Animating Deformable Objects using Sparse Spacetime Constraints

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

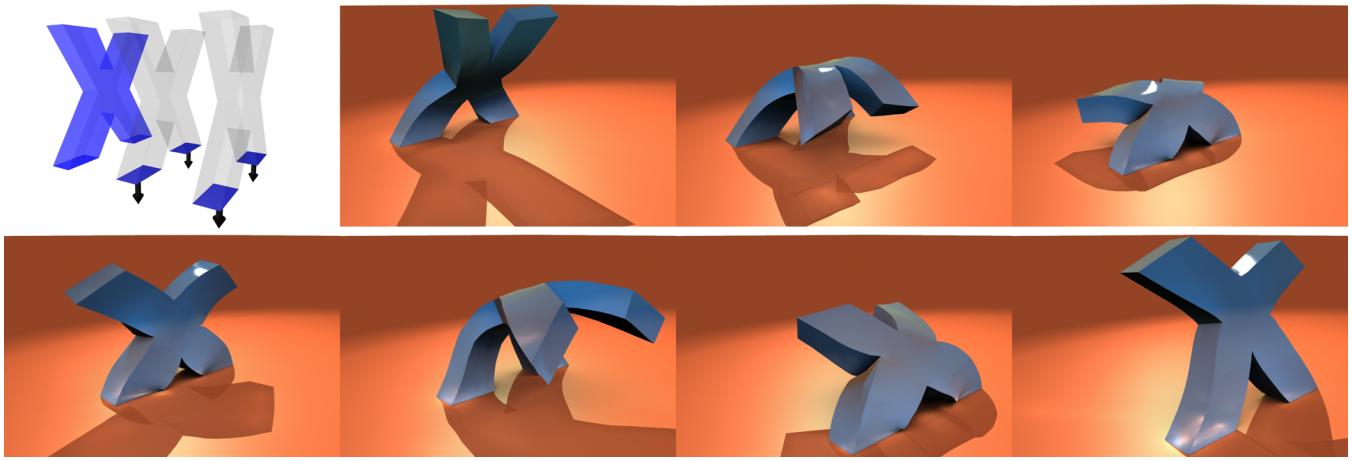

Figure 1: *Animation of a "letter X"soft body character performing a handspring. The animation is automatically generated from a sparse set of partial keyframes, which is illustrated in the image on the upper left.*

Abstract

We propose a scheme for animating deformable objects based on spacetime optimization. The main feature is that it robustly and within a few seconds generates interesting motion from a sparse set of spacetime constraints. Providing only partial (as opposed to full) keyframes for positions and velocities is sufficient. The computed motion satisfies the constraints and the remaining degrees of freedom are determined by physical principles using elasticity and the spacetime constraints paradigm. Our modeling of the spacetime optimization problem combines dimensional reduction, modal coordinates, wiggly splines, and rotation strain warping. Our solver is based on a theorem that characterizes the solutions of the optimization problem and allows us to restrict the optimization to low-dimensional search spaces. This treatment of the optimization problem avoids a time discretization and the resulting method can robustly deal with sparse input and wiggly motion.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling;

Keywords: spacetime constraints, physically-based animation, model reduction, optimal control, wiggly splines

Links: DL PDF

1 Introduction

Directing and controlling physical systems is a challenging task in computer animation. We look at the problem of creating realistic looking motion of soft body characters or deformable objects within a scene. Modeling such motion using traditional computer animation techniques, like spline fitting, is difficult since many degrees of freedom must be determined and secondary motion effects must be modeled by hand. Physical simulation can be of great help for creating realistic and detailed animations, but one needs to determine the forces and physical parameters that produce an effect an animator wants to produce. The *spacetime constraints* paradigm combines the realism provided by physical simulation with the control over an animation offered by keyframing. The idea is to compute the motion as the solution of a constrained spacetime optimization problem in which the keyframes serve as constraints in spacetime. The result is a planned motion showing desired effects like squash-and-stretch, timing, and anticipation. Spacetime optimization has been used for animating articulated characters (which are controlled by a skeleton) since the late eighties. More recently, this approach has been extended to soft body characters and deformable objects.

In this paper, we present a method that robustly generates the motion of deformable objects from sparse input. For example, instead of providing a set of full keyframes, it suffices to specify spacetime constraints (position, velocities) for parts of the object at a sparse set of points in time. This permits the generation of interesting and realistic looking motion with rich secondary motion from a rough sketch. Our goal is to reduce the information an animator must provide to generate a motion and thereby simplify the generation of motion. Creating full poses and velocities that lead to a natural looking motion can be difficult. For example, let us look at one keyframe of the forward jump of X shown in Figure 3. At the point in time when the character leaves the ground, we specify a partial keyframe that prescribes only the average velocity of X. If we want to create such a motion using full keyframes, we would need to replace the partial keyframe by a full keyframe that specifies the positions and the velocities at this point in time. But what are the

pose and velocity at this point in time? When generating a motion, our system automatically generates full keyframes from the partial input. Interpolating these full keyframes produces the same motion. In this sense the system can be used as a modeling tool for poses and velocites that takes into account not only the geometry and material of a static shape but also the dynamics of the motion.

From a technical point of view, solving a spacetime optimization problem with sparse constraints is challenging. Recent methods, like [Huang et al. 2011] and [Hildebrandt et al. 2012], cannot deal with partial keyframes because they perform computations that require full keyframes. The scheme presented by Barbič et al. [2012] is the first method that addresses the problem of sparse constraints for spacetime optimization of deformable objects. The method is designed for editing animations in a physically plausible way, but, in principle, it can also be used for generating motion (by using a static animation as input motion). However, for the problem considered here, there are fundamental limitations. Only a very special type of sparse constraints (namely constraining the position of a few vertices) is considered, and, more importantly, the control over the positions is only strict for small deformations. Animations that include larger deformations are warped in a post-process to remove linearization artifacts, which implies that the constraints are no longer enforced.

Our method relies on two main contributions. The first is the mathematical modeling of the spacetime optimization problem with partial keyframes for the positions and velocities. Following Barbič et al. [2012], we consider linearized elasticity and rotation strain warping. In contrast to their approach, we impose the constraints on the warped motion and its velocity. One technical difficulty in formulating the optimization problem is that we need the derivative of the rotation strain warp map, which has not been used before. We derive a scheme for the efficient and robust computation of the derivative. The second contribution is that we develop an efficient solver for the resulting optimization problem. One important step is that we prove (in Theorems 1 and 2) that the minimizers of the optimization problem must be of a certain form. As a consequence, we can restrict the optimization problem from the infinite-dimensional space of all (regular enough) motions to a finite-dimensional space of all motions that are of the specified form. By combining this result with a dimensional reduction of the finite-dimensional space, we construct low-dimensional spaces ($<$100 dim.) to which we restrict the optimization problem. We demonstrate that the resulting low-dimensional nonlinear least-squares problems can be solved within a few seconds with standard solvers. This treatment of the optimization problem avoids a time discretization and the resulting method can robustly deal with sparse input and wiggly motion. The explicit form of the minimizers is based on wiggly splines [Kass and Anderson 2008] that have been used for physical-based interpolation of *full* keyframes in [Huang et al. 2011] and [Hildebrandt et al. 2012]. An important difference between full and partial keyframes is that the full keyframes decouple in the modal basis, whereas the partial keyframes couple all the modes. Therefore, interpolation of full keyframes is a simpler problem. In the modal basis, the linearized spacetime optimization problem with full keyframes decouples and only one-dimensional problems need to be solved. The solutions of the one-dimensional problems are explicitly given by the wiggly splines. Hence no optimization is necessary. Such a treatment is not possible for more general constraints that couple the different modes, like partial keyframes. Therefore, to be able to use the wiggly splines for spacetime optimization with sparse constraints, a new modeling of the problem (as presented in this work) is necessary.

2 Related Work

Simulation of deformable objects is widely used in graphics. An introduction to the topic can be found in the recent tutorial [Sifakis and Barbič 2012]. Among the various techniques that have been proposed for accelerating the computation of deformable object simulations, two are most important for our work. One is *dimension reduction*. This is an established technique in mechanics [Nickell 1976; Krysl et al. 2001] and graphics [Pentland and Williams 1989; Barbič and James 2005; An et al. 2008; Kim and James 2009; von Tycowicz et al. 2013]. The principle is to construct a low-dimensional subspace to which the simulation is restricted. Since a motion typically is of low rank, a good approximation of the simulation can already be obtained in low-dimensional subspaces. Subspace constructions are based on local analysis, *e.g.* vibration modes and modal derivatives, or sampling and proper orthogonal decomposition. The second technique is *rotation strain warping* [Huang et al. 2011]. Like co-rotation methods [Müller et al. 2002] and modal warping [Choi and Ko 2005], the goal is to improve linearized elasticity while keeping the computation cost low. Rotation strain warping aims at removing visible linearization artifacts from a deformation. For this the matrix exponential is used to map the antisymmetric part of the deformation gradient of every tet to a rotation matrix. Then, the rotation matrices are used to construct target deformation gradients for all tets. The warped deformation is a deformation whose gradients best match the target gradients.

Animating characters or objects often requires not only simulation but also control over the simulation. The idea of using constrained spacetime optimization for animation was introduced by Witkin and Kass [1988] in the late eighties. Their examples demonstrate that the resulting animations naturally include motion effects that are desired in character animation. The so-called *spacetime constraints paradigm* has inspired many researchers. The focus has been on articulated characters and human motion, see [Cohen 1992; Gleicher 1997; Fang and Pollard 2003; Safonova et al. 2004] and references therein. More recently, constrained spacetime optimization has been used to control other physical systems including rigid body motions [Popović et al. 2003], fluids [Treuille et al. 2003; McNamara et al. 2004], particle systems [Wojtan et al. 2006], and deformable objects [Barbič et al. 2009; Hildebrandt et al. 2012]. The resulting optimization problems are solved with gradient-based or Newton methods. Automatic [Safonova et al. 2004] and symbolic [Witkin and Kass 1988] differentiation and the adjoint method [McNamara et al. 2004] have been used to compute the derivatives of the objective functional. Local-to-global strategies [Cohen 1992; Treuille et al. 2003] and dimension reduction [Safonova et al. 2004; Sulejmanpašić and Popović 2005; Barbič et al. 2009] have been applied to speed up the optimization. In [Kass and Anderson 2008], tools for motion design based on spacetime constraints for one-dimensional mass-spring systems were presented. The variational characterization of the solution of the one-dimensional optimization problem is analogous to that of cubic splines. Since these splines are oscillatory instead of smooth, they are called *wiggly splines*. An explicit form of the wiggly splines was derived in [Hildebrandt et al. 2012], which yield fast and robust computations of the wiggly splines. In addition, they showed that for linearized elastic objects, every modal coordinate of the solution of the constrained spacetime optimization problem is a wiggly spline. Based on this and a multipoint linearization, a method for spacetime control of deformable objects with full keyframes was presented. In [Huang et al. 2011] the rotation strain warping was used for generating motions of deformable objects that interpolate (full) keyframes. Barbič et al. [2012] introduced a scheme for physically-based editing of animations using spacetime optimization. Another editing tool for animations has recently

been introduced by Li et al. [2013].

An alternative approach for controlling deformable objects is the use of *proportional-derivative controllers*, which were introduced to computer graphics by Kondo et al. [2005]. Another alternative is locomotion control twhich was recently used for animating soft body characters with a skeleton [Kim and Pollard 2011] and without a skeleton [Coros et al. 2012; Tan et al. 2012]. In contrast to spacetime optimization, the control forces in all of these approaches are computed without a longer planning horizon. Tracking solvers [Bergou et al. 2007] take a coarse animation as input and augment it with physical details by running a fine simulation that is forced to remain close to the coarse input. *Example-based materials* [Martin et al. 2011] direct a simulation towards a desired behavior by providing a set of example poses. During the simulation, a force pushes the deformable object towards a submanifold of the shape space, which is generated from the example shapes.

Related to spacetime optimization is the boundary value problem for the equations of motion: For two poses and two points in time, compute the trajectory that connects them. In contrast to the spacetime optimization we consider, no additional force is allowed. Compared to the initial value problem, the numerical treatment of the boundary value problem received less attention. We refer to [Huang et al. 2011] for a recent method and an overview.

Another related problem is *geometric shape interpolation*. Approaches that define a Riemannian metric on the space of all possible shapes and then compute geodesics to interpolate between shapes have been proposed [Kilian et al. 2007; Heeren et al. 2012]. Such a Riemannian metric on a shape space measures the metric distortion or the viscous dissipation required to physically deform an elastic object. Simpler approaches compute interpolating shapes by solving large but sparse linear systems, *e.g.* [Sumner et al. 2005; Xu et al. 2005]. In contrast to spacetime constraints, geometric shape interpolation does not include the dynamics, *e.g.* does not exhibit secondary motion effects.

3 Constrained Spacetime Optimization

The generation of motion based on the *spacetime constraints* paradigm has been studied in graphics for a long time. Various constrained spacetime optimization problems have been formulated and tested since then. In this section, we give a brief overview of the spacetime optimization considered in this work.

We represent deformable objects by a tetrahedral mesh and use a finite elements discretization of elastic solids. Any deformation of the object is described by a displacement vector $u \in \mathbb{R}^n$, where n is the number of the object's degrees of freedom. The dynamics of the object is governed by the equations of motion

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

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

Our goal is to construct a trajectory $u(t)$ that satisfies a set of spacetime constraints, *e.g.* interpolating positions or velocities at the nodes $\{t_0, t_1, \ldots, t_m\}$. Since a solution of (1) is determined by an initial position and velocity at some point in time, it cannot in general satisfy the constraints. To allow the object to modify its motion, we add an extra force $F_{add}(t)$ to the system (1)

$$M\,\ddot{u}(t) = F(u, \dot{u}, t) + F_{add}(t).$$

This force can be regarded as an additional interior force generated by the deformable object or as an additional exterior force. Among all possible forces $F_{add}(t)$, which result in a motion that satisfies the constraints, we choose the force with minimal (squared) spacetime L^2-norm

$$\frac{1}{2}\int_{t_0}^{t_m} \|F_{add}\|^2_{M^{-1}}\ \mathrm{d}t = \frac{1}{2}\int_{t_0}^{t_m} \|M\,\ddot{u} - F\|^2_{M^{-1}}\ \mathrm{d}t. \tag{2}$$

Here, we denote $\|F(t)\|^2_{M^{-1}} = F^T(t)M^{-1}F(t)$. The reason, we use the M^{-1}-norm here, is that F is an integrated quantity and $M^{-1}F$ is the pointwise force at the vertices. The term $\|F(t)\|^2_{M^{-1}} = F^T(t)M^{-1}F(t) = \|M^{-1}F(t)\|^2_M$ is the squared L^2-norm of the force at time t. The concept of penalizing the squared norm of the forces goes back to [Witkin and Kass 1988]. For deformable objects this energy was used by Barbič et al. [2009] and the M^{-1} notation was used in [Barbič et al. 2012].

The resulting motion is a planned motion and the required force is distributed over the whole animation. If we think of F_{add} as a force generated by a soft body character, the character tries to use its force efficiently planning its motion ahead. If F_{add} is an external force that is used to manipulate the system to achieve an effect, the required force is distributed over the whole animation so that the manipulation is less visible.

4 Linearized Spacetime Optimization

Our method is based on properties of the spacetime optimization problem for linear dynamics, which enables a robust and efficient computation of the solution. We briefly summarize these properties in this section. We show that the solutions of the spacetime optimization problem for linear dynamics can be explicitly formulated in terms of the solutions of one-dimensional problems. We look at the one-dimensional problems first.

4.1 One-dimensional linear problems: Wiggly splines

Let us consider the following linear equation of motion for a point in a one-dimensional space

$$\ddot{\omega}(t) + 2\,\delta\dot{\omega}(t) + \lambda\,\omega(t) + g = 0,$$

where $\omega(t)$ is the position of the point, $\delta \in \mathbb{R}_{\geq 0}$ a damping parameter, $\lambda \in \mathbb{R}$ a spring constant and $g \in \mathbb{R}$ a constant. For this system, the functional (2) takes the form

$$\omega \mapsto \frac{1}{2}\int_{t_0}^{t_m} \left(\ddot{\omega}(t) + 2\,\delta\dot{\omega}(t) + \lambda\,\omega(t) + g\right)^2\ \mathrm{d}t. \tag{3}$$

This is a quadratic functional on the Sobolev space $H^2((t_0, t_m), \mathbb{R})$. The corresponding Euler-Lagrange equation is

$$\dddot{\omega}(t) + 2(\lambda - 2\delta^2)\,\ddot{\omega}(t) + \lambda^2\,\omega(t) + \lambda\,g = 0. \tag{4}$$

The solution space of this linear fourth-order ODE is described in [Hildebrandt et al. 2012]. In the generic case, where $\delta, \lambda, \delta^2 - \lambda \neq 0$, it is spanned by the four functions

$$b^{1,2,3,4}(t) = \mathrm{Re}\left(e^{\left(\pm\delta \pm \sqrt{\delta^2 - \lambda}\right)t}\right). \tag{5}$$

If we have $m + 1$ nodes $\{t_0, t_1, \ldots, t_m\}$ at which constraints are specified, then the minimizer of (3) is a linear combination of the four basis functions (plus a constant) within each interval

(t_k, t_{k+1}). It is twice differentiable at nodes where only the position (and not the velocity) is prescribed and once differentiable at all other nodes. Explicitly, the restriction $\omega_k(t)$ of the solution $\omega(t)$ to the interval $[t_{k-1}, t_k]$ has the form

$$\omega(t)_{|[t_{k-1},t_k]} = \omega_k(t) = \sum_{l=1}^{4} w_k^l b^l(t) - c,$$

where w_k^1, w_k^2, w_k^3, and w_k^4 are coefficients and $c = g/|\lambda|$ if $\lambda \neq 0$ and $c = 0$ if $\lambda = 0$. The construction of the function $\omega(t)$ is analogous to that of a cubic spline and is therefore called a *wiggly spline*, see [Kass and Anderson 2008]. The property that ω is once or twice differentiable at the nodes translates to linear conditions on the coefficients w_k^l :

$$\begin{aligned} \omega_k(t_k) &= \omega_{k+1}(t_k) \\ \dot{\omega}_k(t_k) &= \dot{\omega}_{k+1}(t_k) \\ \ddot{\omega}_k(t_k) &= \ddot{\omega}_{k+1}(t_k) \end{aligned} \tag{6}$$

for all $k \in \{1, 2, ..., m-1\}$, where the last equation holds only if ω is twice differentiable at t_k. The interpolation and boundary conditions specify additional linear conditions that uniquely determine a wiggly spline.

4.2 Spacetime optimization for linear dynamics

After a finite elements discretization, the linear dynamics of a deformable object is described by the equations

$$M\,\ddot{u} + (\alpha\,M + \beta\,K)\dot{u} + K\,u + g = 0, \tag{7}$$

where K is the stiffness matrix, $\alpha\,M + \beta\,K$ a Rayleigh damping term, and g a constant vector. Then, the functional (2) takes the form

$$E(u) = \frac{1}{2}\int_{t_0}^{t_m} \|M\,\ddot{u} + (\alpha\,M + \beta\,K)\dot{u} + K\,u + g\|^2_{M^{-1}}\ \mathrm{d}t. \tag{8}$$

This energy was used for spacetime optimization in [Barbič et al. 2012]. The eigenvalues and eigenmodes of (7) are solutions of the equation

$$K\,\phi_i = \lambda_i M\,\phi_i.$$

We consider a basis $\{\phi_1, \phi_2, ..., \phi_n\}$ of $\mathbb{R}^n$ consisting of eigenmodes. In such a basis the system (7) decouples and the functional (8) can be written as a sum over functionals of the form (3) that depend on only one variable. Therefore, the solutions of the Euler-Lagrange equation of (8) can be written as

$$u(t) = \sum_i \omega_i(t)\phi_i, \tag{9}$$

where any $\omega_i(t)$ is a wiggly spline. For any i, the parameters λ and δ of the wiggly spline $\omega_i(t)$ depend on λ_i and the damping coefficients α and β. Explicitly, $\lambda = \lambda_i$ and $2\delta = \alpha + \beta\lambda_i$. This explicit form of the solutions was introduced in [Hildebrandt et al. 2012].

5 Sparse Constraints for Linear Dynamics

In this section, we describe a fast solver for spacetime optimization problems with sparse constraints for linear dynamics. In the following sections, we will extend this approach to warped linear dynamics. By spacetime constraints, we mean a set of linear equality constraints on the position $u(t_k)$ and the velocity $\dot{u}(t_k)$ of the object at a set of $m+1$ nodes t_k in the time interval $[t_0, t_m]$. Here, we are interested in sparse constraints, which means that the number of constraints at each node t_k is small compared to the number n of the object's degrees of freedom. For example, we want to prescribe only partial (as opposed to full) keyframes.

In the optimization, we treat the constraints as soft constraints modeled by a least-squares energy. There are two main reasons for doing so. One is that, for efficiency, we restrict the optimization to a low-dimensional subspace (as we will discuss below). Within this subspace, it is usually not possible to satisfy the constraints exactly because the subspace dimension is smaller than the number of constraints. The justification to our approach is that with increasing least-squares weights and subspace dimension, the soft constraints converge towards equality constraints. The second reason is that in many cases, we prefer soft over hard constraints. Soft constraints give the method the freedom to adjust the shape of the constraints. For the user, this has the advantage that roughly modeled input can be used. For example, we did not need a surface modeler for any of our examples; we simply selected parts of the geometry and translated and rotated them in space. Similarly, the velocities we specified are constant vector fields at selected vertices.

The constraints are represented by the energy E_C

$$E_C(u) = \frac{1}{2}\sum_{k=0}^{m} \left(c_A \left\|A_k u(t_k) - a_k\right\|^2 + c_B \left\|B_k \dot{u}(t_k) - b_k\right\|^2\right),$$

where A_k, B_k are rectangular matrices, a_k, b_k are vectors, and c_A, c_B are constants. In our examples, we set only constraints on the positions and velocities of individual vertices. Then, the matrices A_k and B_k have only one non-zero entry per row. This entry has the value one and the corresponding entry in the vector specifies the position or velocity of one coordinate of one vertex. To make the energy robust to remeshing and refinement, we multiply each row and entry in the vector by the square root of the mass of the vertex that is constrained. By the mass of a vertex we mean a quarter of the combined volume of all adjacent tets. For example, if we constrain the position of all vertices of the base of the Buddha model, weighing them with the masses ensures that the strength of the constraint energy depends on the volume of the base and the constant c_A and not on the discretization of the model.

To get a motion that follows the sparse keyframes, we need to minimize the energy

$$\mathcal{E}(u) = E(u) + E_C(u), \tag{10}$$

where $E(u)$ is the functional defined in (8). The space over which we minimize is the Sobolev space $H^2((t_0, t_m), \mathbb{R}^n)$ of all motions whose derivatives up to second order are square integrable—an infinite-dimensional space. The following theorem shows that the minimizer can be constructed using wiggly splines. The theorem greatly reduces the search space and is the basis of the presented approach. A proof of the theorem can be found in [Schulz et al. 2014].

Theorem 1 *The minimizers of the energy $E(u)+E_C(u)$ among all functions in the Sobolev space $H^2((t_0, t_m), \mathbb{R}^n)$ are of the form (9) and are twice differentiable at any node t_k where no velocity is prescribed and once differentiable at all other nodes.*

The theorem implies that instead of minimizing over all possible motions, we only need to determine the coefficients of the wiggly splines ω_i in (9). Thus the infinite-dimensional optimization problem is replaced by a finite-dimensional problem. Since (9) involves all n eigenfunctions ϕ_i, this still requires complex computation. Therefore, we apply dimension reduction to the problem. This is a common technique in graphics. For a recent tutorial on the subject,

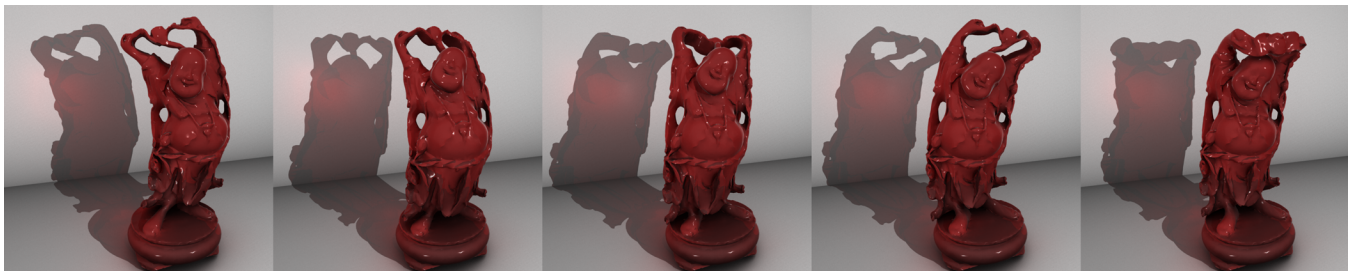

Figure 2: *Snapshots from a dancing Buddha animation. The motion exhibits interesting secondary motion effects that are automatically generated. Three partial keyframes that prescribe the positions of a few vertices on the belly of the model were used to generate the motion.*

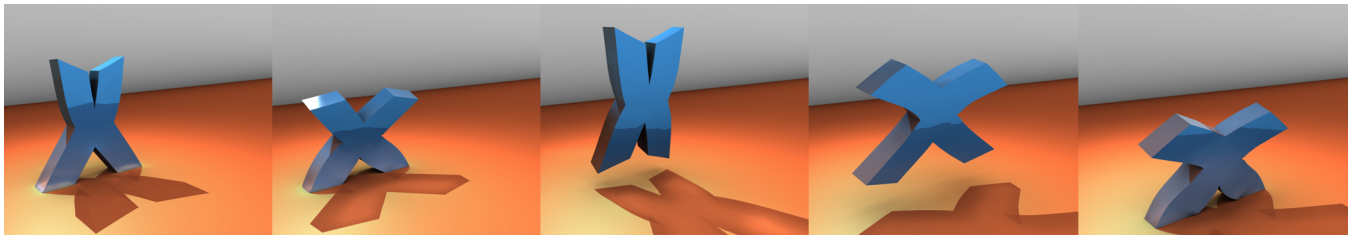

Figure 3: *Jumping X animation generated from sparse input. The example illustrates the anticipation effect.*

we refer to [Sifakis and Barbič 2012]. The motivation is that animations of deformable objects are typically of low rank and can be well-approximated in a low-dimensional subspace of $\mathbb{R}^n$. In our examples, we used the space spanned by the 10-50 lowest-frequency eigenmodes and restricted the optimization problem (and hence the resulting motion) to this space. This can be implemented by restricting the sum in (9) to a sum over only the first d modes ϕ_i instead of all n modes.

The space of all functions of the form (9) is a linear space and can be described by the coefficients of the wiggly splines, see Section 4.1. If constraints at $m+1$ nodes t_k are specified, we have m intervals $[t_k, t_{k+1}]$. Since a wiggly spline ω_i has four coefficients $w^l_{i,k}$ (where $k \in \{1,2,...,m\}$ is the index of the segment and $l \in \{1,2,3,4\}$) for every segment and we restrict to d eigenmodes, $4md$ coefficients have to be determined. Since the minimizer of (10) is once or twice differentiable at the nodes t_k (depending on whether or not velocities are prescribed), any wiggly spline ω_i in (9) must satisfy the continuity conditions listed in (6). Depending on whether velocities are prescribed at the keyframes, these are between $2(m-1)d$ and $3(m-1)d$ linear conditions. In addition, we specified (full) positions and velocities at the boundaries, t_0 and t_m, in most of the examples. These are again linear conditions, which determine up to $4d$ coefficients. A basis of the remaining space of admissible motions can be computed using a singular value decomposition of the matrix representing the continuity and boundary conditions. The resulting spaces are low-dimensional, less than 100-dimensional in all our examples. Using the singular value decomposition to eliminate the linear constraints is not necessary for the linear system discussed in this section, but it is effective for the nonlinear system, we introduce in Section 7.

Since E and E_C are quadratic energies, they can be represented as matrices. To set up the matrix representation of E, integrals over products of the basis functions and their first and second derivatives must be computed. Explicit formulas for these integrals can be computed using a computer algebra system (or a table of integrals). Once the matrices are computed, a motion can be determined by solving a low-dimensional linear system.

6 Rotation strain warping and its derivative

For small deformations, linear elasticity is a good approximation of its nonlinear counterpart. Various approaches that are intended to be an improvement on linear elasticity have been introduced. Here, we consider the *rotation strain warping*, which was introduced in [Huang et al. 2011] (see also [Barbič et al. 2012; Sifakis and Barbič 2012]). The warping map removes artifacts caused by linearized elasticity and worked very well in our experiments. Without using warping, all the examples presented in the accompanying video would develop visible artifacts; examples like the handspring would be completely distorted. To integrate rotation strain warping to the spacetime optimization in the next section, we need the derivative of the rotation strain warp map. Since no scheme for computing the warping derivatives has been introduced before, we derive a formula here. We begin with a brief review of rotation strain warping.

The displacement u of the vertices can be extended linearly into the tets and we get a piecewise linear and continuous function (or linear Lagrange finite element) from the rest shape into $\mathbb{R}^3$. For each tet T_j, the derivative of this map (at any point in T_j) is a linear map of $\mathbb{R}^3$. The corresponding map $\nabla_j : \mathbb{R}^n \mapsto \mathbb{R}^{3\times 3}$ (which maps a displacement vector to the derivative in T_j) is linear. By adding the displacement to the coordinates of the vertices of the rest pose, we get the coordinates of the deformed mesh. The corresponding map from the rest shape to the deformed mesh is piecewise affine and continuous. The derivative of this map in any tet T_j is called the deformation gradient G_j and we have $G_j(u) = Id + \nabla_j u$, where $Id \in \mathbb{R}^{3\times 3}$ is the identity matrix. The deformation gradient can be split into a symmetric and an antisymmetric part

$$G_j = G_j^s + G_j^a = \frac{1}{2}\left(G_j + G_j^T\right) + \frac{1}{2}\left(G_j - G_j^T\right).$$

The antisymmetric part can be seen as a linearizations of a rotation of the tet since the antisymmetric 3×3-matrices form the Lie algebra of the Lie group of rotation matrices of $\mathbb{R}^3$. The matrix exponential maps any antisymmetric 3×3-matrix to the corresponding rotation matrix. The rotation strain coordinates use the tensor

$$\tilde{G}_j = \exp(G_j^a) G_j^s.$$

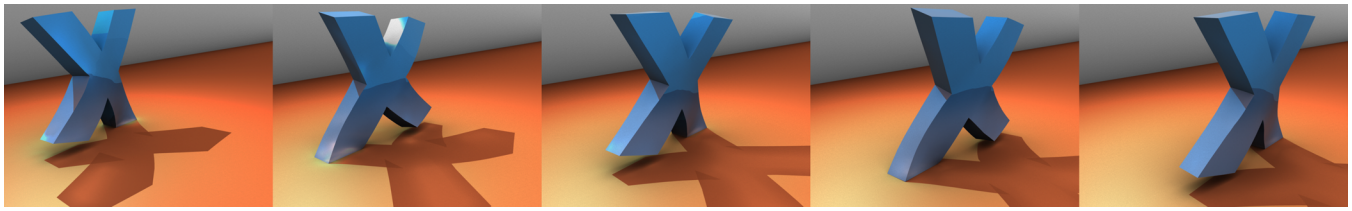

Figure 4: *Walking X animation generated from sparse input. Only a sparse set of constraints on the positions and the velocities of the feet are prescribed.*

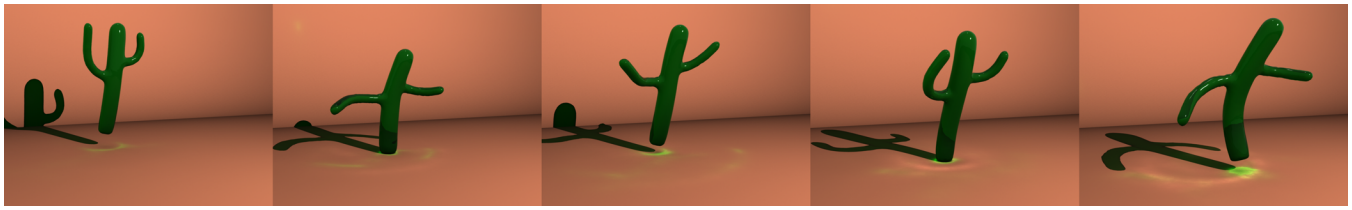

Figure 5: *Snapshots from an animation of a forward jumping cactus, performing three consecutive jumps. The motion is controlled through constraints on the position of the "foot" and on the position and velocity of the center of mass. The cactus exhibits natural secondary motion and crouches to prepare the jumps and to absorb the impacts of the landings.*

For any displacement vector u, one can compute the tensor $\tilde{G}_j(u)$ for every T_j. In general, there is no displacement $\tilde{u}$ whose deformation gradient $G_j(\tilde{u})$ equals $\tilde{G}_j(u)$ for every tet. But we can compute a deformation φ whose deformation gradients $G_j(\varphi)$ best match the $\tilde{G}_j(u)$ in a least-squares sense, that is,

$$\min_{\varphi} \sum_{j=1}^{\tau} |T_j| \left\| G_j(\varphi) - \tilde{G}_j(u) \right\|^2, \tag{11}$$

where $|T_j|$ denotes the volume of T_j, τ is the number of tets, and the norm is the Frobenius norm. The warp map $W : \mathbb{R}^n \mapsto \mathbb{R}^n$ maps the displacement u to the minimizer φ. One motivation for considering $\tilde{G}_j$ is that ${G_j^s}^T G_j^s = \tilde{G}_j^T \tilde{G}_j$, which means that the desired deformation tensor induces the same metric strain as the symmetric part of the tensor G_j.

Using the linear maps ∇_j, the least-squares problem (11) can be written as

$$\min_{\varphi} \sum_{j=1}^{\tau} |T_j| \left\| \nabla_j \varphi - \tilde{G}_j(u) + Id \right\|^2.$$

We consider two maps $\Gamma, \widetilde{\Gamma} : \mathbb{R}^n \mapsto \mathbb{R}^{9\tau}$. Γ associates to any deformation $\varphi \in \mathbb{R}^n$ the $\mathbb{R}^{9\tau}$-vector that lists the 9 coordinates of the τ matrices $\nabla_j \varphi$ times the square root of $|T_j|$. Similarly, $\widetilde{\Gamma}$ maps any deformation $u \in \mathbb{R}^n$ to the vector that lists coordinates of the matrices $|T_j|^{\frac{1}{2}} (\tilde{G}_j(u) - Id)$. While Γ is a linear map, $\widetilde{\Gamma}$ is nonlinear since the definition of $\tilde{G}_j$ involves the matrix exponential. Both maps are sparse in the sense that $\nabla_j \varphi$ and $\widetilde{G}_j(u)$ are determined by the deformation of the four vertices of the jth tet. For a fixed u, the warped deformation $W(u)$ is determined by the linear least-squares problem

$$W(u) = \arg\min_{\varphi} \left\| \Gamma \varphi - \widetilde{\Gamma}(u) \right\|^2,$$

where the norm is the standard norm in $\mathbb{R}^{9\tau}$. The corresponding normal equation is

$$\Gamma^T \Gamma W(u) = \Gamma^T \widetilde{\Gamma}(u). \tag{12}$$

For an unconstrained object, the matrix $\Gamma^T \Gamma$ is the usual stiffness (or Laplace) matrix (of the linear Lagrange finite elements) of the rest shape. In most examples, we constrain the positions of some vertices, *e.g.* the base of the blocks. Then, the matrix $\Gamma^T \Gamma$ is the stiffness matrix without the rows and columns corresponding to the constrained vertex coordinates. Then, $\Gamma^T \Gamma$ has full rank. In the case of an unconstrained object, $\Gamma^T \Gamma$ has the translations in its kernel. Then, $W(u)$ is only determined up to translations. To get a unique solution in this case, we fix the center of mass.

Our aim is to compute the derivative $\mathrm{D}W$ of the warp map with respect to variations of u. Equation (12) shows that $W(u)$ depends linearly on $\widetilde{\Gamma}(u)$. To verify that the same holds for the derivatives of $W(u)$ and $\widetilde{\Gamma}(u)$, we differentiate both sides of the equation and use the fact that Γ is linear

$$\Gamma^T \Gamma \mathrm{D}W = \Gamma^T \mathrm{D}\widetilde{\Gamma}.$$

The matrix $\Gamma^T \Gamma$ is sparse and symmetric and independent of u, it depends only on the rest configuration. Therefore, using a sparse factorization of $\Gamma^T \Gamma$, $\mathrm{D}W$ can be efficiently computed once $\mathrm{D}\widetilde{\Gamma}$ is known.

To get $\mathrm{D}\widetilde{\Gamma}$, we need the derivatives of the $\widetilde{G}_j$s. Using the chain and product rule, the identities $G_j^a(u) = \nabla_j^a u$ and $G_j^s(u) = Id + \nabla_j^s u$, and the fact that $\nabla_j^a u$ and $\nabla_j^s u$ depend linearly on u; we get a representation of the derivative of $\widetilde{\Gamma}$ at u in the direction of v in terms of ∇_j^a, ∇_j^s, the matrix exponential and its derivative

$$\begin{aligned} \mathrm{D}_v \widetilde{G}_j(u) &= \mathrm{D}_v \left(\exp\left(G_j^a(u)\right) G_j^s(u) \right) \\ &= \mathrm{D}_v \exp(\nabla_j^a u)(Id + \nabla_j^s u) + \exp(\nabla_j^a u) \nabla_j^s v. \end{aligned}$$

The matrix exponential (on the space of antisymmetric matrices) and its derivative can be efficiently evaluated using Rodrigues' rotation formula. A discussion is contained in the appendix.

7 Sparse Constraints and Warping

In this section, we integrate rotation strain warping to the framework for spacetime optimization under sparse constraints, which

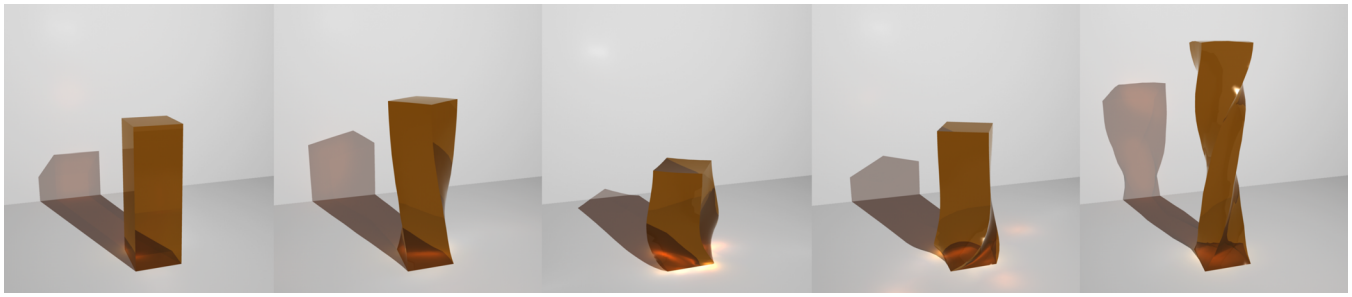

Figure 6: *Snapshots from an animation of a deformable block that spirals upwards.*

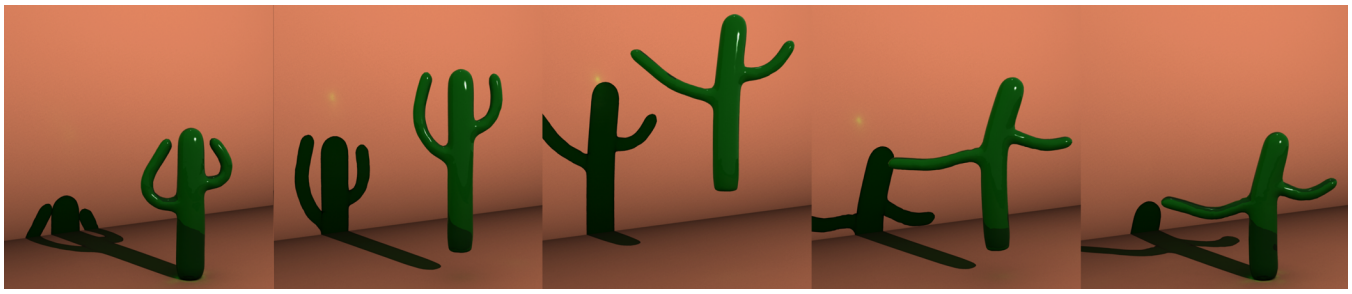

Figure 7: *Stills from an upward jumping cactus animation. The cactus uses its body and arms to create the velocity needed for the jump and to prepare for the landing.*

was introduced in Section 5. The goal is to impose sparse constraints on the warped motion. The warping map has been used for generating motion in [Huang et al. 2011] and [Barbič et al. 2012]. However, we consider a different problem from the aforementioned papers. Huang et al. [2011] generate motions that interpolate full keyframes. For full keyframes, one can use the inverse of the warp map to get "linearized" keyframes that will be warped to the desired positions. Using these "linearized" keyframes, an optimization over the warped motion can be avoided. This is an effective technique, but the approach is limited to full keyframes for the positions. In [Barbič et al. 2012], the positions of a few vertices are constrained, but the warping is only used as a post-process after a motion has been generated based on linearized elasticity. Therefore, only a linear problem need to be solved, but the position constraints are no longer enforced. Examples that illustrate the resulting deviation form the prescribed positions are shown in [Li et al. 2013].

Controlling the warped (instead of the linear) motion turns the linear optimization problem of Section 5 into a nonlinear optimization problem. For this problem the techniques we developed to get a low-dimensional space of admissible motions in Section 5 become even more important. We show that an analog construction of low-dimensional spaces is possible for the nonlinear problem, as well.

To be able to impose constraints on the warped motion $W(u(t))$, we exchange E_C by the energy

$$\begin{aligned} E_{WC}(u) \quad = \quad & \frac{1}{2}\sum_{k=0}^{m}(c_A \left\|A_k W(u(t_k)) - a_k\right\|^2 \\ & + c_B \left\|B_k \mathrm{D}W\, \dot{u}(t_k) - b_k\right\|^2), \end{aligned}$$

where c_A, c_B, A_k, B_k, a_k, and b_k are defined as in Section 5. Then the objective functional we consider is

$$\mathcal{E}(u) = E(u) + E_{WC}(u). \tag{13}$$

The resulting motion is the warped solution $W(u(t))$. Analogous to the linear problem treated in Section 5, we have the following theorem. For a proof, we refer to [Schulz et al. 2014].

Theorem 2 *The minimizers of the energy $E(u) + E_{WC}(u)$ among all functions in the Sobolev space $H^2((t_0, t_m), \mathbb{R}^n)$ are of the form (9) and are twice differentiable at any node t_k where no velocity is prescribed and once differentiable at all other nodes.*

The theorem justifies the use of techniques developed in Section 5 to solve this problem. Instead of optimizing over the infinite-dimensional Sobolev space, we use the representation (9) of the solution and optimize over the coefficients of the wiggly splines ω_i. We want to emphasize that though we can use vibration modes and wiggly splines to find a solution u of the minimization problem, the resulting motion $W(u)$ is not a linear combination of vibration modes since the nonlinear warping map couples the modes. As in Section 5, we restrict the sum in (9) to the lowest 10-50 modes and use an SVD to eliminate the linear constraints resulting from the continuity assumptions (guaranteed by the theorem) and the boundary conditions. The resulting optimization problem is low-dimensional and the objective functional is a sum of a quadratic energy and a nonlinear least-squares energy. To solve the problem, we use a standard Gauss–Newton solver. In our experiments, the resulting runtime for solving the problems was on the order of seconds. The most expensive part of the evaluation is that of E_{WC} and its gradient. This part is expensive because we need to warp the whole geometry to evaluate warped positions of the selected parts or their velocities. This could be accelerated by using a reduced space for the warped results. The solver could also be parallelized.

8 Experiments and Discussion

The supplementary video shows examples of motions produced using our method. Table 1 contains details on the size of the meshes, dimension of the reduced spaces, and runtime. All objects are tet meshes and we used linear elasticity with homogeneous and isotropic materials and rotation strain warping for all examples. To

convert the surface meshes to volumetric meshes, we first coarsen the triangle meshes using the approach of Valette et al. [2004] (and the implementation provided by the authors) and then use *tetgen* [Si and Gärtner 2005] to generate the tet meshes. In this section, a wiggly spline means a vector-valued spline that describes a motion (and not a one-dimensional spline).

Models	Tets	Ws	Sub	Dim	Pre	Opt
Block (stretch)	2015	1	30	30	2.4s	0.3s
Block (twist)	2015	1	30	30	2.4s	1.8s
Block (tilt)	2036	1	30	30	2.6s	1.7s
Buddha	77642	1	10	30	8.8s	45s
X walk	3002	5	30	60	4.4s	26s
X hand spring	3002	3	30	60	4.4s	4.9s
X jump	3002	3	30	60	4.4s	10s
cactus upward jump	8370	3	30	60	9.7s	16.8s
cactus forward jump	8370	7	30	60	10.2s	48.6s

Table 1: *Statistics measured on a custom Laptop. From left to right: number of tets, number of composite splines, dimension of the reduced space, dimension of the resulting optimization problems (for each spline), time for setting up the optimization problems, and total time for solving the optimization problems.*

The first sequence of examples shows a block that stretches, stretches and twists, and stretches and tilts. All motions are computed using one partial keyframe and boundary conditions (the rest state and zero velocity). In addition, we fix the base of the block to the ground. This means that the space $\mathbb{R}^n$ of all possible deformations excludes the coordinates of the vertices at the base of the block. We also compute the eigenmodes of the constrained vibrations. Since we have one partial keyframe, the wiggly spline has two segments. This means, we have $8d$ wiggly spline coefficients $w^l_{i,k}$ to determine, where d is dimension of the reduced space. The C^2-continuity conditions determine $3d$ and the boundary conditions $4d$ degrees of freedom. Hence, the dimension of the optimization problem equals the dimension of the reduced space, which is 30 for all three examples. For all three motions, we generated the partial keyframe by moving the top of the block in space. The stretch-and-twist-motion in Figure 6 illustrates the effect of the rotation strain warping. Using only linear elasticity would introduce strong artifacts; twists are difficult for linear elasticity. The warping nicely prevents such artifacts.

The second example, the dancing Buddha, is modeled with a circular wiggly spline. The base of the object is fixed to the ground. For snapshots see Figure 2. Since velocities are not prescribed, it is twice continuously differentiable everywhere. It is modeled with three partial keyframes, which we generated by selecting a part of the model and translating it in space, see Figure 8. For this setting, the dimension of the optimization problem is 30. This is three times the dimension of the reduced space, which is 10. The resulting motion shows interesting secondary motion effects.

The third sequence of examples shown in Figures 1 and 4 present a "letter X" soft body character that walks and performs a handspring. The walk is composed of five wiggly splines, one for each step. In each step, one foot is fixed to the ground with equality constraints. The first wiggly spline starts with the rest shape and zero velocity and has a partial keyframe at the end. The position of the moving foot and a downwards velocity at this foot is specified. This is illustrated in Figure 8. The following wiggly splines (the steps) interpolate the position and velocity the preceding spline ended with. Each of the splines ends with a partial keyframe: the position of the sole (of the moving foot) is specified and a downward pointing velocity is prescribed. To calculate the motion, five optimization problems must be solved. Each of the problems is 60-dimensional. Table 1 shows the total time required to solve all five optimization problems. In this example, we used high least-squares weights a_i to ensure that the partial keyframes are well-approximated. The handspring animation is modeled analogously to the walk. It is composed of three wiggly splines. In each spline either both hands or both feet are constrained to the ground. The first spline starts with rest shape and zero velocity and ends with a partial keyframe, which prescribes the position and velocity of the hands. The second spline starts with the position and velocity the first spline ended with. It ends with a partial keyframe, which prescribes the position and velocity of the feet. The last spline starts with the position and velocity the second spline ended with and ends with the rest shape and zero velocity. For an illustration, see Figure 1.

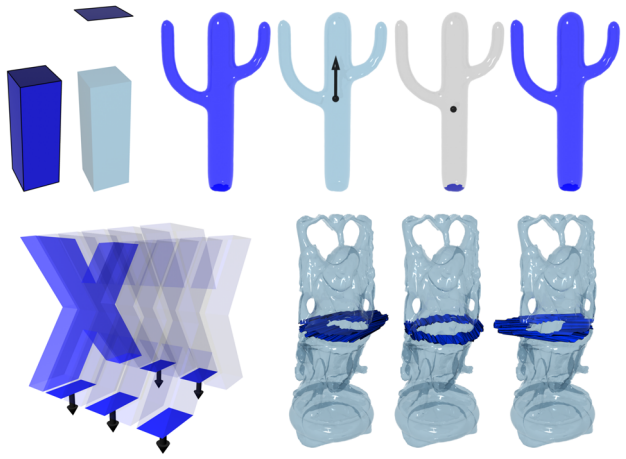

Figure 8: *Illustrations of the sparse constraints used to generate the animations.*

The fourth type of examples are jumps of the cactus and the letter X, see Figures 3, 5, and 7. The jumping X and the upward jump of the cactus are modeled with three splines: preparing to jump, jumping, and landing. In the first spline, the feet are fixed to the ground. It starts with the rest pose and zero velocity. At the end only the velocity of the center of mass is prescribed. During the jump, the object is unconstrained. Since the warp map has the translations in its kernel, we fix the position of the center of mass during the optimization and add the motion of the center of mass after the optimization. In addition, we remove the linearized rotations from the subspace basis for the optimization, to ensure that no rotation is generated. The motion of the center of mass is explicitly computed from its initial position and velocity. We choose the time of the landing such that the center of mass has reached a certain height over the ground. The spline that models the jump starts with the position and velocity the first spline ended with and ends with prescribed positions and velocities for the feet (relative to the center of mass and its velocity). The last spline again fixes the feet to the ground and starts with the positions and velocities the jump ended with and ends with the rest pose and zero velocity. For the triple jump of the cactus seven splines are used. The motion is modeled analogously to the other jumps, with the differences that the jumping phases is repeated three times.

In Figure 8, we illustrate constraint types we used to generate the motions. The dark blue parts describe the partial keyframes for the positions. The bright transparent parts are unconstrained. On the top left are the constraints used to create the twisting block animation. Next on the top right are all keyframes for the upward jumping cactus. The arrow at the second cactus describes the velocity constraint for the center of mass. The single black dot at the

third cactus indicates the position of the center of mass. Constraints for the positions of the "foot" are specified relative to the center of mass. The bottom left illustrates the one full and the five partial keyframes and velocities used for the five step X walk. The three partial keyframes used to create the dancing Buddha animation are shown on the bottom right.

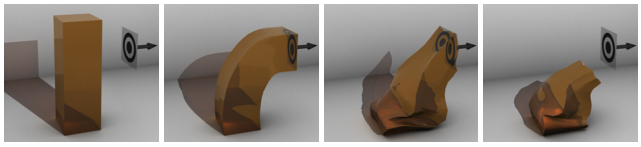

Figure 9: *Comparison of results of our nonlinear framework to alternative linear approaches. From left to right: a simple setup, solution of the nonlinear problem of controlling the warped motion (Section 7), solution of the linear problem (Section 5), and warping applied to the solution of the linear problem.*

In Figure 9, we compare results produced by our nonlinear problem modeling (Section 7) with the linear modeling (Section 5) for a simple setup of partial positions and velocities. The example illustrates the limitation of the linear model, which is that larger deformations lead to strong linearization artifacts. Such examples indicate that the nonlinear modeling is needed to produce interesting motions that undergo larger deformations. In addition to the solution of the linear problem, we show what happens if we warp the motion computed with the linear model. The resulting motion no longer satisfies the constraints.

In our experiments, the Newton scheme could solve the optimization problems within 5-30 iterations. As an alternative, we used the BFGS scheme with a warm start using the Hessian at the rest pose. The BFGS scheme was faster than the Gauss–Newton on many examples, but failed for some of the examples. For example, using BFGS, we could solve the optimization problem for the Buddha in 11.4s—almost four times faster than the Gauss–Newton scheme. The times listed in Table 1 are the total times required to set up and solve all the optimization problems needed to generate a motion. The number of splines used for each example is listed as well. The times are all produced with the Gauss–Newton solver.

One difference our scheme has compared to most other spacetime optimization schemes, is that we do not need a time discretization. The resulting advantage is that the dimension of the optimization problems we obtain is much lower. In addition, our scheme does not suffer from stability issues caused by time discretization and therefore robustly computes motion from sparse input and even wiggly motion. For time discretization, finite differences are typically used. Kass and Anderson [2008] discuss the resulting stability issues for the one-dimensional spacetime optimization problems.

9 Conclusion

We present an approach to compute animations of deformable objects from partial keyframes via constrained spacetime optimization. The scheme is based on an explicit characterization of solutions of a spacetime optimization problem with sparse constraints. Using this characterization, we can restrict the optimization to a low-dimensional search space and do not need a time discretization of the motion. Since we combine linearized elasticity with rotation strain warping, the scheme can deal with large deformations. For controlling the warped motion, derivatives of the warp map are needed. We derive a representation of these derivatives that allows for an efficient and robust evaluation. Our formulation of the spacetime optimization problem robustly computes motions from sparse input. This only requires solving a low-dimensional nonlinear least-squares problem, which can be done with standard solvers.

9.1 Limitations and challenges

Currently, the most expensive part in the optimization is the evaluation of the warp map and its derivative. For this evaluation, the whole object is warped. Since we restrict the optimization to a subspace, one idea would be to speed up the evaluation of the warp map by restricting the result to a subspace as well.

In our examples, we only work with homogeneous materials. It would be interesting to test inhomogeneous materials. One challenging problem would be to extend the method to deformable objects with a skeleton. This would require an efficient spacetime optimization for deformable objects that are coupled with rigid bodies.

Our current framework is limited to equality constraints, both at the keyframes and for the whole animation. A challenge for our approach (and for spacetime constraints in general) is to find an effective way to integrate inequality constraints into the spacetime optimization. Related is the problem of collision handling. Currently, we model collision using the keyframes, velocities, and equality constraints. Examples are the walk and handspring of X in which the feet and hands collide with the ground.

Acknowledgements

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

Appendix

Computation of the matrix exponential. Rotation strain warping of the positions and velocities, see Section 6, involves the evaluation of the matrix exponential of an antisymmetric 3×3 matrix J and its derivative. Here, we show how Rodrigues' rotation formula can be used for this. We set $\iota = \left\|\frac{1}{2}J\right\|$, where the norm is the Frobenius norm. Then, Rodrigues' rotation formula is

$$\exp(J) = Id + \frac{\sin\iota}{\iota}J + \frac{1-\cos\iota}{\iota^2}JJ.$$

One can compute Dexp by a straight forward application of basic rules of differentiation. For small values of ι, the computation of the functions $\sin\iota/\iota$ and $(1-\cos\iota)/\iota^2$ is unstable since both the numerator and the denominator converge to zero. A robust way to handle this problem is to replace the functions by the following truncated Taylor series for small values of ι

$$\frac{\sin\iota}{\iota} \approx 1 - \frac{1}{6}\iota^2 + \frac{1}{120}\iota^4 \qquad \frac{1-\cos\iota}{\iota^2} \approx \frac{1}{2} - \frac{1}{24}\iota^2 + \frac{1}{720}\iota^4$$

and analogous for the derivatives. We used the truncated Taylor series for $\iota < 0.2$. For a thorough treatment of the subject, we refer to [Wisniewski 2010, Chapter 8.2].

References

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

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

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

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

BERGOU, M., MATHUR, S., WARDETZKY, M., AND GRINSPUN, E. 2007. TRACKS: Toward Directable Thin Shells. *ACM Trans. Graph. 26*, 3, 50:1–50:10.

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

COHEN, M. F. 1992. Interactive spacetime control for animation. *Proc of ACM SIGGRAPH 26*, 293–302.

COROS, S., MARTIN, S., THOMASZEWSKI, B., SCHUMACHER, C., SUMNER, R., AND GROSS, M. 2012. Deformable objects alive! *ACM Trans. Graph. 31*, 4, 69:1–69:9.

FANG, A. C., AND POLLARD, N. S. 2003. Efficient synthesis of physically valid human motion. *ACM Trans. Graph. 22*, 3, 417–426.

GLEICHER, M. 1997. Motion editing with spacetime constraints. In *Proc. of Symp. on Interactive 3D Graphics*, 139–148.

HEEREN, B., RUMPF, M., WARDETZKY, M., AND WIRTH, B. 2012. Time-discrete geodesics in the space of shells. *Comp. Graph. Forum 31*, 5, 1755–1764.

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

HUANG, J., TONG, Y., ZHOU, K., BAO, H., AND DESBRUN, M. 2011. Interactive shape interpolation through controllable dynamic deformation. *IEEE Transactions on Visualization and Computer Graphics 17*, 7, 983–992.

KASS, M., AND ANDERSON, J. 2008. Animating oscillatory motion with overlap: wiggly splines. *ACM Trans. Graph. 27*, 3, 28:1–28:8.

KILIAN, M., MITRA, N. J., AND POTTMANN, H. 2007. Geometric modeling in shape space. *ACM Trans. Graph. 26*, 3, 64:1–64:10.

KIM, T., AND JAMES, D. L. 2009. Skipping steps in deformable simulation with online model reduction. *ACM Trans. Graph. 28*, 5, 123:1–123:9.

KIM, J., AND POLLARD, N. S. 2011. Fast simulation of skeleton-driven deformable body characters. *ACM Trans. Graph. 30*, 5, 121:1–121:19.

KONDO, R., KANAI, T., AND ANJYO, K.-I. 2005. Directable animation of elastic objects. In *Symp. Comp. Anim.*, 127–134.

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

LI, S., HUANG, J., DESBRUN, M., AND JIN, X. 2013. Interactive elastic motion editing through spacetime position constraints. *Computer Animation and Virtual Worlds 24*, 3-4, 409–417.

MARTIN, S., THOMASZEWSKI, B., GRINSPUN, E., AND GROSS, M. 2011. Example-based elastic materials. *ACM Trans. Graph. 30*, 4, 72:1–72:8.

MCNAMARA, A., TREUILLE, A., POPOVIĆ, Z., AND STAM, J. 2004. Fluid control using the adjoint method. *ACM Trans. Graph. 23*, 3, 449–456.

MÜLLER, M., DORSEY, J., MCMILLAN, L., JAGNOW, R., AND CUTLER, B. 2002. Stable real-time deformations. In *Proc. Symp. Comp. Anim.*, 49–54.

NICKELL, R. 1976. Nonlinear dynamics by mode superposition. *Comput. Meth. Appl. Mech. Eng. 7*, 1, 107–129.

PENTLAND, A., AND WILLIAMS, J. 1989. Good vibrations: modal dynamics for graphics and animation. *Proc. of ACM SIGGRAPH 23*, 207–214.

POPOVIĆ, J., SEITZ, S. M., AND ERDMANN, M. 2003. Motion sketching for control of rigid-body simulations. *ACM Trans. Graph. 22*, 4, 1034–1054.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. Graph. 23*, 3, 514–521.

SCHULZ, C., VON TYCOWICZ, C., SEIDEL, H.-P., AND HILDEBRANDT, K., 2014. Proofs of two theorems concerning sparse spacetime constraints. http://arxiv.org/abs/1405.1902.

SI, H., AND GÄRTNER, K. 2005. Meshing piecewise linear complexes by constrained Delaunay tetrahedralizations. In *Proceedings of the 14th International Meshing Roundtable*. Springer, 147–163.

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

SULEJMANPAŠIĆ, A., AND POPOVIĆ, J. 2005. Adaptation of performed ballistic motion. *ACM Trans. Graph. 24*, 1, 165–179.

SUMNER, R. W., ZWICKER, M., GOTSMAN, C., AND POPOVIĆ, J. 2005. Mesh-based inverse kinematics. *ACM Trans. Graph. 24*, 3, 488–495.

TAN, J., TURK, G., AND LIU, C. K. 2012. Soft body locomotion. *ACM Trans. Graph. 31*, 4, 26:1–26:11.

TREUILLE, A., MCNAMARA, A., POPOVIĆ, Z., AND STAM, J. 2003. Keyframe control of smoke simulations. *ACM Trans. Graph. 22*, 3, 716–723.

VALETTE, S., AND CHASSERY, J.-M. 2004. Approximated centroidal Voronoi diagrams for uniform polygonal mesh coarsening. *Computer Graphics Forum 23*, 3, 381–389.

VON TYCOWICZ, C., SCHULZ, C., SEIDEL, H.-P., AND HILDEBRANDT, K. 2013. An efficient construction of reduced deformable objects. *ACM Trans. Graph. 32*, 6, 213:1–213:10.

WISNIEWSKI, K. 2010. *Finite Rotation Shells*. Springer.

WITKIN, A., AND KASS, M. 1988. Spacetime constraints. *Proc. of ACM SIGGRAPH 22*, 159–168.

WOJTAN, C., MUCHA, P. J., AND TURK, G. 2006. Keyframe control of complex particle systems using the adjoint method. In *Proc. Symp. Comp. Anim.*, 15–23.

XU, D., ZHANG, H., WANG, Q., AND BAO, H. 2005. Poisson shape interpolation. In *Symp. Solid and Phys. Model.*, 267–274.