Real-Time Subspace Integration for St.Venant-Kirchhoff Deformable Models

Jernej Barbič Doug James *

Carnegie Mellon University

Abstract

In this paper, we present an approach for fast subspace integration of reduced-coordinate nonlinear deformable models that is suitable for interactive applications in computer graphics and haptics. Our approach exploits dimensional model reduction to build reduced-coordinate deformable models for objects with complex geometry. We exploit the fact that model reduction on large deformation models with *linear* materials (as commonly used in graphics) result in internal force models that are simply cubic polynomials in reduced coordinates. Coefficients of these polynomials can be precomputed, for efficient runtime evaluation. This allows simulation of nonlinear dynamics using fast implicit Newmark subspace integrators, with subspace integration costs independent of geometric complexity. We present two useful approaches for generating low-dimensional subspace bases: modal derivatives and an interactive sketching technique. Mass-scaled principal component analysis (mass-PCA) is suggested for dimensionality reduction. Finally, several examples are given from computer animation to illustrate high performance, including force-feedback haptic rendering of a complicated object undergoing large deformations.

CR Categories: I.6.8 [Simulation and Modeling]: Types of Simulation—Animation, I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling, I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: simulation, animation, deformation, precomputation, model reduction, finite element method, interactive, haptics

1 Introduction

Objects undergoing physically based *large* deformations play an important part of computer graphics and animation where shape changes must be visible, and their simulation is notorious for being computationally demanding. For example, the high update rates of force-feedback haptic rendering make it difficult to accurately simulate large deformations, especially with complex geometry and distributed contact interactions (see Figure 1). Many interactive and offline simulations, such as those used in the computer animation industry, would also benefit from having highly interactive large deformation models.

In this paper, we show that dimensional model reduction on deformable models with geometric nonlinearities but *linear materials*, as commonly used in graphics (the so-called St. Venant-Kirchhoff model, or StVK), can lead to extremely fast and precomputable approximations for real-time applications. We exploit the fact that dimensional model reduction in this case results in internal force models that are simply cubic polynomials in reduced coordinates. Coefficients of these reduced force polynomials can be precomputed for efficient runtime evaluation of exact internal forces and stiffness matrices. All the integration costs are independent of geometric complexity. Consequently, large deformation physics can be integrated at extremely fast rates using trusted subspace integrators, e.g., implicit Newmark, while graphical rendering is done at slower rates. For example, the large bridge example shown in Figure 1 can only be dynamically rendered at about 40 Hz, but its dynamics can be integrated at more than a kilohertz, thus enabling haptic simulations of complex large-deformation models. In general, the integration speed is proportional to the number of subspace dimensions employed, e.g., with 4 dimensions the bridge dynamics can be integrated at over 200 kHz.

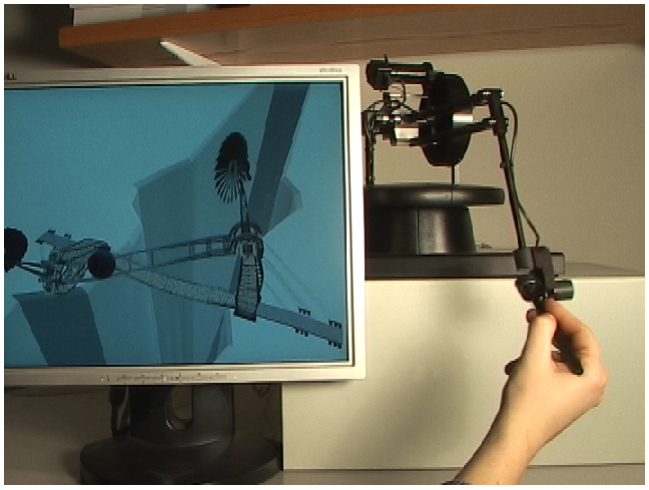

Figure 1: **Large-deformation dynamics at kilohertz rates:** *Force-feedback haptic rendering of distributed contact interactions between a user-controlled ball and a flexible bridge model (59630 triangles, $r = 15$). Subspace dynamics and contact handling are simulated at a hard real-time (1000 Hz) update rate. Reduced coordinates are exploited for real-time Bounded Deformation Tree collision processing [James and Pai 2004].*

Our proposed approach is most closely related to linear modal vibration models, first introduced to graphics by Pentland and Williams [July 1989]. These linear dynamics models are simple and fast, and have seen extensive use [Shinya and Fournier 1992; Stam 1997; Basdogan 2001; James and Pai 2002; Hauser et al. 2003; Choi and Ko 2005]. Unfortunately, geometric linearity leads to distortions for large deformations, which is a significant limitation for computer graphics. Our proposed approach preserves several nice properties of linear modal analysis, but overcomes the serious limitation of linear Cauchy strain by employing full quadratic Green strain in all computations. Namely, we can still capture the large-scale motion of a model with very few modes. We also preserve the property that progressively more modes can be used to increase simulation accuracy.

Given the deformation basis, our approach can automatically generate a fast reduced-coordinate model. A key challenge there-

*e-mail: barbic|djames@cs.cmu.edu

fore is to construct a good reduced deformation basis for describing general large deformation problems. To this end, we present two approaches to good quality basis motion generation: modal derivatives and a sketch interface. Modal derivatives provide a fully-automatic approach where the standard linear modal analysis basis is augmented by the derivatives of the linear modal basis vectors. In the sketch-based interface, the user is presented a linear modal analysis model and interacts with it. The imposed forces are recorded, and then an offline FEM solver generates the deformation samples. Finally, we use a variant of the PCA data-reduction method to process the obtained samples, extracting the nonlinear modal shape basis (the empirical eigenvectors) of the characteristic deformation space. We demonstrate our method on a variety of examples, including force-feedback haptic rendering.

2 Related Work

Real-time deformable objects: Simulation of large-deformation models is a well-understood area in interactive computer graphics and offline solid mechanics. Physics-based large-deformation models have been used successfully in graphics for almost two decades [Terzopoulos et al. 1987; Baraff and Witkin 1992; Metaxas and Terzopoulos 1992], and enjoy widespread application in mature graphics areas, such as cloth simulation [Baraff and Witkin 1998; Bridson et al. 2002].

StVK models are often sufficient for the purposes of computer animation, and their use in many recent papers attests to that [Zhuang and Canny 2000; O'Brien and Hodgins 1999; Picinbono et al. 2001; Debunne et al. 2001; Capell et al. 2002a]. For example, interactive simulations using direct integration include geometric nonlinearities, however the runtime assembly of all the cubic force terms for every element limits the interactivity to only a few hundred elements [Zhuang and Canny 2000; Picinbono et al. 2001].

Multi-resolution approaches use hierarchical deformation bases to adaptively refine the analysis based on deformation activity of the model [Debunne et al. 2001; Capell et al. 2002a; Grinspun et al. 2002]. Domain embedding approaches are commonly used in graphics for interactive applications, since high resolution meshes can be deformed using coarse deformable models [Pentland and Williams July 1989; Faloutsos et al. 1997; Müller and Gross 2004].

For linear material models, nonlinear kinematics can be simplified by exploiting local frames of reference. Multibody dynamics approaches exploit local frames of reference when time-stepping small deformations, and are widely used in graphics [Terzopoulos and Witkin 1988; Metaxas and Terzopoulos 1992; Shabana 1990]. Closely related to this are so-called "stiffness warping" methods (c.f. corotational formulations) [Müller et al. 2002; Müller and Gross 2004; Irving et al. 2004], wherein an element undergoing large deformations, with linear materials, simply reuses the undeformed linear element by rotating it to the current frame of reference. Linear materials have also been exploited for fast large-deformation kinematics of Cosserat rods [Pai 2002].

Modal warping [Choi and Ko 2005] is an approximation of StVK models that is based on extrapolating per-element rotations during modal dynamics to produce a fast parametric nonlinear shape model. This approach is easy to implement and is useful for eliminating gross distortions associated with linear modal analysis. However, by virtue of linear modal analysis, the dynamics of warped modes are driven by *independent* simple harmonic oscillators. Consequently, an initial condition exciting only one of the modes will generate single-mode motion (regardless of amplitude), and hence the well-known nonlinear coupling of modes cannot be captured. On the other hand, our nonlinear modes are accurately coupled via an analytic reduction of the StVK model. Also, there is no guarantee that "warped modes" are sufficient for large deformation dynamics. In contrast, our approach uses a reduced displacement basis produced from actual nonlinear shape statistics. Another difference is illustrated by deformations in which no element rotations occur, such as a beam's axial extension mode. With modal warping, forces and volume grow linearly as the beam extends, whereas in our model, forces are cubic polynomials and structure becomes stiffer with extension. Modes also couple to counteract volume growth. Finally, one benefit of our linear shape model is that it can accelerate collision detection [James and Pai 2004].

StVK models are perhaps the simplest kind of physical large-deformation model, and one well-known deficiency is that forces are inaccurate under larger compressions (see [Irving et al. 2004] for a discussion). In the worst case, elements may actually invert without proper restoring forces, and suitable steps must be taken to address element inversion [Irving et al. 2004]. Although our approach is not suitable for simulating the general and complex deformations found in Irving et al. [2004], it is designed to be substantially faster for interactive applications. Finally, we note that concerns about element inversion are constrained to our precomputation phase, and are not a major concern for runtime subspace integration, since the shape subspace greatly restricts the likelihood of element inversion.

To this date, most precomputation-based approaches for real-time simulation have considered geometrically and materially linear models. For fast elastostatics, condensation approaches have been used to obtain boundary responses [Bro-Nielsen and Cotin 1996], as well as precomputation of boundary Green's function responses [Cotin et al. 1999; James and Pai 1999].

James and Fatahalian [2003] precompute nonlinear deformation responses to a finite set of user impulses, and apply dimensional reduction using PCA. Although their approach handles self-collisions, it greatly restricts the range of possible runtime interactions to a small discrete set of pre-selected impulses. On the other hand, our approach allows general runtime forcing within the reduced-dimensional subspace.

Subspace integration is closely related to discretizations using global displacement bases that are commonly used in graphics to avoid solving large systems (e.g., during semi-implicit integration), and reducing numerical stiffness (for explicit timestepping), e.g., global polynomial shape functions [Baraff and Witkin 1992], deformable super-quadrics [Metaxas and Terzopoulos 1992], free-form deformation basis functions [Faloutsos et al. 1997], and multiresolution discretizations also project dynamical equations using multiresolution scaling functions [Grinspun et al. 2002]. However, one drawback with these approaches for interactive applications is that they all suffer from evaluating unreduced internal forces (and any Jacobians) at each time step, with cost typically proportional to geometric complexity.

Model reduction in solid mechanics: *Dimensional model reduction* is a technique to simplify simulation of dynamical systems described by differential equations. Complex systems can be simulated by reducing the dimensionality of the problem, yielding systems of differential equations involving fewer equations and fewer unknown variables. These equations can be solved much more quickly than the original problem, with some accuracy cost to the solution. This method also appears in literature under the names of *Principal Orthogonal Directions Method*, and *Subspace Integration Method*, and it has a long history in the engineering and applied mathematics literature [Lumley 1967].

In nonlinear solid mechanics, early methods extended the principle of mode superposition for linear vibration analysis by using local tangent mode superposition [Nickell 1976], and later the derivatives of tangent eigenmode vectors were also included [Idelsohn and Cardona 1985b]. Explicit computation of the coefficients of reduced force polynomials for a time-varying basis of motion is suggested in [Almroth et al. 1978]. These techniques are not suit-

able for interactive applications because they periodically involve timesteps with a large amount of computation, such as when the local basis is updated, and the number of derivative modes required for accuracy can grow too quickly to be efficient. Recently, a statistical approach to basis generation for finite element models was presented by Krysl et al. [2001], wherein a full-degree of freedom system is first simulated, and then standard PCA is applied to the resulting deformations to obtain a typical deformation basis. This is a non-interactive technique with external forces known and fixed in advance, and the simulated nonlinear deformations were relatively small compared to deformations in our method. Also, reduced internal forces and reduced stiffness matrices were assembled by first constructing *unreduced* quantities (followed by subspace projection), which is prohibitively expensive for interactive simulation of complex models.

3 Background: Subspace Integration

3.1 Basic Deformation Concepts

Continuum mechanics provides the physical background to modeling deformable objects, and we refer the reader to [Fung 1977] for an introduction. Background on nonlinear solid mechanics can be found in [Belytschko 2001; Bonet and Wood 1997; Holzapfel 2000]. StVK material is defined by a linear stress-strain relationship of the form

$$\mathbf{S} = \lambda(\mathrm{tr}\,\mathbf{E})\mathbf{I}_3 + 2\mu\mathbf{E}, \tag{1}$$

where $\mathbf{S}$ is the second Piola stress tensor, $\mathbf{E}$ is the Green-Lagrange strain tensor, $\mathbf{I}_3$ is the 3×3 identity matrix, and λ and μ are (possibly spatially varying) Lamé constants. It is an example of a hyperelastic isotropic material: elastic strain energy is a unique function of body deformation only (and not of deformation history), and at any location, material is equally stretchable in all directions.

Without loss of generality, we use the Finite Element Method (FEM) to discretize partial differential equations of solid continuum mechanics. The deformable body is represented as a volumetric mesh consisting of 3D polyhedra called *elements*. A particular body deformation is specified by the displacements of mesh vertices. For a volumetric mesh consisting of n vertices, the *displacement vector* $u \in \mathbb{R}^{3n}$ contains the x,y,z world-coordinate displacements of model vertices. A small set of vertices are constrained to have zero displacements [1].

In computer graphics, it is often useful to simulate models which are essentially polygon soups. We follow a common approach in graphics, wherein a 3D volumetric simulation mesh drives the deformations of a triangle mesh. The volumetric mesh is obtained by voxelizing the triangle mesh into tiny elastic cubes (8-node first order brick elements) [James et al. 2004; Müller et al. 2004]. Inhomogeneous material parameters can be assigned to the cubes. External forces acting on the triangle mesh vertices are transfered to simulation mesh vertices via simple trilinear interpolation. Likewise, resulting displacements are transfered back to the triangle mesh. While we found this discretization convenient during precomputation, we remind the reader that this paper's contribution is general, and can be applied to arbitrary elements.

3.2 Equations of Motion

After the FEM discretization, the motion of a deformable solid can be described by the Euler-Lagrange equation [Shabana 1990], which is a second order system of ordinary differential equations

$$M\ddot{u} + D(u,\dot{u}) + R(u) = f. \tag{2}$$

[1] An extension to unconstrained meshes is possible, see Appendix D.

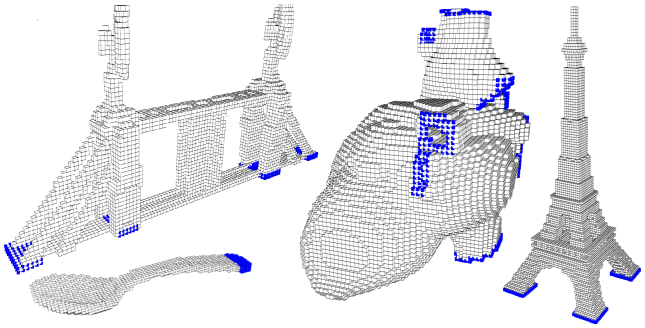

Figure 2: **Simulation Meshes:** *Blue vertices are constrained.*

Here, $u \in \mathbb{R}^{3n}$ is the displacement vector (the unknown), $M \in \mathbb{R}^{3n,3n}$ is the *mass matrix*, $D(u,\dot{u})$ are damping forces, and $R(u)$ are internal deformation forces. The mass matrix depends only on the object's mesh and mass density distribution in the rest configuration. In general, it is a sparse non-diagonal matrix, however for algorithmic convenience, it is often simplified into a diagonal matrix by accumulating all the row entries onto the diagonal element (*mass lumping*). Our approach can handle both lumped and non-lumped versions of the mass matrix. Internal forces corresponding to the displacement u are given by the vector $R(u) \in \mathbb{R}^{3n}$. The mapping R is nonlinear due to the nonlinearity of the Green-Lagrange strain tensor, and (in general) due to any material nonlinearities. Note that the matrix M, and the mappings D and R are independent of time. Apart from u, the only time-dependent term in the equation is the vector of external forces f, used to model, e.g., user interactions or collision response. Let $K(u) \in \mathbb{R}^{3n,3n}$ denote the Jacobian matrix of the internal forces R, evaluated at u, i.e., the *tangent stiffness matrix*. Also, let $K = K(0^{3n})$ denote the stiffness matrix at the origin (here 0^{3n} denotes the $3n-$dimensional zero vector). We use a local Rayleigh damping model of the form

$$D(u,\dot{u}) = \Big(\alpha M + \beta K(u)\Big)\dot{u}, \tag{3}$$

This damping model is controlled by two positive real-valued parameters, α and β, which, roughly, have the effect of damping low and high time-frequency components of deformations, respectively. This damping model is a generalization of the more familiar linear Rayleigh damping model, which would be obtained if $K(u)$ were replaced by K. In practice, the presence of high frequency damping significantly improves the stability of the simulation.

3.3 Reduced Equations of Motion

In model reduction for solid mechanics, the displacement vector is expressed as $u = Uq$, where $U \in \mathbb{R}^{3n,r}$ is some *displacement basis matrix*, and $q \in \mathbb{R}^r$ is the vector of *reduced coordinates*. Here, U is a time-independent matrix specifying a basis of some r-dimensional ($r \ll 3n$) linear subspace of $\mathbb{R}^{3n}$. There is an infinite number of possible choices for this linear subspace and for its basis. Good subspaces are low-dimensional spaces which well-approximate the space of typical nonlinear deformations. The choice of subspace

	rendering		voxel	simulation	
	vertices	triangles	resolution	vertices	elements
spoon	3321	6638	100	3698	2005
bridge	41361	59630	128	11829	5854
tower	45882	105788	140	20713	11304
heart	12186	23616	80	28041	14444

Figure 3: **The characteristics of models used in our paper.**

depends on geometry, boundary conditions and material properties. Selection of a good subspace is a non-trivial problem and we will return to it in the next sections. For now, simply assume that a good subspace basis U is available. Also, for a given $r-$dimensional subspace of the full deformation space $\mathbb{R}^{3n}$, there are many choices for a specific basis for this subspace, and this choice can impact numerical stability. One choice would be to pick an orthogonal basis, however, it is more natural to make columns of U mass-orthonormal (see Appendix A), i.e., impose $U^T M U = I_r$, where I_r is the $r \times r$ identity matrix. By inserting $u = Uq$ into Equation 2, and premultiplying by U^T, one obtains the *reduced equations of motion*. These equations determine the dynamics of the reduced coordinates $q = q(t) \in \mathbb{R}^r$, and therefore also the dynamics of $u(t) = Uq(t)$:

$$\ddot{q} + \tilde{D}(q,\dot{q}) + \tilde{R}(q) = \tilde{f} \tag{4}$$

where $\tilde{D}$, $\tilde{R}$ and $\tilde{f}$ are r-dimensional reduced forces,

$$\begin{aligned} \tilde{D} &= U^T D(Uq, U\dot{q}), & (5)\\ \tilde{R}(q) &= U^T R(Uq), & (6)\\ \tilde{f} &= U^T f. & (7) \end{aligned}$$

Similarly, one can form the *reduced tangent stiffness matrix*,

$$\tilde{K}(q) = U^T K(Uq) U \in \mathbb{R}^{r,r}. \tag{8}$$

The existence theorem for systems of ordinary differential equations assures that the system in Equation 4 has a well-defined unique solution, given a specific instance of initial conditions and time-dependent external forces. Since $r \ll 3n$, the integration of (4) is much faster than the integration of the unreduced system (2), albeit with some accuracy loss.

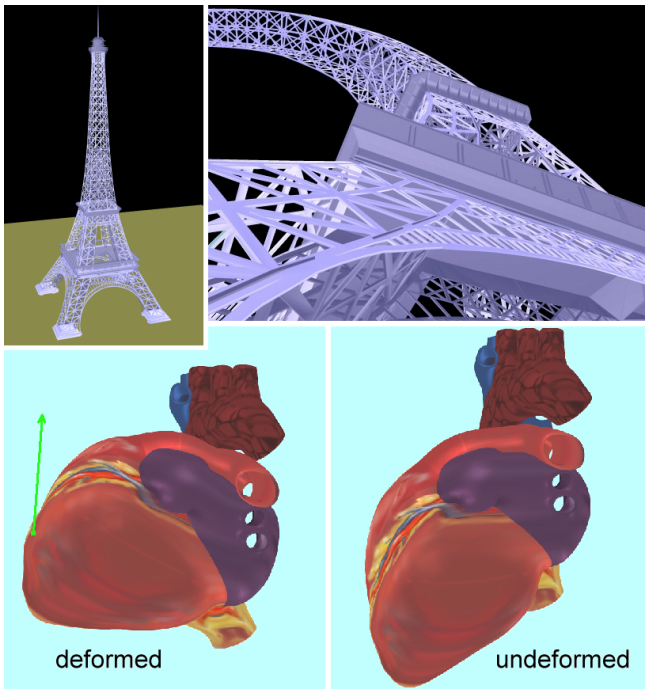

Figure 4: **Subspace integration of Eiffel tower and heart models**

4 Polynomial Reduced Forces

Following equations of continuum mechanics, it can be shown that for StVK material with nonlinear Green-Lagrange strain, the strain energy of a given deformation $u \in \mathbb{R}^{3n}$ is a fourth order multivariate polynomial function in the components of u. The terms of this polynomial are localized, in the sense that the displacements of two vertices can only appear together in a term if the two vertices share an element. Full internal force on a mesh vertex equals the gradient of the energy with respect to the x,y,z coordinates of the deformation of the vertex. Consequently, each component of the unreduced force is a third-order multivariate polynomial function in the displacements of the vertex and all its immediate mesh neighbors.

4.1 Reduced Internal Forces are Cubic Polynomials

Consequently, for deformations of the form $u = Uq$, each component of the reduced internal force $\tilde{R}(q) \in \mathbb{R}^r$ is a multivariate cubic polynomial in components of reduced coordinates q:

$$\begin{aligned} \tilde{R}(q) &= U^T R(Uq) = & (9)\\ &= P^i q_i + Q^{ij} q_i q_j + S^{ijk} q_i q_j q_k, & (10) \end{aligned}$$

where $P^i, Q^{ij}, S^{ijk} \in \mathbb{R}^r$ are some constant vector coefficients. Furthermore, the reduced tangent stiffness matrix $\tilde{K}(q) \in \mathbb{R}^{r,r}$ is just the Jacobian of $\tilde{R}(q)$, and therefore, each component of $\tilde{K}(q)$ is a multivariate quadratic polynomial in q. Specifically, column ℓ of $\tilde{K}$ equals

$$\frac{\partial \tilde{R}(q)}{\partial q_\ell} = P^\ell + (Q^{\ell i} + Q^{i\ell}) q_i + (S^{\ell ij} + S^{i\ell j} + S^{ij\ell}) q_i q_j. \tag{11}$$

In general all polynomial coefficients P^i, Q^{ij}, S^{ijk} are non-zero.

4.2 Precomputing Polynomial Coefficients

The coefficients of all the cubic and quadratic polynomials from the previous subsection can be efficiently precomputed. Note that there is one cubic polynomial per reduced force dimension (r cubic polynomials total), and one quadratic polynomial per entry of the reduced tangent stiffness matrix ($r(r+1)/2$ quadratic polynomials total due to symmetry of the stiffness matrix). Precomputation proceeds by first computing all the coefficients of the reduced force polynomials. This can be done in $O(r^4)$ time per element, and the algorithm is given in Appendix B. After these coefficients are known, the coefficients of the reduced tangent stiffness matrix polynomials can be obtained easily (Equation 11).

Model	r	num elements	precomputation time	size of coefficients
spoon	12	2005	60 sec	98 Kb
bridge	15	5854	186 sec	223 Kb
tower	30	11304	79.2 min	3.0 Mb
heart	30	14444	97.4 min	3.0 Mb

Figure 5: **Precomputing polynomial coefficients:** *Reported numbers are totals for both reduced force and reduced stiffness matrix.*

5 Deformation Basis Generation

Deformation basis generation is a hard open problem in solid mechanics, and there exist no algorithms for automatic proven-quality global deformation basis generation under general forcing. Existing approaches use PCA on example motion to generate a low-dimensional basis for a specific context, i.e., "empirical eigenvectors" [Krysl et al. 2001]. However two problems with this approach are that (a) for interactive applications, it is unclear what example motion would best describe the essential deformation behavior

of future uses, and (b) it is not automatic, since we can not simply press a button and build a general purpose model. In this section, we present two techniques, one providing an automatic, and one providing an interactive way to building nonlinear deformation bases. Both basis generation techniques apply to general nonlinear materials and aren't limited to StVK.

5.1 Modal Derivatives

Linear modal analysis [Shabana 1990] (LMA) provides the best deformation basis for small deformations away from the rest pose. Intuitively, modal basis vectors are directions into which the model can be pushed with the smallest possible increase in elastic strain energy. A generalization is possible: for any deformation $u_0 \in \mathbb{R}^{3n}$, *tangent linear vibration modes* give the best basis for small deformations away from the deformation pose u_0. The first $k \geq 1$ tangent linear vibration modes at u_0 (denoted by $\Psi^i(u_0), i = 1,\ldots,k$) are the mass-normalized eigenvectors corresponding to the k smallest eigenvalues $0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_k$ of the symmetric generalized eigenproblem $(K(u_0))x = \lambda Mx$. Tangent linear modes coincide with LMA modes at the origin (define $\Psi_i := \Psi_i(0^{3n})$). Standard LMA simulation uses linear modes with *linear* forces and suffers from very visible errors for large deformations. A small improvement can be achieved by using $U = \{\Psi^i \,|\, i = 1,\ldots,k\}$ as a deformation basis in a reduced subspace integrator (i.e. with *nonlinear* internal forces). In our experiments, we clearly detected a modest improvement.

Alternatively, one can investigate how tangent linear vibration modes change with u_0. We combine this approach with mass-PCA to generate the deformation basis U. We evaluate the directional derivative of $\Psi^i(u_0)$, at the origin, for the LMA directions $u_0(p) = \Psi^\ell p_\ell$ (note the summation convention), as shown in [Idelsohn and Cardona 1985b]. Here, parameter $p = p_\ell e^\ell \in \mathbb{R}^k$ is the vector of *modal participation factors*. The unnormalized *modal derivatives* can be defined as

$$\Phi^{ij} = \frac{\partial}{\partial p_j}\Big(\Psi^i(\Psi^\ell p_\ell)\Big)_{|p=0^k}. \qquad (12)$$

It can be shown that a quadratic term now extends the LMA linear deformation space into a parabola:

$$u(p) = \Psi^i p_i + \frac{1}{2}\Phi^{ij} p_i p_j + O(p^3). \qquad (13)$$

If the effects of inertia terms are neglected, derivatives are symmetric ($\Phi^{ij} = \Phi^{ji}$), and can be precomputed by solving linear systems

$$K\Phi^{ij} = -(H : \Psi^i)\Psi^j, \qquad \text{where} \qquad (14)$$

$$H_{ij\ell} = \frac{\partial}{\partial u_\ell}\Big(K_{ij}(u)\Big)_{|u=0^{3n}}, \qquad i,j,\ell = 1,\ldots,3n \qquad (15)$$

denotes the *Hessian stiffness tensor*. This third rank tensor is the derivative of the stiffness matrix at the origin (see Appendix B). Contraction $H : a$ (for a vector $a = a^\ell e_\ell$) denotes the matrix where element (i,j) equals $H_{ij\ell}a^\ell$, for $i,j = 1,\ldots,3n$. Normalized modal derivatives $\overline{\Phi}^{ij}$ are obtained by mass-normalizing Φ^{ij}.

Equation 13 suggests that the linear space spanned by all vectors Ψ^i and Φ^{ij} is a natural candidate for a motion subspace. It could be processed with mass-Gramm-Schmidt to obtain a mass-orthonormal basis [Idelsohn and Cardona 1985b]. However, its dimension $k + k(k+1)/2$ quickly becomes prohibitive. Instead, we scale the derivatives according to the eigenvalues of the corresponding linear modes. Namely, we obtain the low-dimensional deformation basis by applying mass-PCA on

$$\left\{\frac{\lambda_1}{\lambda_j}\Psi^j \mid j = 1,\ldots,k\right\} \cup \left\{\frac{\lambda_1^2}{\lambda_i\lambda_j}\overline{\Phi}^{ij} \mid i \leq j;\ i,j = 1,\ldots,k\right\} \qquad (16)$$

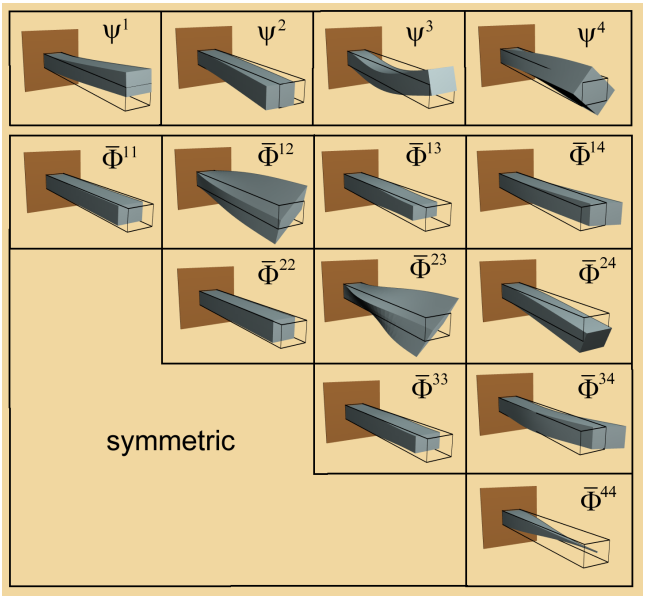

Figure 6: **Dominant linear modes and modal derivatives** *We exploit the statistical redundancy of these modes using mass-PCA of suitably scaled modes. All vectors are shown mass-normalized.*

Scaling is necessary to put greater weight on the more important low-frequency modes and their derivatives, which could otherwise be masked by high-frequency modes and derivatives. Note that K is a sparse symmetric matrix, and that different modal derivatives can be computed in parallel. Preconditioning K by the incomplete Cholesky factorization speeds up the computation.

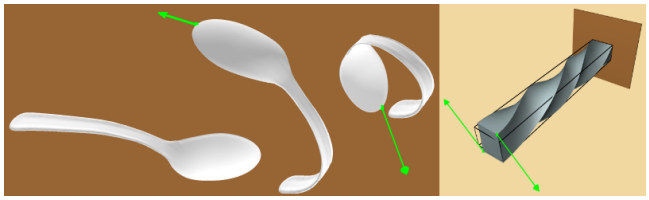

Figure 7: **Extreme shapes captured by modal derivatives:** *Although modal derivative are computed about the rest pose, their deformation subspace contains substantial nonlinear content to describe large deformations. (Left) Spoon ($k = 6, r = 15$) is constrained at far end. (Right) Beam ($r = 5$, twist angle=270°) is simulated in a subspace spanned by "twist" linear modes and their derivatives $\Psi^4, \Psi^9, \Phi^{44}, \Phi^{49}, \Phi^{99}$.*

Model	k	Compute linear modes	Build right-hand sides of Eq. 14	Solve Eq. 14
spoon	6	24 sec	6.5 sec	33 sec
tower	20	65 sec	226 sec	26 min
heart	20	111 sec	291 sec	28 min

Figure 8: **Computation of Modal Derivatives:** *All performance data is given for a single 3.0 Ghz Pentium workstation with 2Gb of memory. Mass-normalization and mass-PCA times are small.*

5.2 Interactive Sketching

Fast interactive linear models are available, and they can be used as a bootstrapping mechanism to obtain a basis of nonlinear deformations. The user first interacts with a linear vibration model [James and Pai 2002]. We use a static model to avoid the dynamic effects

which could confuse the user. Due to linearity, the model distorts badly for large deformations, but still provides a clue to the deformation involved. The forces imposed by the user are recorded to disk. A subset of these forces is automatically selected so that a certain separation distance is maintained among consecutive forces. These forces are then sent as input to a full unreduced offline static solver which for every imposed load f computes the static rest configuration u. Again, a subset of all deformations is automatically selected to maintain a certain separation mass-distance. Mass-PCA is then applied on the resulting shapes to extract the basis of motion U. When this basis is later used for an interactive nonlinear simulation, the model will be able to simulate nonlinear deformations similar to those sketched. Additional sketches can be used to refine the motion basis as desired.

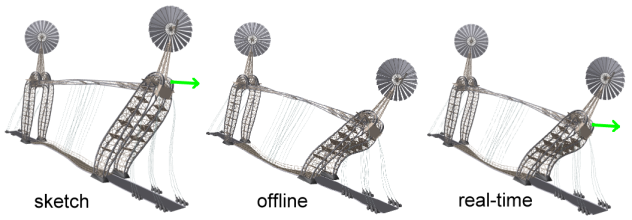

Figure 9: **Basis from Sketch:** *(Left) User interacts with a linear model. Resulting shape is distorted. (Center) Applied force is recorded and sent to an unreduced offline static solver to solve for the corresponding nonlinear shape. Several such shapes are then processed by mass-PCA to obtain a basis of motion. (Right) If same force is re-applied during the reduced runtime simulation, a shape which is visually almost indistinguishable from the center image emerges.*

Model	num selected force loads	num selected deformations	static solve
spoon	353	45	45 min
bridge	326	142	2.4 hours

Figure 10: **Precomputation Timings for the Basis from Sketch.**

6 Runtime Computation

6.1 Implicit Integration

To timestep the simulation at runtime, we numerically integrate the system from Equation 4. This is a nonlinear system of r coupled second order differential equations. Nonlinearity is due to the forcing and damping terms. We use the implicit Newmark integrator (see [Wriggers 2002]), which is second-order accurate and commonly used in structural dynamics. An alternative choice would be the central differences explicit Newmark integrator, which doesn't require the assembly of the reduced tangent stiffness matrix nor a linear system solve at every time step. However, we found it hard to control the explicit timestep as numerical stiffness can cause the explicit integrator to be unstable. A necessary condition for the explicit integrator to be stable is that the timestep be able to represent the oscillations of the highest eigenfrequency of the linearized reduced system around the origin. When r is increased, more high frequency content tends to enters the solution, and explicit timestep is progressively limited. Moreover, stability of the model at the origin doesn't guarantee global stability, since stiffness typically increases as the model moves away from the origin. Because guaranteed stability is very important for interactive applications, and because local Rayleigh damping model requires the assembly of the reduced tangent stiffness matrix anyway, we decided to use the implicit integrator.

In general, one implicit Newmark step involves several Newton-Raphson sub-steps, each requiring the solution of a dense $r \times r$ linear system. However, in line with previous research in graphics, we found it sufficient to perform a single Newton-Raphson iteration per timestep. This is a speed-accuracy tradeof, and if necessary, multiple Newton-Raphson iterations can be performed per timestep. The linear system to be solved is a dense $r \times r$ symmetric linear system, and we solve it using a direct symmetric matrix solver. Note that iterative solvers are not as attractive in this case due to relatively small r and dense matrices. The implicit Newmark integrator is given in Appendix C. At any timestep, with the system in state $q \in \mathbb{R}^r$, it is necessary to evaluate reduced internal forces $\tilde{R}(q)$ and the reduced tangent stiffness matrix $\tilde{K}(q)$. We note that for a general nonlinear material, $\tilde{R}(q)$ is a complicated function. For a general isotropic hyperelastic material, it is a large sum of rational functions involving logarithmic terms. In general, it has several poles, and doesn't possess an immediate compact and simple analytical expression. Hence, direct evaluation of such functions is non-trivial. One could proceed by evaluating full unreduced forces $R(Uq) \in \mathbb{R}^{3n}$ and forming explicit projection $\tilde{R}(q) = U^T R(Uq)$ (and similarly for the reduced tangent stiffness matrix), however such approach would currently not be real-time for large models.

6.2 Runtime Polynomial Evaluation

For the special case of the StVK material, there is a simple exact polynomial formula for reduced internal forces, as shown in the previous section. At runtime, given a state q, we directly evaluate the precomputed polynomials. Evaluation of each component of $\tilde{R}(q)$ involves $\Theta(r^3)$ operations, and evaluation of each component of the reduced tangent stiffness matrix involves $\Theta(r^2)$ operations, so both evaluations can be performed in $\Theta(r^4)$ time. Note that evaluation time is independent of the number of vertices and elements in the model. About half of the computation time can be saved with the tangent stiffness matrix by exploiting that it is symmetric. Even though polynomials are low-degree and involve all possible terms, evaluation order does matter. During pre-process, we organize all the precomputed coefficients of the quadratic terms of the reduced stiffness matrix $\tilde{K}(q)$ into a constant matrix $\overline{S}$. Each row of this matrix corresponds to one entry of $\tilde{K}(q)$: it contains all the quadratic coefficients of the entry. Then, to evaluate the quadratic terms of $\tilde{K}(q)$ at runtime, we first assemble $q_i q_j$ for all $i \leq j$ into a vector $\overline{q}$, and multiply $\overline{S}$ by $\overline{q}$. A similar scheme was used to quickly evaluate the cubic terms of $\tilde{R}(q)$. The number of lower-order terms is smaller and their evaluation is faster.

6.3 External Forces

Before each rendering step, we reconstruct the full $3n$-dimensional displacement vector u by performing matrix-vector multiply $u = Uq$. A collision detection routine can then use vector u to determine the external forces f for the next timestep. External forces also occur as a result of user interaction, e.g. a user pulling a certain vertex or set of vertices in certain directions. Subsequently, the external forces are projected into the basis U by equation $\tilde{f} = U^T f$. Implementation can make use of the fact the user interaction vector f is typically sparse.

6.4 GPU-accelerated Implementation

Matrix-vector multiply $u = Uq$ can be easily performed on CPU. We have also implemented it in graphics hardware. Matrix U is stored in texture memory (16-bit floating point format is sufficient). In pass 1, a fragment shader multiplies $u = Uq$ and renders the resulting deformation vector u to texture. In pass 2, a vertex shader fetches u from texture memory, and a standard rendering pipeline follows. Such an implementation leaves more room on CPU for other computations. Also, model geometry is now effectively static

Model	r	evaluate $[\mu s]$		solve linear system $[\mu s]$	integration total $[\mu s]$	N	time for $u = Uq$ $[\mu s]$	graphics frame rate	
		force	stiffness matrix					standard impl.	GPU-accelerated
spoon	12	8.2	9.5	12.5	30.2	25	565	275 Hz	470 Hz
bridge	15	22.0	25.0	18.4	65.4	10	14500	38 Hz	84 Hz
tower	30	550	770	75	1395	15	25500	17 Hz	40 Hz
heart	30	550	770	75	1395	15	6500	31 Hz	45 Hz

Figure 11: **Runtime Computation Performance.** *Integration times refer to one integration step. The number of integration steps per graphics frame is N.*

and can be efficiently cached in a display list, which avoids bus-bandwidth bottlenecks of rendering dynamic deformable geometry.

6.5 Runtime Modification of Material Parameters

If necessary, our method allows for interactively changing material parameters of the mesh at runtime. Exact polynomials for the new values of material parameters can be generated interactively, since Lamé coefficients λ, μ and mass density appear linearly in the formulas for internal forces and the mass matrix. Mesh needs to be divided into separate groups, with constant material parameters over each group. Two polynomials are precomputed for each group, one collecting only the λ-terms (and setting $\lambda = 1$), and one involving only the μ-terms (and setting $\mu = 1$, see Appendix B). To edit parameters, polynomials for each group are weighted by current group values of λ, μ, and all the group polynomials are summed together to produce the exact global polynomials. Changing mass density for different parts of the mesh can be done in a similar fashion. Note that the precomputed basis will become less optimal if material parameters deviate too far from those used for precomputation. It can however be shown that the modal derivative basis is invariant under uniform global scaling of Young's modulus and/or mass density. Also, it is possible to omit any subset of basis vectors from the basis before each individual runtime invocation: the terms corresponding to omitted dimensions simply need to be dropped from the polynomials. In particular, any first $r' \leq r$ basis vectors can be used for a particular runtime invocation.

7 Evaluation

We compared our methods to an unreduced implicit Newmark simulation with full internal force and stiffness matrix computation. Same simulation parameters were used in all cases. Using a reduced interactive model, we recorded a short user-exerted vertical external force impulse, applied at the end of the spoon. This impulse was used to generate all the simulations, and was strong enough to push the spoon deeply into the nonlinear region. If mass-PCA is applied on unreduced motion, and the resulting basis is used to re-simulate the motion, the resulting trajectory lies very close to the original motion. At around the first maximum, a short transient wave motion occurs in the full solution and such traveling localized deformations are difficult to capture by subspace dynamics. The modal derivatives and sketching bases produce almost correct amplitudes and 4.6%, 10.1% smaller nonlinear frequencies, respectively.

8 Discussion

Deformation modes in our paper have global support. Typically, the number of modes is too small to represent deformations involving high spatial frequencies, so such deformations can't be simulated. Of course, one solution is to add the corresponding localized basis functions into the basis. However, doing so for all localities on the model would quickly result in a basis whose size prohibits interactive applications. In the future, we plan to incorporate our simulation into an adaptive multi-resolution FEM framework.

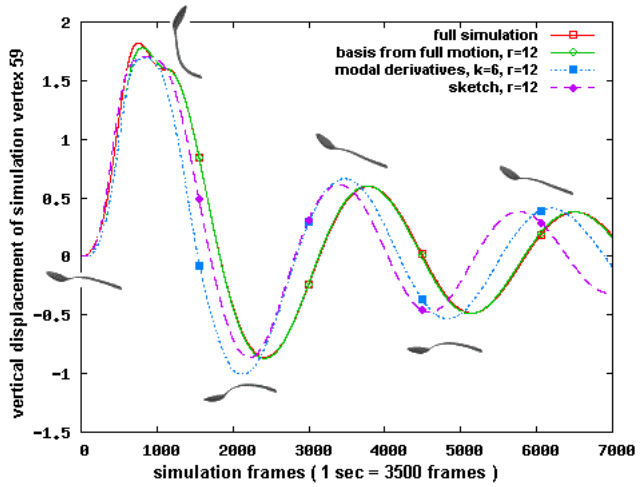

Figure 12: *Vertical displacement of a spoon simulation mesh vertex, located centrally at the end of the spoon. Length of spoon is about 2.5 units. Triangle mesh poses are shown for reference.*

Deformations in our paper are large and self-collisions can occur in extreme poses. Self-collisions were not a focus of our paper, but could be addressed in the future, for example by augmenting the Bounded Deformation Tree [James and Pai 2004] method to detect self-collisions efficiently. During self-contact, basis refinement may be required due to the changed boundary conditions.

For certain isolated extreme deformation poses, and for extremely low values of r (e.g. $r = 2$ for the bridge), the reduced internal force field can contain spurious stable equilibriums. This is a manifestation of the fact that the chosen value of r is simply too small to represent the problem. In our experience, this problem can always be solved by increasing r.

Strain-rate damping (see [Debunne et al. 2001]) could be used instead of local Rayleigh damping. The damping forces are again cubic polynomials in q and $\dot{q}$, and the coefficients could be precomputed. A matrix of quadratic polynomials in q would now appear in front of the $\ddot{q}$ term in the Euler-Lagrange equation. However, we found local Rayleigh damping model sufficient for our applications.

Acknowledgements: We would like to thank NSF (CAREER-0430528), The Link Foundation, Pixar, The Boeing Company, and NVIDIA for generous support. We also thank Guido Dhondt for help with his solid mechanics package CalculiX.

Appendix

A Mass-scaled PCA

PCA is usually performed with respect to the standard Euclidean metric, however a generalization is possible to any inner-product-originating distance metric between pairs of deformation vectors u and v. Standard Euclidean metric is suboptimal: for non-uniform

meshes it over-emphasizes deformations in parts of the mesh where vertices are dense. It also ignores the mass distribution of the object. Alternatively, *mass-scaled metric* ($M > 0$ is the mass matrix)

$$\|u-v\|_M := \sqrt{< M(u-v), u-v >} \tag{17}$$

weights the vertices according to the local amount of mass. Given a set of deformations $u^{(1)}, u^{(2)}, \ldots, u^{(N)}$, and dimensionality r, the objective of mass-scaled PCA is to find the $r-$dimensional hyperplane for which the sum of squared mass-projection errors in the mass metric is minimized. Only the hyperplanes passing through the origin are of interest, since we want the zero deformation to be representable by the model. Using Cholesky decomposition $M = LL^T$, it can be shown that substitution $z^{(i)} = L^T u^{(i)}$ translates the problem to a standard Euclidean PCA problem for the dataset $\mathscr{Z} = \{z^{(i)} \,|\, i = 1, \ldots, N\}$. Also, the resulting best Euclidean-orthonormal basis V for $\mathscr{Z}$ satisfies $V = L^T U$, where U is the optimal mass-scaled basis. To perform mass-scaled PCA, we explicitly form the $z^{(i)}$, and perform standard PCA. Mass-orthonormal basis U is then obtained by solving linear systems $L^T U = V$. Note that for models of constant mass density mass-scaled PCA reduces to volume-scaled PCA.

B Reduced Force Polynomials

Let $u_a^i \in \mathbb{R}^3$ denote the deformation of vertex a under deformation mode i, for $i = 1, \ldots, r$. Denote the contribution of element e to the global reduced internal force polynomial coefficients by $P_e^i, Q_e^{ij}, S_e^{ijk}$. These contributions can be obtained by inserting standard FEM formulas for StVK unreduced internal forces [Capell et al. 2002b] into Equation 9 (summation is over all vertices of e):

$$P_e^i = U_c^T \left(A_1^{ca} u_a^i + B_1^{ac} u_a^i + A_2^{ac} u_a^i \right) \tag{18}$$

$$Q_e^{ij} = U_c^T \left((\frac{1}{2} C_1^{cab} + C_2^{abc})(u_a^i \cdot u_b^j) + (u_b^i \otimes u_a^j)(C_1^{abc} + C_2^{cab} + C_2^{bac}) \right) \tag{19}$$

$$S_e^{ijk} = U_c^T \left((\frac{1}{2} D_1^{abcd} + D_2^{acbd})(u_a^i \cdot u_b^j) u_a^k \right) \tag{20}$$

$$A^{ab} = \int_e \nabla\phi_a \otimes \nabla\phi_b dV \quad \in \mathbb{R}^{3,3} \tag{21}$$

$$B^{ab} = \int_e \nabla\phi_a \cdot \nabla\phi_b dV \quad \in \mathbb{R} \tag{22}$$

$$C^{abc} = \int_e \nabla\phi_a (\nabla\phi_b \cdot \nabla\phi_c) dV \quad \in \mathbb{R}^3 \tag{23}$$

$$D^{abcd} = \int_e (\nabla\phi_a \cdot \nabla\phi_b)(\nabla\phi_c \cdot \nabla\phi_d) dV \quad \in \mathbb{R} \tag{24}$$

$$A_1^{ab} = \lambda A^{ab}, \quad A_2^{ab} = \mu A^{ab}, \quad B_1^{ab} = \mu B^{ab}, \quad C_1^{abc} = \lambda C^{abc},$$
$$C_2^{abc} = \mu C^{abc}, \quad D_1^{abcd} = \lambda D^{abcd}, \quad D_2^{abcd} = \mu D^{abcd}. \tag{25}$$

Here, ϕ_a denotes the shape function corresponding to vertex a, i.e. $\phi_a(a) = 1$ and $\phi_a(b) = 0$ for $a \neq b$. Lamé constants λ and μ relate to Young's modulus E and Poisson ratio ν as follows:

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \qquad \mu = \frac{E}{2(1+\nu)}. \tag{26}$$

To obtain the global coefficients P^i, Q^{ij}, S^{ijk}, sum the contributions of all the elements. Efficient parallel implementations are possible. Contribution of element e to blocks corresponding to vertices a, b, c of the full unreduced stiffness matrix and Hessian tensor at the origin, and the mass matrix (ρ is mass density) are:

$$M_e^{ab} = \left(\int_e \rho \phi_a \cdot \phi_b dV \right) I_3 \ \in \mathbb{R}^{3,3}, \qquad K_e^{ab} = A_1^{ab} + B_1^{ba} + A_2^{ba} \ \in \mathbb{R}^{3,3}, \tag{27}$$

$$H_e^{abc} = (C_1^{abc} + C_2^{bca} + C_2^{cba}) \otimes I_3 \quad + \quad I_3 \otimes (C_1^{cba} + C_2^{acb} + C_2^{bca}) \quad + \\ + \Sigma_{i=1}^3 \left(e_i \otimes (C_1^{bca} + C_2^{abc} + C_2^{cba}) \otimes e_i \right) \ \in \mathbb{R}^{3,3,3}. \tag{28}$$

Note that all the coefficients A, B, C, D and parameters λ, μ, ρ are in general element-specific.

C Implicit Newmark Subspace Integration

Algorithm *One step of implicit Newmark subspace integration*
Input: values of $q, \dot{q}, \ddot{q}$ at timestep i, reduced external force $\tilde{f}_{i+1}$ at timestep $i+1$; max number of Newton-Raphson iterations per step $j_{\max}$ (semi-implicit solver: $j_{\max} = 1$); tolerance TOL to avoid unnecessary Newton-Raphson steps; timestep size Δt.
Output: values of $q, \dot{q}, \ddot{q}$ at timestep $i+1$

$q_{i+1} \leftarrow q_i$;
for $j = 1$ **to** $j_{\max}$ // perform a Newton-Raphson iteration:
  Evaluate reduced internal forces $\tilde{R}(q_{i+1})$;
  Evaluate reduced stiffness matrix $\tilde{K}(q_{i+1})$;
  Form the local damping matrix
    $\tilde{C} = \alpha\tilde{M} + \beta\tilde{K}(q_{i+1})$; // in our work $\tilde{M} = I_r$
  Form the system matrix $A = \alpha_1\tilde{M} + \alpha_4\tilde{C} + \tilde{K}(q_{i+1})$;
  residual $\leftarrow \tilde{M}(\alpha_1(q_{i+1} - q_i) - \alpha_2\dot{q}_i - \alpha_3\ddot{q}_i) +$
    $+\tilde{C}(\alpha_4(q_{i+1} - q_i) + \alpha_5\dot{q}_i + \alpha_6\ddot{q}_i) + \tilde{R}(q_{i+1}) - \tilde{f}_{i+1}$;
  if ($\|\text{residual}\|_2 < TOL$)
    break out of for loop;
  Solve the $r \times r$ dense symmetric linear system:
    $A\left(\Delta q_{i+1}\right) = -\text{residual}$
  $q_{i+1} \leftarrow q_{i+1} + \Delta q_{i+1}$;
$\dot{q}_{i+1} \leftarrow \alpha_4(q_{i+1} - q_i) + \alpha_5\dot{q}_i + \alpha_6\ddot{q}_i$; // update velocities
$\ddot{q}_{i+1} \leftarrow \alpha_1(q_{i+1} - q_i) - \alpha_2\dot{q}_i - \alpha_3\ddot{q}_i$; // update accelerations
Return $q_{i+1}, \dot{q}_{i+1}, \ddot{q}_{i+1}$;

Integrator uses parameters $0 \leq \tilde{\beta} \leq 0.5$, $0 \leq \tilde{\gamma} \leq 1$, and constants

$$\alpha_1 = \frac{1}{\tilde{\beta}(\Delta t)^2}, \ \alpha_2 = \frac{1}{\tilde{\beta}\Delta t}, \ \alpha_3 = \frac{1-2\tilde{\beta}}{2\tilde{\beta}}, \ \alpha_4 = \frac{\tilde{\gamma}}{\tilde{\beta}\Delta t}, \ \alpha_5 = 1 - \frac{\tilde{\gamma}}{\tilde{\beta}}, \ \alpha_6 = \left(1 - \frac{\tilde{\gamma}}{2\tilde{\beta}}\right)\Delta t.$$

We chose $\tilde{\beta} = 0.25$, $\tilde{\gamma} = 0.5$, which is a common setting for many applications. Explicit central differences integrator is defined by $\tilde{\beta} = 0, \tilde{\gamma} = 0.5$. Constants α, β are Rayleigh damping constants.

D Modal Derivatives for Unconstrained Deformable Models

Section 5.1 demonstrated how to determine modal derivatives for anchored meshes. For models with no constrained vertices, the first six eigenvalues $\lambda_1, \ldots, \lambda_6$ are zero with the eigenvectors spanning the space of infinitesimal rigid body motions. Derivatives are however still defined via Equation 12. To form a motion basis, we combine linear modes $\Psi^i, i \geq 7$ with derivatives $\overline{\Phi}^{ij}, i, j \geq 7$ (appropriately scaled, followed by mass-PCA). Rigid body motion can then be coupled with deformations [Terzopoulos and Witkin 1988].

To compute the derivatives, first note that the approach from Equation 14 is not directly applicable: stiffness matrix K is now singular (nullspace dimension is six), and there is no guarantee that Equation 14 has a solution. One approach to determine $\Phi^{ij}, i, j \geq 7$ is to use the full formulation from [Idelsohn and Cardona 1985a]:

$$\left(K - \lambda_i M\right)\Phi^{ij} = \left(M\Psi^i(\Psi^i)^T - I_{3n}\right)\left((H : \Psi^i)\Psi^j\right). \tag{29}$$

Note that Equation 14 follows by neglecting mass terms and that modal derivatives are now no longer symmetric. The matrix $K - \lambda_i M$ is singular and its nullspace consists of multiples of Ψ_i. However, it can be shown that Equation 29 always has a solution, and that to find a solution, one can solve the regularized version of the system, obtained by replacing $K - \lambda_i M$ with $\overline{K} := K - \lambda_i M + \Psi^i(\Psi^i)^T$. Any multiple of Ψ_i can be added to any solution of Equation 29. A particular solution Φ^{ij} can be chosen by imposing $(\Psi^i)^T M \Phi^{ij} = 0$. Even though $\overline{K}$ is not sparse and will often have negative eigenvalues, the "black-box" multiplication $x \mapsto \overline{K}x$ can be efficiently performed and can be used in a fast

sparse symmetric (since $\overline{K}^T = \overline{K}$) solver, such as MINRES. This gives one approach to generating a motion basis for unconstrained models, however the topic is a subject of ongoing research.

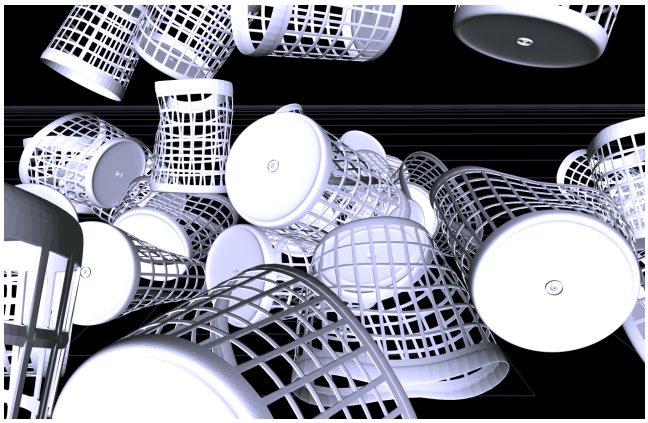

Figure 13: **Multibody dynamics simulation with large deformations:** *Motion basis ($r = 40$) uses linear modes $\Psi_7, \ldots \Psi_{26}$ and their derivatives.*

References

ALMROTH, B. O., STERN, P., AND BROGAN, F. A. 1978. Automatic Choice of Global Shape Functions in Structural Analysis. *AIAA Journal 16*, 5, 525–528.

BARAFF, D., AND WITKIN, A. 1992. Dynamic Simulation of Non-penetrating Flexible Bodies. *Computer Graphics (Proc. of ACM SIGGRAPH 92) 26(2)*, 303–308.

BARAFF, D., AND WITKIN, A. 1998. Large Steps in Cloth Simulation. In *Proc. of ACM SIGGRAPH 98*, 43–54.

BASDOGAN, C. 2001. Real-time Simulation of Dynamically Deformable Finite Element Models Using Modal Analysis and Spectral Lanczos Decomposition Methods. In *Medicine Meets Virtual Reality (MMVR'2001)*, 46–52.

BELYTSCHKO, T. 2001. *Nonlinear Finite Elements for Continua and Structures*. Wiley.

BONET, J., AND WOOD, R. D. 1997. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press.

BRIDSON, R., FEDKIW, R. P., AND ANDERSON, J. 2002. Robust Treatment of Collisions, Contact, and Friction for Cloth Animation. *ACM Trans. on Graphics 21*, 3, 594–603.

BRO-NIELSEN, M., AND COTIN, S. 1996. Real-time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation. *Comp. Graphics Forum 15*, 3, 57–66.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002. A Multiresolution Framework for Dynamic Deformations. In *Proc. of the Symp. on Comp. Animation 2002*, 41–48.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002. Interactive Skeleton-Driven Dynamic Deformations. *ACM Trans. on Graphics 21*, 3 (July), 586–593.

CHOI, M. G., AND KO, H.-S. 2005. Modal Warping: Real-Time Simulation of Large Rotational Deformation and Manipulation. In *IEEE Trans. on Vis. and Comp. Graphics*, vol. 11, 91–101.

COTIN, S., DELINGETTE, H., AND AYACHE, N. 1999. Realtime Elastic Deformations of Soft Tissues for Surgery Simulation. *IEEE Trans. on Vis. and Comp. Graphics 5*, 1, 62–73.

DEBUNNE, G., DESBRUN, M., CANI, M.-P., AND BARR, A. H. 2001. Dynamic Real-Time Deformations Using Space & Time Adaptive Sampling. In *Proc. of ACM SIGGRAPH 2001*, 31–36.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 1997. Dynamic Free-Form Deformations for Animation Synthesis. *IEEE Trans. on Vis. and Comp. Graphics 3*, 3, 201–214.

FUNG, Y. 1977. *A First Course in Continuum Mechanics*. Prentice-Hall, Englewood Cliffs, NJ.

GRINSPUN, E., KRYSL, P., AND SCHRÖDER, P. 2002. CHARMS: A Simple Framework for Adaptive Simulation. *ACM Trans. on Graphics 21*, 3 (July), 281–290.

HAUSER, K. K., SHEN, C., AND O'BRIEN, J. F. 2003. Interactive Deformation Using Modal Analysis with Constraints. In *Proc. of Graphics Interface*.

HOLZAPFEL, G. A. 2000. *Nonlinear Solid Mechanics*. Wiley.

IDELSOHN, S. R., AND CARDONA, A. 1985. A Load-dependent Basis for Reduced Nonlinear Structural Dynamics. *Computers and Structures 20*, 1-3, 203–210.

IDELSOHN, S. R., AND CARDONA, A. 1985. A Reduction Method for Nonlinear Structural Dynamic Analysis. *Computer Methods in Applied Mechanics and Engineering 49*, 253–279.

IRVING, G., TERAN, J., AND FEDKIW, R. 2004. Invertible Finite Elements for Robust Simulation of Large Deformation. In *Proc. of the Symp. on Comp. Animation 2004*, 131–140.

JAMES, D., AND FATAHALIAN, K. 2003. Precomputing Interactive Dynamic Deformable Scenes. In *Proc. of ACM SIGGRAPH 2003*, ACM, 879–887.

JAMES, D. L., AND PAI, D. K. 1999. ARTDEFO: Accurate Real Time Deformable Objects. In *Proc. of ACM SIGGRAPH 99*, vol. 33, 65–72.

JAMES, D. L., AND PAI, D. K. 2002. DyRT: Dynamic Response Textures for Real Time Deformation Simulation with Graphics Hardware. In *Proc. of ACM SIGGRAPH 2002*.

JAMES, D. L., AND PAI, D. K. 2004. BD-Tree: Output-Sensitive Collision Detection for Reduced Deformable Models. *ACM Trans. on Graphics 23*, 3 (Aug.), 393–398.

JAMES, D. L., BARBIČ, J., AND TWIGG, C. D. 2004. Squashing Cubes: Automating Deformable Model Construction for Graphics. In *Proc. of ACM SIGGRAPH Sketches and Applications*.

KRYSL, P., LALL, S., AND MARSDEN, J. E. 2001. Dimensional model reduction in non-linear finite element dynamics of solids and structures. *Int. J. for Numerical Methods in Engineering 51*, 479–504.

LUMLEY, J. L. 1967. The structure of inhomogeneous turbulence. In *Atmospheric turbulence and wave propagation*, 166–178.

METAXAS, D., AND TERZOPOULOS, D. 1992. Dynamic Deformation of Solid Primitives with Constraints. *Computer Graphics (Proc. of ACM SIGGRAPH 92) 26(2)*, 309–312.

MÜLLER, M., AND GROSS, M. 2004. Interactive Virtual Materials. In *Proc. of Graphics Interface 2004*, 239–246.

MÜLLER, M., DORSEY, J., MCMILLIAN, L., JAGNOW, R., AND CUTLER, B. 2002. Stable Real-Time Deformations. In *Proc. of the Symp. on Comp. Animation 2002*, 49–54.

MÜLLER, M., TESCHNER, M., AND GROSS, M. 2004. Physically-Based Simulation of Objects Represented by Surface Meshes. In *Proc. of Comp. Graphics Int. (CGI)*, 26–33.

NICKELL, R. E. 1976. Nonlinear Dynamics by Mode Superposition. *Computer Methods in Applied Mechanics and Engineering 7*, 107–129.

O'BRIEN, J., AND HODGINS, J. 1999. Graphical Modeling and Animation of Brittle Fracture. In *Proc. of ACM SIGGRAPH 99*, 111–120.

PAI, D. 2002. Strands: Interactive simulation of thin solids using Cosserat models. *Computer Graphics Forum 21*, 3, 347–352.

PENTLAND, A., AND WILLIAMS, J. July 1989. Good vibrations: Modal dynamics for graphics and animation. *Computer Graphics (Proc. of ACM SIGGRAPH 89) 23*, 3, 215–222.

PICINBONO, G., DELINGETTE, H., AND AYACHE, N. 2001. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *IEEE Int. Conf. on Robotics and Automation 2001*.

SHABANA, A. A. 1990. *Theory of Vibration, Volume II: Discrete and Continuous Systems*. Springer–Verlag, New York, NY.

SHINYA, M., AND FOURNIER, A. 1992. Stochastic motion - Motion under the influence of wind. *Comp. Graphics Forum*, 119–128.

STAM, J. 1997. Stochastic Dynamics: Simulating the Effects of Turbulence on Flexible Structures. *Comp. Graphics Forum 16(3)*.

TERZOPOULOS, D., AND WITKIN, A. 1988. Physically Based Models with Rigid and Deformable Components. *IEEE Comp. Graphics & Applications 8*, 6, 41–51.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically Deformable Models. *Computer Graphics (Proc. of ACM SIGGRAPH 87) 21(4)*, 205–214.

WRIGGERS, P. 2002. *Computational Contact Mechanics*. John Wiley & Sons, Ltd.

ZHUANG, Y., AND CANNY, J. 2000. Haptic Interaction with Global Deformations. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*.