COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2013; **24**:409–417
Published online 13 May 2013 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1521

SPECIAL ISSUE PAPER

Interactive elastic motion editing through space–time position constraints

Siwang Li[1], Jin Huang[1]*, Mathieu Desbrun[2] and Xiaogang Jin[1]

[1] State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310058, China
[2] Department of Computing & Mathematical Sciences MS 305-16, California Institute of Technology, 1200 E. California Boulevard Pasadena, CA 91125, USA

ABSTRACT

We present an intuitive and interactive approach for motion editing through space–time constraints on positions. Given an input motion of an elastic body, our approach enables the user to interactively edit node positions in order to alter and fine-tune the motion. We formulate our motion editing as an optimization problem with dynamics constraints to enforce a physically plausible result. Through linearization of the editing around the input trajectory, we simplify this constrained optimal control problem into an unconstrained quadratic optimization. The optimal motion thus becomes the solution of a dense linear system, which we solve efficiently by applying the adjoint method in each iteration of a conjugate gradient solver. We demonstrate the efficiency and quality of our motion editing technique on a series of examples.

KEYWORDS
motion editing; elastic animation; space–time constraints; model reduction; adjoint method

Supporting information may be found in the online version of this article.

***Correspondence**
Jin Huang, State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310058, China.
E-mail: hj@cad.zju.edu.cn

1. INTRODUCTION

Physically based simulation of deformable objects has become pervasive in computer graphics. Although current physics-based methods can generate exquisite results, fine-tuning of an animation often requires a time-consuming trial-and-error process to find the right simulation parameters. Editing an existing simulated sequence to meet user-specified position constraints without making the resulting motion visually implausible is therefore crucial in practice. Solutions that offer accurate and flexible control and fast feedback, yet maintain physical plausibility of the edited motion remain rare.

Existing methods for controlling elastic animation can be roughly classified into two categories: key frame interpolation and sequence editing. The key frame interpolation methods do not take an existing animation as input: they typically specify position constraints for all the nodes of a few key frames. Significant user interaction is required because one cannot prescribe only the positions of a portion of the object, which reduces control flexibility. Conversely, sequence editing methods try to find small alterations of an input animation to satisfy user-specified constraints. This type of approach lends itself well to partial editing of an input sequence in space and time. Recently, Barbič *et al.* [1] introduced an interactive editing approach for given complex deformable object animations that allows for direct manipulation of the deformable body at any time frame; however, this latest development in motion editing does not provide precise *position* control.

In this paper, we propose an efficient solution to motion editing that offers tight position control at interactive rates. Our approach casts the editing process as a constrained optimal control problem. Previous methods based on optimal control are typically compute-intensive because of their high-dimensional and nonlinear nature. Instead, we simplify this formulation via model reduction and linearization. The resulting unconstrained quadratic optimization problem is significantly smaller than the original one. To further improve numerical performance and allow for interactive feedback, we efficiently solve the resulting dense linear system using an iterative solver and the adjoint method. Our specific combination of well-known numerical techniques (model reduction, linearization around the input trajectory, and adjoint method) results in a novel motion editing approach with the following unique features:

- it allows for interactive editing;
- it inherits the physical behavior of the input animation;
- it remains physically plausible for reasonable changes of the animation;
- it allows for partial position constraints in both space and time; and
- it offers tight position control.

2. RELATED WORK

Interactive control for physically based animation has received significant attention in the past decade. Although motion editing has been proposed for character animation [2], rigid body simulation [3,4], and fluid simulation [5–7], methods designed specifically to edit elastic object animation can be classified into two main categories: key frame interpolation and sequence editing. (Note that we do not review techniques that only add dynamic details to an existing coarse animation, such as that in [8].) Representative interpolation techniques include those in [9–11]. These methods use elastic dynamics to provide interpolation between a set of key frames, thus generating an animation with visually plausible behavior. However, much alike traditional geometrical interpolation methods, they are not suitable to edit an existing physics-based animation and cannot accommodate partial position constraints as a means to locally control an animation. Sequence editing methods, on the contrary, take an existing animation as input and locally (in both time and space) edit it. Kondo *et al.* [12] presented a method to edit an input animation by user-specified key frames and trajectories. To obtain more natural results, researchers adopted space–time constraints [13–16] and formulated motion editing as an optimal control problem constrained by physical equations of motion. Because the resulting optimization problem is often nonlinear and involving a large number of variables, such methods are typically quite slow but offer tight position control.

Linearization is a common approach to reduce the complexity of a physical model, unfortunately leading to noticeable artifacts when applied to large deformations. Corotational methods [17–19] have been employed to drastically reduce linearization artifacts. Choi and Ko further proposed modal warping [20] to handle rotations in modal space, whereas Huang *et al.* [10] linearized the equations of motion in the rotation–strain space to support large deformations. In Barbič *et al.* [21], linearization was made around the input trajectory to treat moderate perturbations from user-specified external forces. Our approach is inspired by this approach, but we instead directly apply position constraints in order to edit an existing animation.

To reduce simulation complexity, model reduction [22,23] has been widely applied in recent years. Barbič and James [24] added modal derivatives to better approximate large deformations. Choi and Ko [25] proposed an approximate time integration to compute positions and reconstruct the shape without large distortions. Kim and James [26] even proposed an approach to adaptively build reduced models as the simulation progresses. Model reduction has also been used in the context of space–time constraints. To avoid noticeable artifacts when applied to large deformations, the interpolation method of Barbič *et al.* [9] built the reduced equations of motion from the vibration modes around the key frames. By combining model reduction and linearization around the key frames, the interpolation method of Hildebrandt *et al.* [11] achieved much faster performance. Recently, Barbič *et al.* presented an approach to adjust input animations interactively [1]: they solved the optimal control problem in a linearly reduced space around the rest shape. To alleviate the artifacts caused by linearization, they use post-warping to reconstruct the final shape. As a consequence, their results may not satisfy the user-specified constraints.

Many optimal control problems involve the gradient computation of the cost function with respect to control variables using the discrete adjoint method. The adjoint method [27] has been used for fluid control [28], cloth control [29], and elastic object interpolation [9]. In our approach, this adjoint method is used, but in the particularly simple case of linearized and reduced equations, which makes it two orders of magnitude faster than previous usages.

3. ALGORITHM

We now present our algorithm that amounts to an ordinary differential equation (ODE)-constrained optimization: we solve for a new trajectory that matches user-specified constraints while satisfying simplified equations of motion.

3.1. Trajectory Offset as ODE

The dynamics of an elastic object discretized through a mesh with N nodes is formulated as follows:

$$\tilde{M}\ddot{\tilde{u}} = \tilde{f}(\tilde{u}, \dot{\tilde{u}}) + \tilde{g} \tag{1}$$

where $\tilde{u}(t)$, $\tilde{f}(\tilde{u}, \dot{\tilde{u}})$, and $\tilde{g}(t) \in \mathbb{R}^{3N}$ are respectively the displacements (with velocity $\dot{\tilde{u}}$ and acceleration $\ddot{\tilde{u}}$), internal elastic forces, and external forces of the nodes, whereas $\tilde{M} \in \mathbb{R}^{3N\times 3N}$ is the mass matrix.

Modal reduction is a conventional technique to reduce the space of deformations to a set of vibration modes, that is especially useful for solving space–time problem efficiently. As in [9], we project the constraint Equation (1) into the reduced space of r vibration modes using a matrix $W \in \mathbb{R}^{3N\times r}$ satisfying $W^T \tilde{M} W = I$, yielding a simplified set of equations:

$$\ddot{u} = f(u, \dot{u}) + g \tag{2}$$

where the modal coordinates,

$$u = W^T \tilde{M}\tilde{u}, \quad g = W^T \tilde{g}, \\ f(u, \dot{u}) = W^T \tilde{f}(Wu, W\dot{u}) \tag{3}$$

If one stacks the resulting displacements and velocities into a configuration vector $q = (\dot{u}^T, u^T)^T \in \mathbb{R}^{2r}$, the second-order Equation (2) is turned into a first-order ODE

$$\dot{q} = A(q) + Bg \tag{4}$$

where we used the following:

$$A(q) = \begin{pmatrix} f(q) \\ \dot{u} \end{pmatrix}, \quad B = \begin{pmatrix} I \\ 0 \end{pmatrix} \tag{5}$$

From a trajectory $q(t)$ satisfying the aforementioned equation of motion (Equation (4)), a small modification of the external forces from $g(t)$ to $g(t) + w(t)$ (where $w(t)$ is small in magnitude) induces a nearby trajectory $q(t)+z(t)$. Through linearization of Equation (4), one can easily show [21] that $z(t)$ and $w(t)$ are related via

$$\dot{z} = \frac{\partial A(q)}{\partial q} z + Bw \tag{6}$$

This last equation indicates how the *configuration offset* $z(t)$ of a trajectory $q(t)$ depends on the addition of external forces $w(t)$. If we use an implicit integrator [30] with a time step h to discretize this differential equation, we obtain

$$z_{i+1} = \left(I - h \left. \frac{\partial A(q)}{\partial q} \right|_{q_i} \right)^{-1} (z_i + hBw_{i+1}) \tag{7}$$

where z_i is the configuration offset at $t_0 + ih$, and w_{i+1} is the additional external force acting between $t_0 + ih$ and $t_0 + (i+1)h$. The value z_0 is the initial condition that we set to $z_0 = w_0$.

Given an n-frame animation sequence $q_i, i = 0, \ldots, n-1$, the equations we presented earlier represent a linear relationship between the control variables $w = \left(w_0^T, w_1^T, \ldots, w_{n-1}^T\right)^T$ (initial condition and external forces acting on the shape) and the resulting trajectory offset $z = \left(z_0^T, \ldots, z_{n-1}^T\right)^T$. We encode them as a linear system of the following form:

$$Fz + Gw = 0 \tag{8}$$

Given a set of control variables w, solving for the trajectory offsets z is efficiently achieved by incrementally evaluating the offsets via Equation (7).

We now need to find a way to define what the optimal control variables are in order to achieve user-specified position constraints in time, which we address next.

3.2. Cost Functional

Trajectory editing often proceeds by starting from an input trajectory q and constraining the positions of some nodes in $\mathbb{R}^3$ in several selected key frames.

3.2.1. Position Constraints

To account for general point-to-point constraints, we assume that the user defines the set of position constraints through a linear equation:

$$\tilde{C} \begin{pmatrix} \tilde{u}_0 \\ \vdots \\ \tilde{u}_{n-1} \end{pmatrix} = \tilde{u}^c \tag{9}$$

where $\tilde{u}_i$ is the resulting node position at frame i. We can further rewrite these linear constraints on positions as $Cz = z^c$, where

$$C = \tilde{C} \begin{pmatrix} (0\ W) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & (0\ W) \end{pmatrix}, \ z^c = \tilde{u}^c - Cq \tag{10}$$

3.2.2. Constrained Optimization

We now wish to define an optimal trajectory by finding small external forces w_i that make the resulting physically derived trajectory offset as close as possible to what the user specified. We propose to define a cost function E that measures how well the node positions in full space $\tilde{u}$ match the user-specified position constraints and how small the additional external forces w are through the following:

$$E(z, w) = \frac{1}{2}\left(\|P(Cz - z^c)\|^2 + \|Rw\|^2 \right) \tag{11}$$

where P and R are weight matrices. Our cost is thus simply the P-weighted L_2 norm between the actual versus desired positions, plus the R-weighted L_2 norm of the added external forces. Because w and z are linearly dependent (Equation (8)), the second term implies a minimal deviation from the reference motion.

The final optimization problem is therefore the following:

$$\min_w E(z, w), \ \text{s.t.}\ Fz + Gw = 0 \tag{12}$$

It will enforce a physical trajectory via the constraints but will match the user-specified edits of the animation through small added forces.

4. NUMERICAL METHOD

We now present a numerical approach to solve the trajectory editing problem we formulated in Equation (12). We leverage the structure of our formulation to devise a particularly efficient computational procedure.

4.1. Unconstrained Optimization through Substitution

The constrained optimization (Equation (12)) can first be easily turned into an unconstrained problem: by substituting $z = -F^{-1}Gw$ into Equation (11), solving the optimization reduces to solving the following linear system:

$$Hw = -b \tag{13}$$

where $H = \frac{\mathrm{d}^2 E(z,w)}{\mathrm{d}w^2}$, and $b = \left.\frac{\mathrm{d}E(z,w)}{\mathrm{d}w}\right|_{w=0}$.

Note that both $H \in \mathbb{R}^{nr\times nr}$ and $b \in \mathbb{R}^{nr}$ (where n is the number of frames in the input sequence) are constant and can be precomputed. However, this matrix H is typically a large and non-sparse matrix. Direct solvers, such as Cholesky factorization, are ill-suited to solve such a large and dense linear system. Iterative methods, such as the conjugate gradient method, would not fare any better as they involve matrix–vector multiplications, which are also quite computationally expensive for dense matrices.

One of our contributions is to propose an efficient approach to solve this problem through conjugate gradient method with the use of the *adjoint method*. We first rewrite the product Hw, evaluated repeatedly in the iterations of the conjugate gradient method, using the following:

$$Hw = \frac{\mathrm{d}E(z,w)}{\mathrm{d}w} - b \tag{14}$$

As we show next, the gradient $\frac{\mathrm{d}E(z,w)}{\mathrm{d}w}$ can be computed efficiently using the adjoint method, leading to a significant acceleration of each iteration of the conjugate gradient process.

4.2. Adjoint Method

The adjoint method is a common technique in optimal control, which has been widely used in computer graphics [9,28,29]. We can use it in our framework for a fast evaluation (through forward and backward propagation) of the gradient of our cost function—and thus of the matrix–vector multiplication Hw through Equation (14).

We proceed in two passes over the n-frames of the trajectory. First, the configuration offsets z_i are integrated forward in time according to Equation (7). We then proceed backward from the last configuration to generate a series of adjoint vectors r_i through the following:

$$r_{n-1} = \partial E(z,w)/\partial z_{n-1}$$

iteratively followed by

$$r_i = \left(\frac{\partial z_{i+1}}{\partial z_i}\right)^{\mathrm{T}} r_{i+1} + \frac{\partial E(z,w)}{\partial z_i}$$

where

$$\frac{\partial z_{i+1}}{\partial z_i} = \left(I - h\left.\frac{\partial A(q)}{\partial q}\right|_{q_i}\right)^{-1}$$

and

$$\frac{\partial E(z,w)}{\partial z} = C^{\mathrm{T}}P^{\mathrm{T}}P(Cz - z^c)$$

Finally, at the end of the second pass, the gradient of E with respect to w is obtained through the following:

$$\frac{\mathrm{d}E(z,w)}{\mathrm{d}w} = \begin{pmatrix} hB^{\mathrm{T}}\left(\frac{\partial z_1}{\partial z_0}\right)^{\mathrm{T}} r_1 \\ \vdots \\ hB^{\mathrm{T}}\left(\frac{\partial z_n}{\partial z_{n-1}}\right)^{\mathrm{T}} r_n \end{pmatrix} + R^{\mathrm{T}}Rw$$

where the last term corresponds to the gradient of the second term of Equation (11). As the matrix $\partial z_{i+1}/\partial z_i$ depends only on the input trajectory and can thus be precomputed, these two passes described earlier only involve products and sums of small matrices and vectors, which is significantly faster than assembling H explicitly and calculating the matrix–vector product Hw (Figure 1). This adjoint-method-based evaluation is also more than two orders of magnitude faster than a nonlinear approach Table (1), as it would involve additional computational costs to calculate the stiffness matrices and to solve a series of linear systems.

5. RESULTS

We implemented our approach and tested it on a number of models.

5.1. Setup

The input of our approach is an elastic object's rest shape, its mass matrix $\tilde{M}$, a model for its internal forces $\tilde{f}$, a time

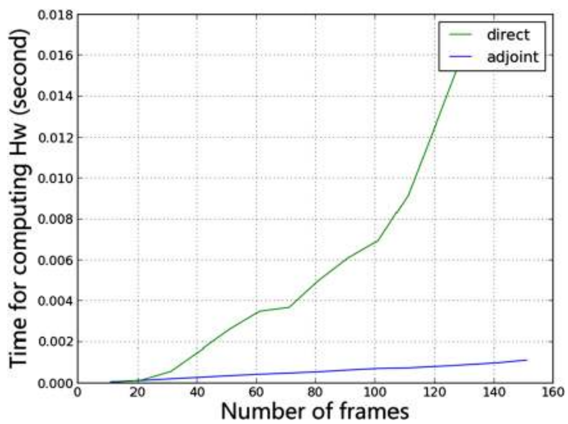

Figure 1. The adjoint method is used to greatly accelerate the (linear) conjugate gradient process; in this plot, the average time of calculating Hw directly (shown in green) is compared with the adjoint-based computation (shown in blue) for a model with $r = 32$.

Comp. Anim. Virtual Worlds 2013; **24**:409–417 © 2013 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

Table 1. Performance measured on a desktop PC with Intel Core i7-930, 8-core, 2.80 GHz, 4 GB memory.

						Forward pass		Backward pass		Total
Model	#verts	#nodes	#elems	#modes	n	Linear	Nonlinear	Linear	Nonlinear	(linear)
Plant	2464	3737	10582	40	200	0.0019	0.2041	0.0016	0.2641	0.0229
Bridge	17874	5629	15466	40	181	0.0016	0.1741	0.0014	0.2493	0.0309
Dino	28098	1493	5249	50	200	0.0030	0.3893	0.0025	0.6278	0.0323
Dino (path)	28098	1493	5249	50	400	0.0061	0.7879	0.0053	1.2613	0.4670

All timings are given in seconds. From left to right: number of triangle mesh vertices, tetrahedral mesh nodes, elements, modes, and total frames of the animation, time for each forward pass and backward pass in the adjoint method, and the total cost to solve our space–time optimization problem.

step size, and a sequence of vectors $\tilde{u}_i \in \mathbb{R}^{3N}$ representing the displacements $\tilde{u}_i$ in full 3D space with respect to the rest shape. In a preprocessing phase, we construct the basis matrix W used for model reduction, then project the input animation sequence into a series of low-dimensional coordinates $u_i = W^T \tilde{M} \tilde{u}_i$. Note that we compute W by applying mass-PCA (mass-scaled principal component analysis) to all frames in the input sequence and the modal derivatives at the rest shape as in [24]: the reduced coordinates still capture the input sequence well and express a rich set of deformations around the input trajectory. In our experiments, 40 to 50 modes suffice to produce plausible results for the type of editing we tried (Figure 2).

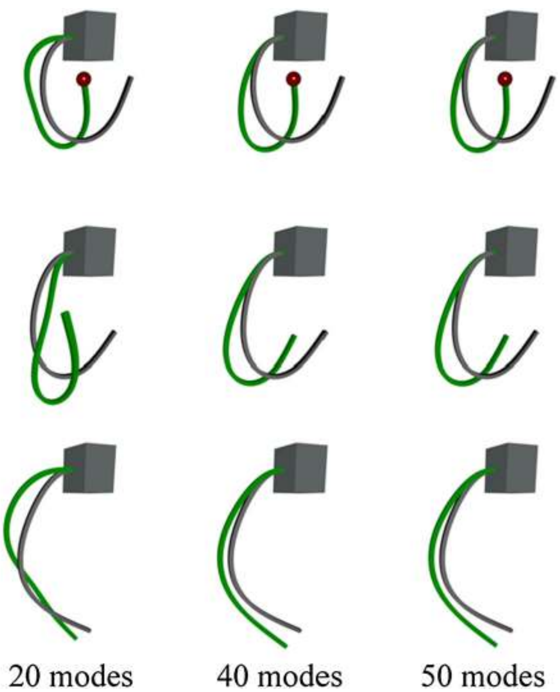

Figure 2. We apply the same position constraint (red sphere) to frame 110 of an input sequence (in gray) with 151 frames. Snapshots of the resulting motion (in green) at frames 110, 120 and 130 are shown from top to bottom, with three different numbers of modes. Plausible results are typically obtained when using about 40 modes.

5.2. Examples of Motion Editing

We first show the interactive motion editing process of a dinosaur animation generated by the physically based interpolation approach of Barbič *et al.* [9], as shown in Figure 3 and the accompanying video (Supporting information). We edit the motion by selecting a few nodes and dragging them to target positions; the animation adjusts immediately and accordingly. Because the left plant is small and far away from the dinosaur, a relatively large edit needs to be applied to make the dinosaur's head closer to the plant. However, as the linearization is made around the input sequence, the result appears artifact-free. To demonstrate that our approach can work with large models, we also show the motion editing process of a swinging rope bridge with complex topology (Figure 4). The input

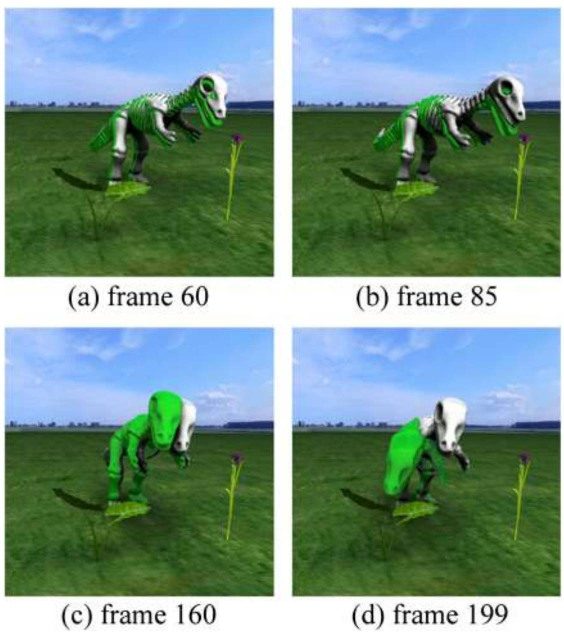

Figure 3. Because user-directed position control can be (even partially) imposed in space and time, we can intuitively modify an input animation (gray) and make the head of the dinosaur closer (green) to the flower (resp., leaves) at frame 85 (resp., 199).

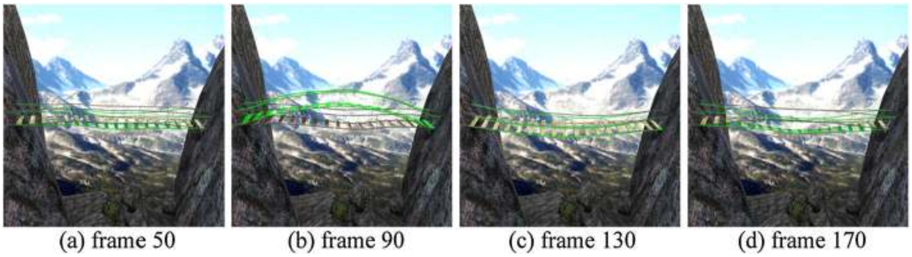

Figure 4. Adjusting the complex input motion of a swinging rope bridge. When the user edits the motion through position constraints, our system produces a new animation at interactive rates in which the constraints are visually met. The rope bridge in the input animation is displayed with textures, whereas the one in the output animation is displayed in green (see the accompanying video to see the animation from various viewpoints).

animation from [24] uses 80 modes and random impulses, but editing is carried out using only 40 modes. By specifying a few spatial and temporal constraints, we can significantly edit this animation, and the dynamics of the input sequence is well preserved. As advocated in [1], we used the L_2 norm in the objective function (Equation (11)) to induce a smooth resulting motion, even when the positional constraints are sparsely distributed in space and time.

5.3. Performance

The performance statistics of the examples discussed earlier are listed in Table 1. Efficiency is achieved thanks to the cumulative use of dimension reduction, linearization around trajectory, and the adjoint-based gradient evaluation. During user manipulation, we use the output of the previous motion edit as the initial value of the optimal control problem, thus achieving interactive editing because only a handful of iterations of the conjugate gradient solve are needed. Even for path control examples involving large amount of position constraints and starting from the initial trajectory with $w = 0$, our system can still generate a plausible new animation sequence in less than 1 s.

5.4. Comparisons

Figure 5 (see also the accompanying video) illustrates the ability of preserving the input behavior. After applying an edit on a single frame in a simulated sequence, the result still resembles the input animation as expected (upper row). To compare our results to the interpolation method of [9], we used several frames from the input sequence and the edited one as key frames for their approach; their interpolation result differs at times widely from the input (bottom row). In this comparison, both methods use the same reduction basis, time step, and elastic model.

Our approach also provides precise position control, which is especially useful to modify an animation sequence to follow a prescribed path, as shown in Figure 6(a,b). In this example, we sample points from the path of the tank

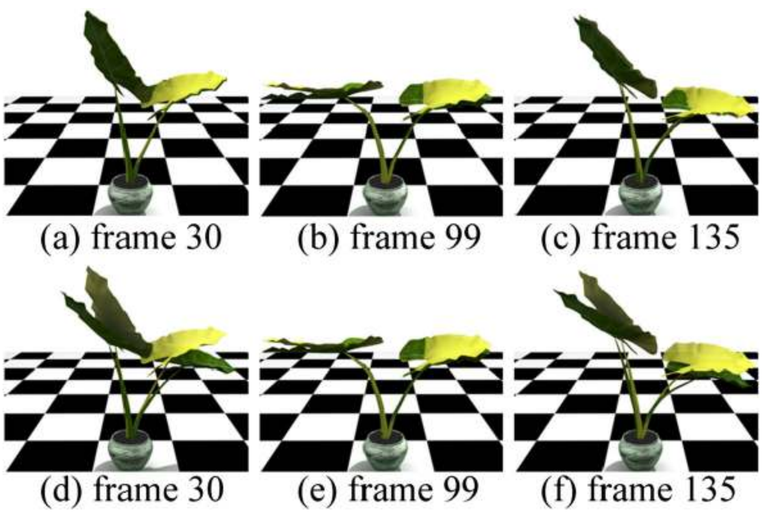

Figure 5. Our method (top row, in green) stays closer to the input behavior (in yellow) than the physically based interpolation method of Barbič *et al.* [9] (bottom row, in green).

Comp. Anim. Virtual Worlds 2013; **24**:409–417 © 2013 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

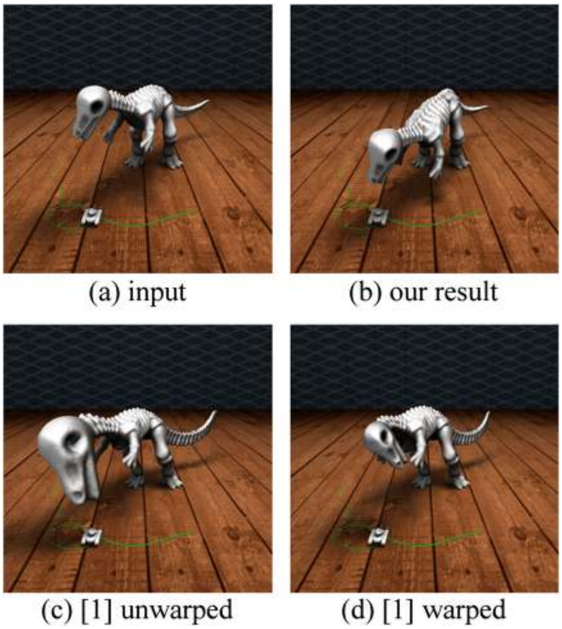

Figure 6. Our motion editing can make a dinosaur animation sequence closely track the trajectory (shown in green) of a tank toy. Images (a), (b), (c), and (d) are snapshots at frame 230 of, respectively, the input animation, our result, and the result of Barbič *et al.* [1] without and with warping.

toy every 20 frames and vertically raise these points as position constraints to control the head of the dinosaur. In the result animation, the head of the dinosaur follows the movement of the tank steadily and accurately. The method [1] proposed by Barbič and colleagues can be used for the same purpose. For efficiency, their method linearizes the equations of motion around the *rest* shape. Consequently, noticeable artifacts appear when the result shape contains large deformations: a small edit can lead to typical linearization distortions if the edited shapes happens to be far from the rest shape (Figure 6(c)). To reduce the artifacts introduced by the linearization, they adopt a post-warping that will no longer enforce the user-specified constraints. As a result, the positions of the manipulated nodes in the final animation may not matched the user's constraints, making the results unpredictable and thus hard to edit (Figure 6(d)). A real-time editing process for this comparison is shown in our accompanying video.

6. CONCLUSION

We have proposed a novel method to interactively edit an input animation of deformable objects through dimension reduction, linearization around input trajectory, and the adjoint method. Unlike previous methods, we can achieve simultaneously high efficiency, close preservation of input dynamics, physical plausibility, and flexible position control. Our test results and comparisons show that our simple method leads to intuitive motion editing that outperforms previous techniques.

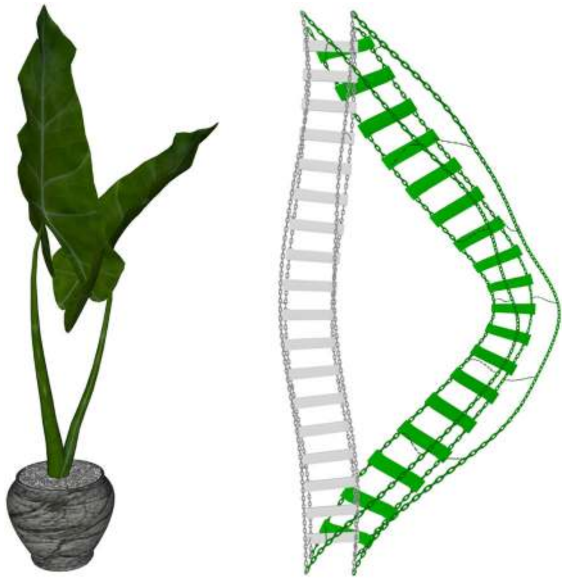

Figure 7. Limitations: Large edits may introduce self-collisions (e.g., for the plant) or noticeable artifacts (e.g., for the bridge).

Our choice of linearization around the input trajectory produce plausible results for small to moderately large edits, meeting most of the needs required to refine existing animations. However, large-enough edits will indubitably lead to visually noticeable artifacts and can even introduce self-collisions (Figure 7). Note also that our method is only guaranteed to be stable if the tangent stiffness matrices of the input sequence are positive definite: negative eigenvalues should be remedied using , for example, [31] or [32]. As future work, it may be interesting to adopt the configuration-invariant linearization scheme [10] or the adaptive dimension reduction [26] instead to make our approach even more robust to large edits. In addition, our method has the same limitation as most of the existing elastic control techniques in that the associated equations of motion should be explicitly provided along with the input animation sequence. Here again, it may be interesting to use learning methods directly from the input sequence.

ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for their valuable comments. We thank Miss Linling Zhou for the creation of the animation for Figures 3, 4, and 6; Mr. Xiaoqiang Zhu for the codes to prepare the volumetric meshes; and Miss Nina Qiu for proofreading. This research was partially supported by NSFC (No. 61210007), National High Technology Research, and Development (863) Program of China (No. 2012AA011503). M. D. was partially supported by NSF grant CCF-1011944. Xiaogang

Jin was supported by Zhejiang Provincial Natural Science Foundation of China (No. Z1110154) and National Natural Science Foundation of China (No. 61272298).

REFERENCES

1. Barbič J, Sin F, Grinspun E. Interactive editing of deformable simulations. *ACM Transactions on Graphics* 2012; **31**(4): 70:1–70:8.
2. Gleicher M. Motion editing with spacetime constraints. In *Symposium on Interactive 3d Graphics*, Providence, Rhode Island, USA, 1997; 139–149.
3. Popovic J, Seitz S, Erdmann M, Popovic Z, Witkin A. Interactive manipulation of rigid body simulations. In *Proceedings of ACM SIGGRAPH*, New York, NY, USA, 2000; 209 –217.
4. Twigg CD, James DL. Many-worlds browsing for control of multibody dynamics. *ACM Transactions on Graphics* 2007; **26**(3): 14:1–14:8.
5. Treuille A, McNamara A, Popović Z, Stam J. Keyframe control of smoke simulations. In *Proceedings of ACM SIGGRAPH*, New York, NY, USA, 2003; 716–723.
6. Kim Y, Machiraju R, Thompson D. Path-based control of smoke simulations. In *Symposium on Computer Animation*, Aire-la-Ville, Switzerland, Switzerland, 2006; 33–42.
7. Nielsen MB, Bridson R. Guide shapes for high resolution naturalistic liquid simulation. *ACM Transactions on Graphics* 2011; **30**(4): 83:1–83:8.
8. Bergou M, Mathur S, Wardetzky M, Grinspun E. Tracks: toward directable thin shells. *ACM Transactions on Graphics* 2007; **26**(3): 50:1–50:10.
9. Barbič J, da Silva M, Popović J. Deformable object animation using reduced optimal control. *ACM Transactions on Graphics* 2009; **28**(3): 53:1–53:9.
10. Huang J, Tong Y, Zhou K, Bao H, Desbrun M. Interactive shape interpolation through controllable dynamic deformation. *IEEE Transactions on Visualization and Computer Graphics* 2011; **17**: 983–992.
11. Hildebrandt K, Schulz C, von Tycowicz C, Polthier K. Interactive spacetime control of deformable objects. *ACM Transactions on Graphics* 2012; **31**(4): 71:1–71:8.
12. Kondo R, Kanai T, Anjyo K. Directable animation of elastic objects. In *Symposium on Computer Animation*, New York, NY, USA, 2005; 127–134.
13. Witkin A, Kass M. Spacetime constraints. In *Proceedings of ACM SIGGRAPH*, New York, NY, USA, 1988; 159–168.
14. Fang AC, Pollard NS. Efficient synthesis of physically valid human motion. *ACM Transactions on Graphics* 2003; **22**(3): 417–426.
15. Safonova A, Hodgins JK, Pollard NS. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *Proceedings of ACM SIGGRAPH*, New York, NY, USA, 2004; 514–521.
16. Jeon H, Choi M-H. Interactive motion control of deformable objects using localized optimal control. In *Proceedings of ICRA*, 2007; 2582–2587.
17. Etzmuss O, Keckeisen M, Strasser W. A fast finite element solution for cloth modelling. In *Proceedings of the Pacific Graphics*, Washington, DC, USA, 2003; 244 –251.
18. Müller M, Gross M. Interactive virtual materials. In *Proceedings of Graphics Interface*, London, Ontario, Canada, 2004; 239–246.
19. Müller M, Dorsey J, McMillan L, Jagnow R, Cutler B. Stable real-time deformations. In *Symposium on Computer Animation*, San Antonio, Texas, 2002; 49–54.
20. Choi MG, Ko H-S. Modal warping: real-time simulation of large rotational deformation and manipulation. *IEEE Transactions on Visualization and Computer Graphics* 2005; **11**(1): 91–101.
21. Barbič J, Popović J. Real-time control of physically based simulations using gentle forces. *ACM Transactions Graphics* 2008; **27**(5): 163:1–163:10.
22. Pentland A, Williams J. Good vibrations: modal dynamics for graphics and animation. *Proceedings of ACM SIGGRAPH* 1989; **23**(3): 207–214.
23. Hauser KK, Shen C, OBrien JF. Interactive deformation using modal analysis with constraints. In *Proceedings of Graphics Interface*, Vol. 3, Halifax, Nova Scotia, 2003; 16–17.
24. Barbič J, James DL. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Transactions on Graphics (SIGGRAPH)* 2005; **24**(3): 982–990.
25. Choi MG, Ko H-S. Modal warping: real-time simulation of large rotational deformation and manipulation. *IEEE Transactions on Visualization and Computer Graphics* 2005; **11**(1): 91–101.
26. Kim T, James DL. Skipping steps in deformable simulation with online model reduction. In *Proceedings of ACM SIGGRAPH Asia*, New York, NY, USA, 2009.
27. Lions JL. Optimal control of systems governed by partial differential equations, 1971.
28. McNamara A, Treuille A, Popović Z, Stam J. Fluid control using the adjoint method. *ACM Transactions on Graphics* 2004; **23**(3): 449–456.
29. Wojtan C, Mucha PJ, Turk G. Keyframe control of complex particle systems using the adjoint method. In *Symposium on Computer Animation*, Aire-la-Ville, Switzerland, Switzerland, 2006; 15–23.

30. Baraff D, Witkin A. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH*, 1998; 43–54.
31. Teran J, Sifakis E, Irving G, Fedkiw R. Robust quasistatic finite elements and flesh simulation. In *Symposium on Computer Animation*, New York, NY, USA, 2005; 181–190.
32. McAdams A, Zhu Y, Selle A, Empey M, Tamstorf R, Teran J, Sifakis E. Efficient elasticity for character skinning with contact and collisions. *ACM Transactions on Graphics* 2011; **30**(4): 37:1–37:12.

AUTHORS' BIOGRAPHIES

Siwang Li is a PhD student of the State Key Lab of CAD & CG, Zhejiang University, China. He received his BSc degree in computer science in 2009 from Northeastern University, China. His recent research focuses on physically based simulation and motion editing.

Jin Huang received the PhD degree from the Computer Science Department, Zhejiang University in 2007 with Excellent Doctoral Dissertation Award of the China Computer Federation (CCF). He is an associate professor in the State Key Laboratory of CAD & CG at Zhejiang University, China. His research interests include geometry processing and physically based simulation. He has served as a reviewer for the ACM SIGGRAPH, the EuroGraphics, the Pacific Graphics, and the TVCG.

Mathieu Desbrun is a Professor at the California Institute of Technology (Caltech). After receiving his PhD from the National Polytechnic Institute of Grenoble (INPG), he spent a year as a post-doctoral researcher at Caltech before joining the Computer Science faculty at the University of Southern California from 2000 to 2004. He now leads the Applied Geometry lab at Caltech, focusing on discrete differential modeling and a wide spectrum of applications. He is also the director of the Computing & Mathematical Science department within the Information Science and Technology initiative at Caltech.

Xiaogang Jin is a professor of the State Key Lab of CAD & CG, Zhejiang University, China. He received his BSc degree in computer science in 1989, and MSc and PhD degrees in applied mathematics in 1992 and 1995, all from Zhejiang University. His current research interests include computer animation, computer graphics, and digital geometry processing.