *Shape Modeling International*. 181–190.

NEALEN, A., SORKINE, O., ALEXA, M., AND COHEN-OR, D. 2005. A sketch-based interface for detail-preserving mesh editing. *ACM Trans. Graph. 24,* 3, 1142–1147.

NOCEDAL, J. AND WRIGHT, S. J. 2006. *Numerical Optimization (2nd edition)*. Springer.

PENTLAND, A. AND WILLIAMS, J. 1989. Good vibrations: modal dynamics for graphics and animation. In *Proc. of ACM SIGGRAPH*. 215–222.

SHI, X., ZHOU, K., TONG, Y., DESBRUN, M., BAO, H., AND GUO, B. 2007. Mesh puppetry: cascading optimization of mesh deformation with inverse kinematics. *ACM Trans. Graph. 26,* 3, 81.

SORKINE, O. AND ALEXA, M. 2007. As-rigid-as-possible surface modeling. In *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. 109–116.

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Symposium on Geometry Processing*. 175–184.

SUMNER, R. W., SCHMID, J., AND PAULY, M. 2007. Embedded deformation for shape manipulation. In *Transactions on Graphics*. Vol. 26(3).

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *Proc. of ACM SIGGRAPH*. 205–214.

VALLET, B. AND LÉVY, B. 2008. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum*.

ZHANG, H., VAN KAICK, O., AND DYER, R. 2010. Spectral mesh processing. *Computer Graphics Forum 29,* 6, 1865–1894.

Appendix

In this appendix, we verify the Taylor expansion of the Newton direction of E around $\bar{x}$ that was stated in eq. (12). Formally, the Taylor series of $-\nabla^2 E_x^{-1} \nabla E_x$ around $\bar{x}$ is

$$-\nabla^2 E_x^{-1} \nabla E_x = -\nabla^2 E_{\bar{x}}^{-1} \nabla E_{\bar{x}} - \nabla(\nabla^2 E_{\bar{x}}^{-1} \nabla E_{\bar{x}})(u) - \frac{1}{2}\nabla^2(\nabla^2 E_{\bar{x}}^{-1} \nabla E_{\bar{x}})(u,u) + \mathcal{O}(\|u\|^3)$$

where $u = x - \bar{x}$. Since $\bar{x}$ is a minimum of E, the first term of the right-hand side vanishes, and we have

$$\nabla(\nabla^2 E_{\bar{x}}^{-1} \nabla E_{\bar{x}}) = \nabla^2 E_{\bar{x}}^{-1} \nabla^2 E_{\bar{x}} = Id,$$

which shows that the second term reduces to $-u$. Furthermore, $\nabla^2 E_x^{-1} \nabla^2 E_x$ equals the identity matrix for all x, which implies

$$0 = \nabla(\nabla^2 E_{\bar{x}}^{-1} \nabla^2 E_{\bar{x}}) = \nabla(\nabla^2 E_{\bar{x}}^{-1}) \nabla^2 E_{\bar{x}} + \nabla^2 E_{\bar{x}}^{-1} \nabla^3 E_{\bar{x}}.$$

Using this, we get

$$\begin{aligned}\nabla^2(\nabla^2 E_{\bar{x}}^{-1} \nabla E_{\bar{x}}) &= \nabla(\nabla(\nabla^2 E_{\bar{x}}^{-1}) \nabla E_{\bar{x}} + \nabla^2 E_{\bar{x}}^{-1} \nabla^2 E_{\bar{x}}) \\ &= \nabla(\nabla^2 E_{\bar{x}}^{-1}) \nabla^2 E_{\bar{x}} = \nabla^2 E_{\bar{x}}^{-1} \nabla^3 E_{\bar{x}}.\end{aligned}$$

Hence, the third term of the formal Taylor series satisfies:

$$-\frac{1}{2}\nabla^2(\nabla^2 E_{\bar{x}}^{-1} \nabla E_{\bar{x}})(u,u) = -\frac{1}{2}\nabla^2 E_{\bar{x}}^{-1} \nabla^3 E_{\bar{x}}(u,u)$$

Altogether, we have verified eq. (12).