큰 회전 변형 및 조작의 실시간 시뮬레이션

최 민 규[0] 고 형 석
{mgchoi,ko}@graphics.snu.ac.kr

Real-Time Simulation of Large Rotational Deformation and Manipulation

Min Gyu Choi[0] Hyeong-Seok Ko

Abstract

This paper proposes a real-time technique for simulating large rotational deformations. Modal analysis based on a linear strain tensor has been shown to be suitable for real-time simulation, but is accurate only for moderately small deformations. In the present work, we identify the rotational component of an infinitesimal deformation, and extend linear modal analysis to track that component. We then develop a procedure to integrate the small rotations occurring at the nodal points. An interesting feature of our formulation is that it can implement both position and orientation constraints in a straightforward manner. These constraints can be used to interactively manipulate the shape of a deformable solid by dragging/twisting a set of nodes. Experiments show that the proposed technique runs in real-time even for a complex model, and that it can simulate large bending and/or twisting deformations with acceptable realism.

1 Introduction

Everything in this world deforms. In many objects or creatures, deformation is such a conspicuous characteristic that their synthetic versions look quite unnatural if the deformation process is not properly simulated. Therefore, modeling of deformation is an important aspect of computer animation production. This paper presents a physically-based technique for real-time simulation of elastodynamic solids, attached to rigid supports and excited by their rigid motions and/or external forces such as gravity.

It is a well-established approach to model elastic solids as continuums and solve their governing equations numerically using finite element methods. Green's strain tensor, which consists of linear terms and a nonlinear term, has been a common choice for simulating large deformations. However, the nonlinear term gives rise to inherent numerical instabilities, making it necessary to use a small time step. As a consequence, simulation with Green's strain tensor can involve a massive amount of computation, hampering its practical use in animation production.

For moderately small deformations, the omission of the nonlinear term makes the stiffness matrix constant and numerically well-conditioned, and thus significantly increases the stability and speed of simulation. The computational load can be further reduced remarkably by employing *modal analysis* [13], which makes use of the precomputed deformation modes that span the free vibration of the elastic model in linear combination. This technique can also synthesize geometrically complex deformations with negligible main CPU costs on programmable graphics hardware [8]. However, when applied to bending or twisting deformations of relatively large magnitudes, the results can look quite unnatural as illustrated in Figure 4. These unnatural results are due to the omission of the nonlinear term, which is not negligible for such deformations.

In this paper, we propose a novel technique that can generate visually plausible shapes of elastodynamic solids undergoing large rotational deformations, while retaining the computational stability and speed of modal analysis. To exploit the modal analysis framework, we omit the nonlinear term during the initial setup, which corresponds to precomputing the modal vibration modes at the rest state. When the simulation is run, however, we keep track of the local rotations that occur during the deformation, based on the infinitesimal rotation tensor. Then, at each time step we warp the precomputed modal basis in accordance with the local rotations of the mesh nodes. The rest of the method is basically the same as in linear modal analysis. Therefore, as in [8], our method can simulate dynamic deformations in real-time by employing a programmable graphics hardware, but with an extended coverage of deformations.

2 Related Work

Since the pioneering work of Terzopoulos et al. [18], much effort has been devoted to simulating deformable objects. Past studies in this area have had two central aims: to speed up the simulation and/or increase the realism of the result. A comprehensive survey on this subject can be found in [4].

The speed and realism of simulations, which usually trade off each other, are heavily dependent on how the nonlinearities are handled. If realism is important, Green's quadratic strain tensor could be used, which produces realistic results even for large deformations. However, it requires the use of small time steps for numerical stabilities. Several methods have been proposed to reduce the computation time [1, 2, 5, 17]. However, the speed-up achieved by those methods is limited, because they must still deal with the inherent problems resulting from the nonlinearities.

The computation time can be greatly reduced by adopting the modal analysis with a linear strain tensor. Since Pentland and Williams [13] introduced this technique to the computer graphics community, it has been used for modeling the dynamic movements of trees in turbulent wind [16], and for generating sounds corresponding to the behavior of deformable objects [12]. In particular, James and Pai [8] showed that the deformation of human skin excited by rigid body motion can be generated in real-time on a programmable graphics hardware. Hauser et al. [6] further addressed the manipulation constraints, and combined modal analysis with rigid body simulation to handle free-floating deformable objects. Although modal analysis significantly accelerates the simulation, it generates noticeable artifacts when applied to large deformations due to the linearization. Here we propose a technique that eliminates the linearization artifacts while retaining the efficiency of the modal analysis.

To reduce the linearization artifacts, Terzopoulos and Witkin [19] modeled the deformation relative to a frame of reference that captures the gross motion of the deformable solid. Capell et al. [1] developed a method that divides an articulated deformable characters into overlapping solids. However, large deformations within the solid are also susceptible to the linearization artifact. To address large relative rotational deformations within a single object, Müller et al. [10] proposed the stiffness warping method that tracks the rotation of each node and warps the stiffness matrix. Our method is similar to that of Müller et al. inasmuch as rotations are handled separately, but differs in that, whereas they solved the governing equation directly, we use linear modal analysis on account of its efficiency, stability, and hardware utilization. Unlike element-wise rotation employed in the corotational methods [3], node-wise rotation of the stiffness matrix can produce a spurious ghost force when applied to a free-floating deformable object. However, our work is currently focused on a deformable object attached to a rigid support, thus the ghost force effect does not show up as described in [10].

Recently, James and Fatahalian [7] proposed a nonlinear modal analysis method for large deformations by employing data-driven tabulation of the state space dynamics and dimensional model reduction of the deformed shapes. Because the tabulation cannot be performed for all possible system responses, their method confines user interactions to certain types of movements. They reported that the precomputation for the dinosaur model shown in Figure 8 took about 30 hours. In comparison, our method is formulated by adding simple extensions to linear modal analysis. Consequently, it does not entail long precomputation times, nor does it restrict the types of user interactions. However, self-collisions and global scene illumination cannot be precomputed in our method, which was possible in [7].

3 Small Deformations

The nonlinear term in Green's strain tensor is responsible for rotational deformations. Even though a linear strain tensor does not properly model the rotational deformation, fortunately, investigating the kinematics of deformation provides a clue to lessen such an inability; Every infinitesimal deformation can be decomposed into a rotation followed by a strain [15]. This forms the basis of our work. Specifically, at every time step of the deformation simulation, we first identify (small) rotations occurring over the material points, and then integrate the effects of those rotations to obtain the deformed shape.

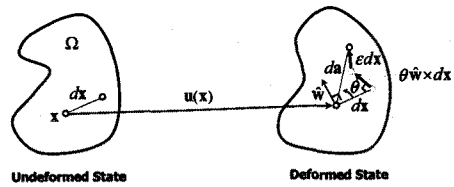

Figure 1: Kinematics of infinitesimal deformation.

This section commences with an investigation of the kinematics of infinitesimal deformation to show how such deformations can be decomposed into a strain and a rotation. This analysis is entirely based on the mechanics literature [15]. We then show how this decomposition can be used to extend modal analysis so that it keeps track of rotations, while still retaining the basic framework of modal analysis. The method for integrating the effects of rotations will be presented in the next section.

3.1 Kinematics of Infinitesimal Deformation

Before introducing the decomposition of infinitesimal deformations, we first define the necessary notations. Suppose that $\mathbf{x} \in \mathbb{R}^3$ denotes the position of a material point of an elastic solid in the undeformed state, which moves to a new position $\mathbf{a}(\mathbf{x})$ due to a subsequent deformation. To focus on the displacements caused by the deformation, we make use of the *displacement field*, $\mathbf{u} : \mathbb{R}^3 \to \mathbb{R}^3$, such that

$$\mathbf{a}(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{x}), \quad \mathbf{x} \in \Omega,$$

where Ω is the domain of the solid. Then, differentiating both sides of the above equation with respect to $\mathbf{x}$ gives a differential relation that gives the position to which a material point neighboring $\mathbf{x}$ will be mapped by the deformation:

$$d\mathbf{a} = (\mathbf{I} + \nabla\mathbf{u})d\mathbf{x}, \tag{1}$$

where $\nabla\mathbf{u}$ is the Jacobian of $\mathbf{u}$.

The *infinitesimal strain tensor* ε, which measures the change in the squared length of $d\mathbf{x}$ during an infinitesimal deformation (i.e., $\|\nabla\mathbf{u}\| \ll 1$), is defined by

$$\varepsilon \triangleq \tfrac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}}).$$

Then, we can decompose $\nabla\mathbf{u}$ as

$$\nabla\mathbf{u} = \tfrac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}}) + \tfrac{1}{2}(\nabla\mathbf{u} - \nabla\mathbf{u}^{\mathrm{T}}) \triangleq \varepsilon + \omega. \tag{2}$$

Here, the *infinitesimal rotation tensor* ω can be further expressed in terms of the curl of the displacement field:

$$\omega = \tfrac{1}{2}(\nabla\mathbf{u} - \nabla\mathbf{u}^{\mathrm{T}}) = \tfrac{1}{2}(\nabla \times \mathbf{u})\times \triangleq \mathbf{w}\times, \tag{3}$$

where $\mathbf{z}\times$ denotes the standard skew symmetric matrix of vector $\mathbf{z}$. Therefore, $\mathbf{w} \triangleq \frac{1}{2}(\nabla \times \mathbf{u})$ can be viewed as a rotation vector that causes rotation of the material points at and near $\mathbf{x}$ by angle $\theta = \|\mathbf{w}\|$ about the unit axis $\hat{\mathbf{w}} = \mathbf{w}/\|\mathbf{w}\|$.

Substitution of Equations (2) and (3) into Equation (1) gives

$$d\mathbf{a} = d\mathbf{x} + \underbrace{\varepsilon d\mathbf{x}}_{\text{strain}} + \underbrace{\theta\hat{\mathbf{w}} \times d\mathbf{x}}_{\text{rotation}},$$

which shows that an infinitesimal deformation consists of a strain and a rotation. This decomposition, illustrated in Figure 1, has the practical benefit that, for small deformations, it is possible to keep track of the rotation of each material point by calculating the curl of the displacement field, $\mathbf{w} = \frac{1}{2}\nabla \times \mathbf{u}$.

3.2 Extended Modal Analysis

This section presents how we extend the conventional modal analysis so that it keeps track of the rotation experienced by each material point during deformation. First, we present a brief introductory description of modal analysis and finite element methods; detailed explanations of these techniques can be found in texts such as [14, 20].

The governing equation for a finite element model can be written as

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{F}, \tag{4}$$

where $\mathbf{u}(t)$ is a $3n$-dimensional vector that represents the displacements of the n nodes from their original positions, and $\mathbf{F}(t)$ is a vector that represents the external forces acting on the nodes. The mass, damping, and stiffness matrices $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ are independent of time and are completely characterized at the rest state, under the commonly adopted assumption that $\mathbf{C} = \xi\mathbf{M} + \zeta\mathbf{K}$, where ξ and ζ are scalar weighting factors.

Modal Displacement. In general, $\mathbf{M}$ and $\mathbf{K}$ are not diagonal, and thus Equation (4) is a coupled system of ordinary differential equations (ODEs). Let Φ and Λ be the solution matrices of the generalized eigenvalue problem, $\mathbf{K}\Phi = \mathbf{M}\Phi\Lambda$, such that $\Phi^{\mathrm{T}}\mathbf{M}\Phi = \mathbf{I}$ and $\Phi^{\mathrm{T}}\mathbf{K}\Phi = \Lambda$. Since the columns of Φ form a basis of the $3n$-dimensional space, $\mathbf{u}$ can be expressed as a linear combination of the columns:

$$\mathbf{u}(t) = \Phi\mathbf{q}(t). \tag{5}$$

Here, Φ is the *modal displacement matrix*, of which the i-th column represents the i-th mode shape, and $\mathbf{q}(t)$ is a vector containing the corresponding modal amplitudes as its components. By examining the eigenvalues we can take only dominant m columns of Φ, significantly reducing the amount of computation. In the following, Φ denotes the $3n \times m$ submatrix.

Substitution of Equation (5) into Equation (4) followed by a premultiplication of Φ^{T} decouples Equation (4) as

$$\mathbf{M}_{\mathbf{q}}\ddot{\mathbf{q}} + \mathbf{C}_{\mathbf{q}}\dot{\mathbf{q}} + \mathbf{K}_{\mathbf{q}}\mathbf{q} = \Phi^{\mathrm{T}}\mathbf{F}, \tag{6}$$

where $\mathbf{M}_{\mathbf{q}} = \mathbf{I}$, $\mathbf{C}_{\mathbf{q}} = (\xi\mathbf{I} + \zeta\Lambda)$, and $\mathbf{K}_{\mathbf{q}} = \Lambda$ are now all diagonal. $\Phi^{\mathrm{T}}\mathbf{F}$ is called the modal force. The above decoupling allows the motion components due to individual modes to be computed independently and combined by linear superposition.

Modal Rotation. We now develop a procedure to represent the rotational part, $\mathbf{w}(t)$, in terms of $\mathbf{q}(t)$. $\mathbf{w}(t)$ is a $3n$ dimensional vector formed by concatenating all of the 3 dimensional rotation vectors, each of which is formed by taking the curl of the displacement field $\mathbf{u}$ at each node, as described in Section 3.1.

For simplicity, we use linear tetrahedral elements in Equation (4). Let $\mathbf{u}_{e,j}$ ($j \in [1,4]$) be the vertex displacement of tetrahedron Ω_e, and let $\mathbf{u}_e = [\mathbf{u}_{e,1}^{\mathrm{T}}|\mathbf{u}_{e,2}^{\mathrm{T}}|\mathbf{u}_{e,3}^{\mathrm{T}}|\mathbf{u}_{e,4}^{\mathrm{T}}]^{\mathrm{T}}$. Then, the displacement of material point $\mathbf{x} \in \Omega_e$ is given by $\mathbf{u}(\mathbf{x}) = \mathbf{H}_e(\mathbf{x})\mathbf{u}_e$, where $\mathbf{H}_e(\mathbf{x})$ is the linear shape function of the element. Substituting this into Equation (3) yields the rotation vector for $\mathbf{x}$:

$$\mathbf{w}_e(\mathbf{x}) = \tfrac{1}{2}(\nabla\times)\mathbf{H}_e(\mathbf{x})\mathbf{u}_e \triangleq \mathbf{W}_e\mathbf{u}_e. \tag{7}$$

Since $\mathbf{H}_e(\mathbf{x})$ is a linear function of $\mathbf{x}$, $\mathbf{W}_e \triangleq \frac{1}{2}(\nabla\times)\mathbf{H}_e(\mathbf{x})$ is constant, and thus $\mathbf{w}_e(\mathbf{x})$ is uniform over Ω_e. For the rotation vector of a node, we use the average of the rotation vectors of all the tetrahedra sharing the node.

Based on the above discussion, we can now assemble the $\mathbf{W}_e$ of all the elements to form the global matrix $\mathbf{W}$ such that $\mathbf{W}\mathbf{u}(t)$ gives the composite vector $\mathbf{w}(t)$ that we are looking for. Finally, expanding $\mathbf{u}(t)$ with Equation (5) gives

$$\mathbf{w}(t) = \mathbf{W}\Phi\mathbf{q}(t) \triangleq \Psi\mathbf{q}(t). \tag{8}$$

Both $\mathbf{W}$ and Φ are characterized by the deformable mesh at the rest state, and are thus constant over time. Therefore we can precompute Ψ. Equation (8) shows that, as in the displacement (Equation (5)), we can represent the rotational component of deformation in terms of $\mathbf{q}(t)$. We call Ψ the *modal rotation matrix*. It should be noted that both of the modal matrices are meaningful only for moderately small deformations.

4 Large Deformations

Equation (8) provides an efficient way to keep track of the rotations occurring at each node over time. However, such rotations are not yet reflected in the calculation of the displacement field $\mathbf{u}(t)$. Therefore, simulations based on Equations (5), (6), and (8) in Section 3.2 will not produce proper rotational deformations. In this section, we develop a method to integrate the effect of the rotational part into the calculation of $\mathbf{u}(t)$.

To accommodate large deformations, the stiffness matrix $\mathbf{K}$ in Equation (4) should be replaced by $\mathbf{K}(\mathbf{u})$. Therefore, we must deal with a governing equation of the form,

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{K}(\mathbf{u})\mathbf{u} = \mathbf{F}. \tag{9}$$

Let $\mathbf{u}_i(t)$ be the i-th 3-dimensional vector of $\mathbf{u}(t)$. Then, $\mathbf{u}_i(t)$ represents the displacement of the i-th node from its original position, measured in the global coordinate frame. In order to measure the local rotations with respect to the global coordinate frame, we embed a local coordinate frame at each node i as shown in Figure 2, such that at the initial state it is aligned with the global coordinate frame. Hereafter, we use the rotation matrix $\mathbf{R}_i(t)$ to represent the orientation of the local coordinate frame embedded at i-th node.

Let $\dot{\mathbf{u}}_i^L(t)dt$ be the differential displacement of the i-th node measured from the i-th local coordinate frame at time t. Then, the finite displacement $\mathbf{u}_i(t)$ measured from the global coordinate frame is given by $\mathbf{u}_i(t) = \int_0^t \mathbf{R}_i(\tau)\dot{\mathbf{u}}_i^L(\tau)d\tau$. This procedure must be carried out for every node. To this end, we form the

block-diagonal matrix $\mathbf{R} = [\delta_{ij}\mathbf{R}_i]$, where $1 \le i, j \le n$ and δ_{ij} is the Kronecker delta. Then, n equations can be assembled into a single equation,

$$\mathbf{u}(t) = \int_0^t \mathbf{R}(\tau)\dot{\mathbf{u}}^L(\tau)d\tau. \tag{10}$$

This equation shows how the effect of the rotations occurring at the nodal points can be integrated. The remainder of this section describes the procedure used to compute the above integration.

4.1 Modal Analysis in Local Frames

Equation (10) guides us that, instead of solving Equation (9) for $\mathbf{u}$, we need to convert the equation into a form that can be solved for $\mathbf{u}^L$. By premultiplying both sides of Equation (9) with $\mathbf{R}^T$, we obtain

$$\mathbf{R}^T\left[\mathbf{M}\ddot{\mathbf{u}} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{K}(\mathbf{u})\mathbf{u}\right] = \mathbf{R}^T\mathbf{F}. \tag{11}$$

The following two assumptions must then be made to convert Equation (11) to the form shown in Equation (15).

The first assumption is that the mesh being simulated is sufficiently fine that the approximation,

$$\mathbf{R}^T\mathbf{M} \approx \mathbf{M}\mathbf{R}^T, \tag{12}$$

is valid. Experimental results showed that the approximation error did not significantly impact the visual realism of the simulation, even for coarse meshes. Differentiating both sides of Equation (10) with respect to time, we obtain $\dot{\mathbf{u}} = \mathbf{R}\dot{\mathbf{u}}^L + \dot{\mathbf{R}}\mathbf{u}^L$. Thus,

$$\mathbf{R}^T\mathbf{M}\ddot{\mathbf{u}} \approx \mathbf{M}\mathbf{R}^T\ddot{\mathbf{u}} = \mathbf{M}\ddot{\mathbf{u}}^L + \mathbf{M}\mathbf{R}^T\dot{\mathbf{R}}\dot{\mathbf{u}}^L, \tag{13}$$

where $\mathbf{M}\mathbf{R}^T\dot{\mathbf{R}}\dot{\mathbf{u}}^L$ is the Coriolis force resulting from the rotational movements of the local coordinate frames. If the rotational movements occur at a moderate rate, the Coriolis force is negligible compared to gravity. Thus, we omit the Coriolis force in the subsequent formulation.

The second assumption is that the nonlinear elastic forces can be approximated by measuring the linear elastic forces in the local coordinate frames, but resolving them in the global coordinate frame:

$$\mathbf{K}(\mathbf{u})\mathbf{u} \approx \mathbf{R}\mathbf{K}\mathbf{u}^L \quad \Leftrightarrow \quad \mathbf{R}^T\mathbf{K}(\mathbf{u})\mathbf{u} \approx \mathbf{K}\mathbf{u}^L. \tag{14}$$

This assumption is similar to the *stiffness warping* proposed in [10], where $\mathbf{K}(\mathbf{u})\mathbf{u} \approx \mathbf{R}\mathbf{K}(\mathbf{R}^T\mathbf{a} - \mathbf{x})$.

Now, we are ready to approximate Equation (11) by a linear equation for modal analysis in the local coordinate frames. Substituting Equations (13) and (14) into Equation (11), we obtain

$$\mathbf{M}\ddot{\mathbf{u}}^L + \mathbf{C}\dot{\mathbf{u}}^L + \mathbf{K}\mathbf{u}^L = \mathbf{R}^T\mathbf{F}, \tag{15}$$

where we use the proportional damping $\mathbf{C} = \xi\mathbf{M} + \zeta\mathbf{K}$. This linear elastodynamic equation for $\mathbf{u}^L$ is same as Equation (4), except that the external force acting on each node needs to be pre-rotated in accordance with its local coordinate frame. Therefore, it is straightforward to reduce Equation (15) into a set of decoupled ODEs. The modal displacement matrix Φ obtained in Section 3.2 gives the relationship

$$\mathbf{u}^L(t) = \Phi\mathbf{q}(t). \tag{16}$$

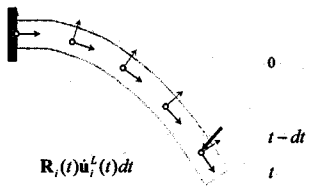

Figure 2: Local coordinate frames attached to the nodes.

Based on this relationship, we can replace $\mathbf{u}^L$ in Equation (15) with $\Phi\mathbf{q}(t)$, and, after premultiplying both sides of the same equation with Φ^T, we obtain

$$\mathbf{M}_\mathbf{q}\ddot{\mathbf{q}} + \mathbf{C}_\mathbf{q}\dot{\mathbf{q}} + \mathbf{K}_\mathbf{q}\mathbf{q} = \Phi^T(\mathbf{R}^T\mathbf{F}). \tag{17}$$

The above decoupled ODEs can be solved numerically using semi-implicit integration. By manipulating Equation (17), we obtain the following expressions for $\mathbf{q}^k = \mathbf{q}(t^k)$ and $\dot{\mathbf{q}}^k = \dot{\mathbf{q}}(t^k)$:

$$\begin{aligned}\mathbf{q}^k &= \alpha\mathbf{q}^{k-1} + \beta\dot{\mathbf{q}}^{k-1} + \gamma\left(\mathbf{R}^{k-1}\Phi\right)^T\mathbf{F}^{k-1}, \\ \dot{\mathbf{q}}^k &= h^{-1}\left[(\alpha - \mathbf{I})\mathbf{q}^{k-1} + \beta\dot{\mathbf{q}}^{k-1} + \gamma\left(\mathbf{R}^{k-1}\Phi\right)^T\mathbf{F}^{k-1}\right],\end{aligned} \tag{18}$$

where α, β, and γ are diagonal matrices, the i-th components of which are respectively

$$\alpha_i = 1 - \frac{h^2 k_i}{d_i}, \quad \beta_i = h\left(1 - \frac{hc_i + h^2 k_i}{d_i}\right), \quad \gamma_i = \frac{h^2}{d_i},$$

in which h is the time step size, $d_i = m_i + hc_i + h^2 k_i$ with m_i, c_i, and k_i representing the diagonal entries of $\mathbf{M}_\mathbf{q}$, $\mathbf{C}_\mathbf{q}$, and $\mathbf{K}_\mathbf{q}$, respectively.

4.2 Formulation of Modal Warping

We now need to evaluate Equation (10) for the finite displacement $\mathbf{u}^k$ at a time step k. A straightforward numerical integration gives an incremental formulation that sums up the current small displacement to the previous result. Since numerical errors can be accumulated through the simulation, this approach gives rise to an hysteresis effect such that the deformable solid cannot return to the initial state after all the external forces are removed.

To circumvent such an hysteresis effect, we analytically evaluate Equation (10) by taking a quasi-static approach that ramps $\mathbf{q}(t)$ from 0 to $\mathbf{q}^k$ at each time step k: $\mathbf{q}(t) = \frac{t}{t^k}\mathbf{q}^k, 0 \le t \le t^k$. Then, the history of $\mathbf{w}(t)$, which determines that of $\mathbf{R}(t)$, is also represented as a linear function. $\mathbf{w}(t) = \frac{t}{t^k}\Psi q^k, 0 \le t \le t^k$. Now, $\mathbf{R}(t)$ can be obtained by simply converting $\mathbf{w}(t)$ into the $3n \times 3n$ block-diagonal rotation matrix. Finally, Equation (10) can be evaluated analytically as follows:

$$\mathbf{u}^k = \int_0^{t^k} \mathbf{R}(t)\Phi\dot{\mathbf{q}}(t)dt = \tilde{\mathbf{R}}^k\Phi\mathbf{q}^k, \tag{19}$$

where $\tilde{\mathbf{R}}^k \triangleq \frac{1}{t^k}\int_0^{t^k}\mathbf{R}(t)dt$. The procedure for computing $\tilde{\mathbf{R}}$ is given in Appendix A.

Equation (19) implies a new deformation scheme; $\tilde{\Phi}^k \triangleq \tilde{\mathbf{R}}^k\Phi$ can be regarded as a warped version of the original modal basis Φ. The columns of $\tilde{\Phi}^k$ give the mode shapes at the time step k,

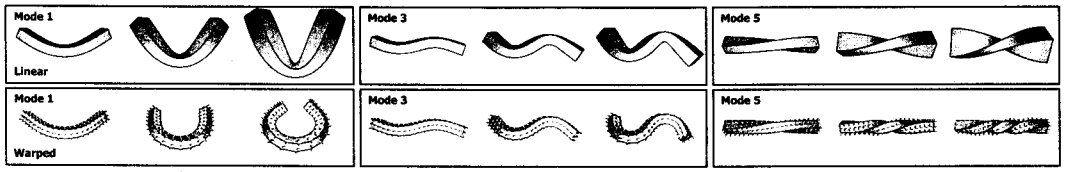

Figure 3: Evolution of mode shapes in linear modal analysis (top row) and modal warping (bottom row).

in which rotations occurred at the nodal points have been accumulated. Figure 3 shows the evolution of three selected mode shapes over time for the case of a bar. The new method works in basically the same way as the linear modal analysis, except that it uses a warped modal basis instead of a fixed linear modal basis.

5 Manipulation Constraints

Thus far, we have discussed the dynamics of an unconstrained elastic body. Motivated by the work of Hauser et al. [6] on positional constraints in a linear modal analysis setting, we extend our deformation scheme to cope with manipulation constraints to allow, for example, dragging/twisting of some nodes to a certain position and/or orientation (see Figure 5). Note that the orientation constraints for a deformable body have not been addressed in previous studies. Such constraints are possible in our formulation because it explicitly takes into account the mean orientation of each node, based on the infinitesimal strain tensor analysis.

5.1 Position Constraints

Let λ be the number of constrained points, and let $\mathbf{u}_c^k$ be the 3λ-dimensional vector consisting of the desired displacements of the constrained nodes at a time step k. Then, the constraint equation can be written as

$$\mathbf{u}_c^k = \tilde{\mathbf{\Phi}}_c^k \mathbf{q}_c^k = \tilde{\mathbf{R}}_c^k \Phi_c \mathbf{q}_c^k, \tag{20}$$

where $\mathbf{q}_c^k$ is the unknown modal amplitude vector, Φ_c is the $3\lambda \times m$ matrix obtained from Φ by taking only the rows for the constrained nodes, and $\tilde{\mathbf{R}}_c^k$ is the $3\lambda \times 3\lambda$ block-diagonal matrix obtained from $\tilde{\mathbf{R}}^k$ by taking only the part corresponding to the constrained nodes. Let the $3n$-dimensional vector $\tilde{\mathbf{F}}^{k-1}$ represent the unknown constraint force measured in the global coordinate frame. Then, $\mathbf{q}_c^k$ should satisfy not only Equation (20) but also Equation (18) when this additional force is applied. That is,

$$\mathbf{q}_c^k = \mathbf{q}^k + \gamma\left(\mathbf{R}^{k-1}\Phi\right)^{\mathrm{T}} \tilde{\mathbf{F}}^{k-1}, \tag{21}$$

where $\mathbf{q}^k$ represents the modal amplitude vector in the unconstrained case. Here, the forces do not need to be exerted only at the constrained nodes, because exerting forces at some unconstrained nodes can still cause the constrained nodes to be positioned at the specified locations. We will refer to the nodes at which forces are exerted as *exercised* nodes.

Let μ be the number of exercised nodes. In $\tilde{\mathbf{F}}^{k-1}$, the portion corresponding to the unexercised nodes should be zero. Let $\mathbf{F}_x^{k-1}$ be the 3μ-dimensional vector consisting only of the constraint forces acting on the exercised nodes, which can be obtained by removing the 3-dimensional vectors corresponding to the unexercised nodes from $\tilde{\mathbf{F}}^{k-1}$. Then, we can rewrite Equation (21) in terms of $\mathbf{F}_x^{k-1}$,

$$\mathbf{q}_c^k = \mathbf{q}^k + \gamma\left(\mathbf{R}_x^{k-1}\Phi_x\right)^{\mathrm{T}} \mathbf{F}_x^{k-1}, \tag{22}$$

where Φ_x is the $3\mu \times m$ matrix obtained from Φ by taking only the rows for the exercised nodes, and the $3\mu \times 3\mu$ block-diagonal matrix $\mathbf{R}_x^{k-1}$ is obtained from $\mathbf{R}^{k-1}$ by taking only the part corresponding to the exercised nodes. Finally, substituting Equation (22) into Equation (20), we obtain the constraint force:

$$\mathbf{F}_x^{k-1} = \mathbf{R}_x^{k-1} \mathbf{A}_p^{\dagger} \mathbf{b}_p, \tag{23}$$

where $\mathbf{A}_p = \tilde{\mathbf{R}}_c^k \Phi_c \gamma \Phi_x^{\mathrm{T}}$, $\mathbf{b}_p = \mathbf{u}_c^k - \tilde{\mathbf{R}}_c^k \Phi_c \mathbf{q}^k$, and $(\cdot)^{\dagger}$ denotes the pseudo-inverse of a matrix. This constraint force can now be applied to the exercised nodes through Equation (22) to yield the desired modal amplitude vector.

5.2 Orientation Constraints

Orientation constraints can be implemented in a similar way to the position constrains. Let η be the number of constrained nodes, and let the 3η-dimensional vector $\mathbf{w}_c^k$ represent the desired rotations of the constrained nodes at a time step k. Then, the constraint equation can be written as

$$\mathbf{w}_c^k = \Psi_c \mathbf{q}_c^k, \tag{24}$$

where $\mathbf{q}_c^k$ is the unknown modal amplitude vector and Ψ_c is the $3\eta \times m$ matrix obtained from the modal rotation matrix Ψ by taking only the rows corresponding to the constrained nodes. Then, as in the position constraint case, $\mathbf{q}_c^k$ should simultaneously satisfy Equations (22) and (24). By manipulating these two equations, we obtain the equation for the constraint force:

$$\mathbf{F}_x^{k-1} = \mathbf{R}_x^{k-1} \mathbf{A}_o^{\dagger} \mathbf{b}_o, \tag{25}$$

where $\mathbf{A}_o = \Psi_c \gamma \Phi_x^{\mathrm{T}}$ and $\mathbf{b}_o = \mathbf{w}_c^k - \Psi_c \mathbf{q}^k$. Finally, we can apply the above constraint force to the exercised nodes through Equation (22) to obtain the desired modal amplitude vector.

6 Experimental Results

Our deformation scheme is implemented as an Alias® MAYA™ plugin for a Microsoft® Windows^{XP} environment, and also as a stand-alone application to exploit a programmable graphics hardware through *n*VIDIA® Cg and Microsoft® DirectX® API. Tetrahedral meshes were generated using public domain software NETGEN. To obtain the m dominant eigenvalues of large sparse

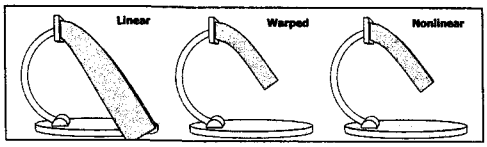

Figure 4: A bar deformed by traditional linear modal analysis (left), by modal warping (center), and by nonlinear FEM (right).

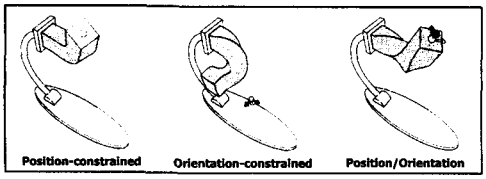

Figure 5: Manipulating a Bar.

square matrices and the corresponding eigenvectors, we used the built-in C++ math function `eigs` in MATLAB®, which is based on the ARPACK [9] eigenvalue solver. All experiments were performed on a PC with an Intel® Pentium®4 3.2GHz processor, 1GB memory, and an *n*VIDIA® GeForce® FX 5900 Ultra 256MB graphics card. We used the time step size of $h = 1/30$ second in all experiments reported in this section. Model statistics and performance data are summarized in Table 1.

Comparison to Other Methods. To evaluate the performance of modal warping, we simulated the deformation of a long bar under a gravitational field using traditional linear modal analysis, modal warping, and nonlinear FEM. We employed explicit integration with the nonlinear FEM [11] for accurate simulation, and set the time step size $h = 1/1000$ second for numerical stabilities. It is clear from the simulation results, shown in Figure 4, that the linearization in modal analysis leads to artifacts that are not observed in the cases of modal warping and nonlinear FEM.

Manipulation Test. This experiment demonstrates the manipulation capability of our technique. Figure 5 shows, from left to right, the resultant deformations in the cases of only position constraints, only orientation constraints, and both position and orientation constraints. For the case of position constraints, the constrained node was identical to the exercised node. For the case of orientation constraints, however, the set of exercised nodes was extended to the nodes neighboring the constrained node.

Manipulation Constraints for Animating Deformable Body Parts. To demonstrate how the manipulation constraints can be used to animate deformable parts of a character, we simulated a character whose only deformable part was its potbellied torso (Figure 6(a)). As the character made a dance motion, the potbelly made a dynamic passive deformation, excited by the gross motion of the character as in [8]. All the mesh nodes contained in the rigid pelvis at the initial setup follows its rigid motion. As shown in Figure 6(b), the deformable solid is attached to the skeleton by two position constraints (the yellow spheres) and one position/orientation constraint (the RGB axes).

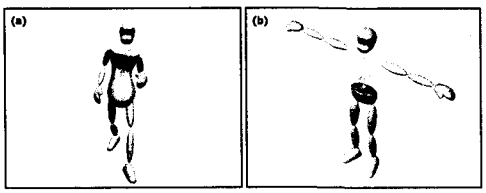

Figure 6: Constraint-driven animation of a character.

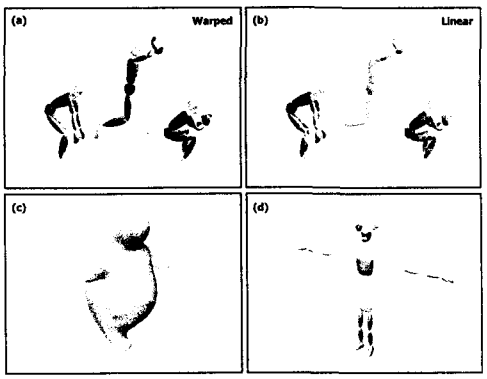

Figure 7: Constraint-based motion retargeting.

Manipulation Constraints for Motion Retargeting. The manipulation constraints can also be used to retarget a motion of an articulated character to that of a deformable character. To demonstrate this, we consider two examples. In the first example, a jumping motion of an articulated character is retargeted to a jelly box, as shown in Figure 7(a). As in the character considered above, the nodes contained in the pelvis are follows its rigid motion. The motion of the jelly box is driven by the movements of the feet and head of the articulated character. Three snapshots taken during this experiment are shown in Figure 7(a). For comparison, we also applied the traditional modal analysis to this case (see Figure 7(b)). In the second example, we applied a dance motion to the flubber shown in Figure 7(c). Because this character has a more articulated shape than that in the previous example, more constraints were used to properly map between the articulated and deformable characters. We placed one position/orientation constraint at the head, and five position constraints at the torso, elbows, and feet (see Figure 7(d)).

Simulation of Large Models. When the modal warping technique is applied to a large model such as the dinosaur model shown in Figure 8, simulating the deformation is not the bottleneck; surprisingly, the dynamic update of the vertex coordinates for display is the slowest procedure. To achieve real-time simulation of the model, we employed a programmable graphics hardware as in [8]. We applied our hardware implementation to the rubber dinosaur model previously used in [7]. The total precomputation time for the finite element method and the modal analysis was less than 2 seconds, and the simulation, including the display, ran at about 100 fps. The result was quite realistic, even for cases involving large deformations. Using our method,

Table 1: Model statistics and performance data.

Example	Figure	Model statistics					Constr.		Precomputation (sec)		Computation (sec/fr)		FPS	
		Vertices	Faces	Nodes	Elements	Modes	λ	η	FEM	MA	ODE	Constr.	Maya	Cg
Bar	4 & 5	354	352	99	240	8	1	1	0.046	0.063	0.001	0.001	60.0	·
Potbelly	6	1026	1056	363	1110	16	3	1	0.062	0.484	0.001	0.001	60.0	·
Jelly Box	7(a)	1642	1640	400	1440	32	2	2	0.062	1.156	0.001	0.001	60.0	·
Flubber	7(c)	2802	2800	552	1513	64	6	1	0.078	2.062	0.001	0.001	60.0	·
Dinosaur	8	28098	56192	1883	5484	8	1	1	0.312	1.422	0.002	0.001	11.9	103.8

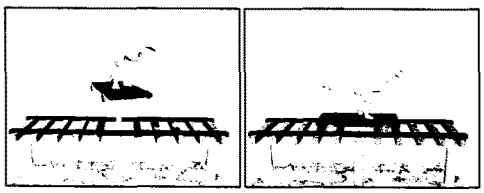

Figure 8: Dynamic deformation and manipulation of a dinosaur.

the types of interactions allowed during runtime did not need to be restricted; for example, the tail of the dinosaur could be manipulated interactively.

7 Conclusion

The present work extends traditional linear modal analysis to create a novel deformation technique that combines the merits of this type of analysis, in particular its ability to give real-time performance [13, 8], with the ability to accommodate large rotational deformations. An interesting feature of our technique is that it supports both position and orientation constraints, and hence could be used for interactively manipulating the shape of a deformable solid. The constraints can also be used for some less obvious but very useful purposes, such as to model articulated deformable characters or to drive a keyframe animation such that the animator controls the movement of only a few constrained points then the technique generates the movement of all the nodal points. We expect the deformation technique proposed here will prove useful in many application areas, including computer games and character animation.

A Computation of Equation (19)

To compute $\tilde{\mathbf{R}}^k$ in Equation (19), we first convert the rotation vector $\mathbf{w}_i(t)$ of each node into the rotation matrix $\mathscr{R}(\mathbf{w}_i(t))$. For this conversion we employ the Rodriguez formula [15] that expresses the rotation matrix in terms of the angle and the unit axis of rotation. Let $\mathbf{w}_i^k$ be the i-th three-dimensional vector of $\Psi\mathbf{q}^k$. Then, $\mathbf{w}_i(t) = \tau\mathbf{w}_i^k$, where $\tau = t/t^k$. The Rodriguez formula gives

$$\mathscr{R}(\tau\mathbf{w}_i^k) = \mathbf{I} + (\hat{\mathbf{w}}_i^k\times)\sin\|\tau\mathbf{w}_i^k\| + (\hat{\mathbf{w}}_i^k\times)^2(1-\cos\|\tau\mathbf{w}_i^k\|),$$

where $\hat{\mathbf{w}}_i^k = \mathbf{w}_i^k/\|\mathbf{w}_i^k\|$. Now, we integrate both sides of this equation from $\tau = 0$ to 1, Then, $\tilde{\mathbf{R}}_i^k \triangleq \int_0^1 \mathscr{R}(\tau\mathbf{w}_i^k)d\tau$ is given by

$$\tilde{\mathbf{R}}_i^k = \left[\mathbf{I} + (\hat{\mathbf{w}}_i^k\times)\tfrac{1-\cos\|\mathbf{w}_i^k\|}{\|\mathbf{w}_i^k\|} + (\hat{\mathbf{w}}_i^k\times)^2\left(1-\tfrac{\sin\|\mathbf{w}_i^k\|}{\|\mathbf{w}_i^k\|}\right)\right].$$

Finally, the composite block-diagonal rotation matrix for $\mathbf{w}^k = [\mathbf{w}_i^k]$ can be constructed by $\tilde{\mathbf{R}}^k = [\delta_{ij}\tilde{\mathbf{R}}_i^k]$.

References

[1] S. Capell, S. Green, B. Curless, T. Duchamp, and Z. Popović. Interactive skeleton-driven dynamic deformations. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2002)*, 21(3):586–593, 2002.

[2] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. Dynamic real-time deformations using space and time adaptive sampling. *Computer Graphics (Proc. ACM SIGGRAPH 2001)*, 35:31–36, 2001.

[3] O. Etzmuss, M. Keckeisen, and W. Straβer. A fast finite element solution for cloth modeling. In *Proc. Pacific Graphics 2003*, pages 244–251, 2003.

[4] S. Gibson and B. Mirtich. A survey of deformable modeling in computer graphics. Technical Report TR-97-19, Mitsubishi Electric Research Lab., Cambridge, MA, 1997.

[5] E. Grinspun, P. Krysl, and P. Schröder. CHARMS: A simple framework for adaptive simulation. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2002)*, 21(3):281–290, 2002.

[6] K. K. Hauser, C. Shen, and J. F. O'Brien. Interactive deformation using modal analysis with constraints. In *Proc. Graphics Interface 2003*, pages 247–255, 2003.

[7] D. L. James and K. Fatahalian. Precomputing interactive dynamic deformable scenes. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2003)*, 22(3):879–887, 2003.

[8] D. L. James and D. K. Pai. DyRT: Dynamic response textures for real time deformation simulation with graphics hardware. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2002)*, 21(3):582–585, 2002.

[9] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia, PA, 1998.

[10] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler. Stable real-time deformations. In *Proc. ACM SIGGRAPH Symp. Computer Animation 2002*, pages 49–54, 2002.

[11] J. F. O'Brien and J. K. Hodgins. Graphical modeling and animation of brittle fracture. *Computer Graphics (Proc. ACM SIGGRAPH '99)*, 33(4):137–146, 1999.

[12] J. F. O'Brien, C. Shen, and C. M. Gatchalian. Synthesizing sounds from rigid-body simulations. pages 175–182, 2002.

[13] A. Pentland and J. Williams. Good vibrations: Model dynamics for graphics and animation. *Computer Graphics (Proc. ACM SIGGRAPH '89)*, 23(3):207–214, 1989.

[14] A. A. Shabana. *Theory of Vibration, Volume II: Discrete and Continuous Systems*. Springer-Verlag, New York, NY, 1990.

[15] A. A. Shabana. *Dynamics of Multibody Systems*. Cambridge University Press, 1998.

[16] J. Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *Computer Graphics Forum (Proc. EUROGRAPHICS '97)*, 16(3):159–164, 1997.

[17] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. *Computer Graphics (Proc. ACM SIGGRAPH '88)*, 22(4):269–278, 1988.

[18] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. *Computer Graphics (Proc. ACM SIGGRAPH '87)*, 21(4):205–214, 1987.

[19] D. Terzopoulos and A. Witkin. Physically based models with rigid and deformable components. *IEEE Computer Graphics & Applications*, 8(6):41–51, 1988.

[20] O. C. Zienkiewicz. *The Finite Element Method*. McGraw-Hill Book Company (UK) Limited, Maidenhead, Berkshire, England, 1977.