Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2006), pp. 1–9
M.-P. Cani, J. O'Brien (Editors)

Keyframe Control of Complex Particle Systems Using the Adjoint Method

Chris Wojtan[†1], Peter J. Mucha[‡2] and Greg Turk[§1]

[1]Graphics, Visualization, and Usability Laboratory
Georgia Institute of Technology, Atlanta 30332 USA
[2]Department of Mathematics and Institute for Advanced Materials
University of North Carolina, Chapel Hill 27599 USA

Abstract

Control of physical simulation has become a popular topic in the field of computer graphics. Keyframe control has been applied to simulations of rigid bodies, smoke, liquid, flocks, and finite element-based elastic bodies. In this paper, we create a framework for controlling systems of interacting particles – paying special attention to simulations of cloth and flocking behavior. We introduce a novel integrator-swapping approximation in order to apply the adjoint method to linearized implicit schemes appropriate for cloth simulation. This allows the control of cloth while avoiding computationally infeasible derivative calculations. Meanwhile, flocking control using the adjoint method is significantly more efficient than currently-used methods for constraining group behaviors, allowing the controlled simulation of greater numbers of agents in fewer optimization iterations.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Physically based modeling; I.3.7 [Computer Graphics]: Animation; I.6.8 [Simulation and Modeling]: Animation;

1. Introduction

Recent advances in numerical methods and physical models have led to an explosion in contributions to the physics-based animation literature. Each year we see more detailed fluid simulations, increasingly robust collision detection and response algorithms, and progressively more powerful deformable models. Given the pervasiveness of physics-based animation, the community has sought to control these inherently self-governing systems, leading to reliable methods for forcing fluids, rigid bodies, flocks, and elastic objects to conform to predetermined configurations. Our work extends previous optimization methods for animation control to accommodate cloth simulation, flocking models, and other particle systems.

There are several instances in which controlled particle systems are desirable. Motion pictures like *Lord of the Rings* and *Star Wars: Episode 1* simulated huge groups of creatures acting in concert using flocking models [Dun02, TO99]. The control of these flocking systems is very important for special effects directors, and the issue has been examined by several researchers already [AMC03, LCF05].

Control of particle system models would also be valuable for a computer-generated transition between two live-action video clips with clothed actors. In the "Burly Brawl" sequence in *The Matrix Reloaded*, Neo battles several agents at once while wearing a long, flowing coat [For03]. Imagine if the director called for a seamless transition from a computer-generated martial arts move into live footage. The computer-generated portion of this special-effects shot would be nearly impossible for a passively-animated cloth model, because the task has now changed from an initial value problem to a boundary value problem. Instead of blindly evolving the cloth, it must pass through a keyframed configuration at the end, defined by the starting state in the live clip. Our system for cloth control can make effects like this less taxing.

Control of a physics-based animation can be generally thought of as an optimization problem. The animator seeks

† wojtan@cc.gatech.edu
‡ mucha@math.unc.edu
§ turk@cc.gatech.edu

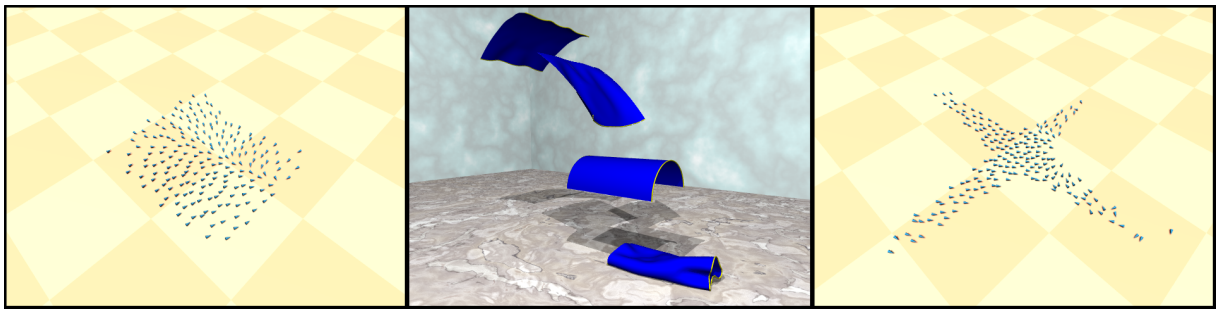

Figure 1: *Our method allows flocking and cloth simulations to form key-framed shapes. Here, a flock forms into a square and a cross shape, while a sheet of cloth bends into a perfect arch.*

to find the optimal balance between visual plausibility and controlled behavior. In order to impose constraints on these passive animations, the system allows some input control parameters to alter the original physical behavior. Because arbitrary changes can cause disturbingly unphysical results, we seek to minimize the effect of these controls while simultaneously attempting to satisfy any constraints on the animation. To find this balance, we create an objective function that evaluates a set of controls by penalizing it whenever it alters the animation, and by rewarding it if the resulting animation meets the animator's constraints. The set of controls that yields the optimal objective function will produce the physical behavior that the animator desires.

We control physical simulations through gradient-based optimization via the adjoint method [MTPS04], though its application here is far from straightforward. Naively applying the adjoint method to certain numerical integration schemes proves to be computationally infeasible — in order to use such an optimization-based framework on the cloth models described by, for example, Baraff and Witkin [BW98] or Choi and Ko [CK02], we must supply an alternative method for efficiently calculating the adjoint states.

The main contributions of our paper include:

Efficient and Robust Flocking Control. We show how to use standard adjoint calculations with gradient-based optimization to control a flocking simulation. Our method converges in fewer iterations than previous sampling approaches while offering more controllability than motion graphs and hand-tweaking. We demonstrate results of keyframed flocking with several hundred agents.

Novel Extension of the Adjoint Method. As it stands, the exact adjoint method is ill-suited for efficiently controlling particle simulations with some numerical integrators, specifically with linearized implicit schemes, because of a dimensional explosion. We derive and present a valuable approximation that allows for the control of more complicated physical simulations like the implicit integration of cloth. Specifically, we use a linearized implicit integrator for the physics calculations, and we use a *fully* implicit integrator for the adjoint calculations.

Control of Cloth Animations. We present methods for adding keyframe control to cloth animations by way of approximate adjoint calculations, demonstrating with a diverse set of examples. We apply center-of-mass constraints as well as simpler per-particle constraints. To our knowledge, controlling the dynamic movements of cloth in such a manner has not been done in the past.

2. Previous Work

2.1. Modeling Flocking and Cloth

Particle systems permeate research in computer graphics. The work by Reynolds [Rey87] is of particular relevance to our research. He used a system of particles with pair-wise interactions to create models of flocking "boids."

Particle systems with pair-wise interactions have also played a central role in the animation of deformable models like cloth. Provot [Pro95] increased the stability of cloth models by enforcing deformation constraints. Baraff and Witkin [BW98], Volino and Magnenat-Thalmann [VMT00], and Desbrun *et al.* [DSB99] each simulated cloth with large time steps using implicit integrators. Choi and Ko [CK02] additionally allowed livelier animations with a more stable buckling model involving nonlinear springs that resist compression and bending.

Bridson *et al.* [BFA02] created a robust system for handling collisions in cloth animations, utilizing both exact intersection tests and repulsion forces. Bridson *et al.* [BMF03] introduced several techniques for improving clothing simulations with wrinkles and folds, while Baraff *et al.* [BWK03] showed a method for a cloth system to repair itself after it was otherwise hopelessly tangled. Govindaraju *et al.* [GKJ*05] greatly increased the efficiency of collision detection in a cloth system by using chromatic decomposition to compare the minimal number of independent sets, and used 2.5-dimensional tests on a graphics processor for an even more dramatic speedup.

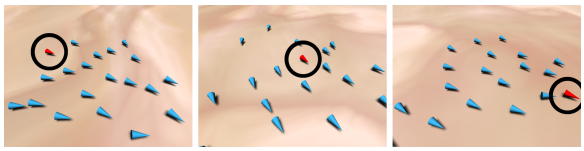

Figure 2: *The circled agent is constrained in the back of the pack at the start, and at the front of the pack at the end of this animation.*

2.2. Control of Simulated Animation

Ever since simulation-based animation was introduced to the computer graphics community, animators have been interested in controlling its behavior. Witkin and Kass [WK88] solved for physically valid motions given some constraints. Barzel *et al.* [BHW96] discussed control in terms of aiming for visually plausible results, and presented controlled animations of billiard balls. Chenney and Forsyth [CF00], Popović *et al.* [PSE*00], and Popović *et al.* [PSE03] focused on controlling animations of rigid bodies.

Recently, a few papers have focused on controlling fluid simulations. Treuille *et al.* [TMPS03] optimized over an objective function to balance between keyframe fidelity and plausible fluid motion. McNamara *et al.* [MTPS04] greatly increased the speed of the optimization using the adjoint method, while also addressing the control of simulations with liquid interfaces. These two papers were an inspiration and a point of departure for our work. Fattal and Lischinski [FL04] proposed a different style of smoke control that is much faster but less precise than an optimization framework. Shi *et al.* [SY05a, SY05b] controlled fluids with guiding shapes and moving targets, and combined motion synthesis with smoothed-particle hydrodynamics in [SYWC05].

A few works have previously addressed control of flocking simulations. Anderson *et al.* [AMC03] used iterative sampling to create flock animations performing under user-defined constraints. Their method is computationally intensive, but produces excellent results. Yu-Chi *et al.* [LCF05] produced controlled group behaviors much more quickly by building group motion graphs. Their technique relies on a comprehensive collection of scripted sample animations that are subsequently linked to produce controlled flocks.

Control of deformable bodies has also been addressed. Kondo *et al.* [KKiA05] enforces trajectory constraints on a finite element-based elastic body and adapts the stiffness matrix in order to match key poses. Cutler *et al.* [CGW*05] concentrate on controlling wrinkles in computer-generated cloth. Their system focuses on controlling configurations of wrinkles in cloth, while we focus on dynamic movements. Bhat *et al.* [BTH*03] used simulated annealing to estimate cloth simulation parameters. They optimized the properties of simulated cloth in order to make it behave like real fabric.

3. Flocking

We use a particle system similar to that of Reynolds [Rey87] for animating group behaviors. We treat each agent in the flock as a particle, and each agent's behavior is determined by a series of forces. These forces are:

Collision Avoidance Force. If one agent comes too close to another, we apply a repulsive force to push them apart.

Velocity Matching Force. We apply a force to make an agent's velocity match that of his neighbors, essentially damping relative velocities between flock members.

Flock Centering Force. Each flock member feels a force pushing him toward his neighbors, implemented as an attractive force toward the center of mass of his nearby neighbors.

Wander Force. We avoid artificial-looking animations by applying a random force at each time step in order to mimic agents having "a mind of their own."

Each particle in the system has a mass m, position $\vec{x}$, and velocity $\vec{v}$ defined at each time step t_n. We collect these quantities into a single position vector $\mathbf{x}$ and velocity vector $\mathbf{v}$ that each vary in time, and a constant mass matrix $\mathbf{M}$. To integrate the particle motions in time, we use an explicit method:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \Delta t \mathbf{M}^{-1}\mathbf{f}_n\,, \quad \mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \mathbf{v}_{n+1}\,, \qquad (1)$$

where $\mathbf{f}_n \equiv \mathbf{f}(\mathbf{x}_n, \mathbf{v}_n)$ is the force on each particle evaluated using current positions and velocities, summing all collision avoidance, velocity matching, flock centering, and wander forces. We note (1) is second-order accurate for position-dependent forces (equivalent to leapfrog after a notational change), but reduces to first-order when velocity-dependent forces are included as here. For the examples in this paper, we used linear collision avoidance, velocity matching, and flock centering forces, though nothing in our method stops us from using more complicated flocking dynamics.

4. Controlled Flocking

Our goal is to find a set of controls that satisfies the animator's constraints while minimizing non-physical behavior. In general, we can find this set of controls, $\mathbf{u}$, for a physical system with a state vector $\mathbf{q}$, a matrix C that converts $\mathbf{u}$ into a meaningful operation on the state, and a set of functions that advances $\mathbf{q}$ through time. For a particle system, the state vector $\mathbf{q}_n = <\mathbf{x}_n, \mathbf{v}_n>$ at frame n consists of the current positions and the current velocities of each particle at that time step. In our implementation, the control vector $\mathbf{u}$ contains time-dependent external forces that act on each particle (one force per particle per time step in all of our flocking animations) and C_n maps these control forces to the particles each frame. We thus seek to minimize the objective function

$$\phi(\mathbf{u}, \mathbf{Q}) = \frac{1}{2}\sum_{n=0}^{N}\left(||W_n(\mathbf{q}_n - \mathbf{q}_n^*)||^2 + \alpha||C_n\mathbf{u}||^2\right) \qquad (2)$$

where W_n is a weight matrix that the animator can use to emphasize the importance of matching the velocity or position at certain keyframes, $\mathbf{q}_n^*$ is a keyframe at frame n, and α is a scalar *smoothness* term that indicates the relative importance of minimizing the effect of external controls. The notation $\mathbf{Q}$ indicates the full set of $\mathbf{q}$ states across all times that are included in the summation. That is, we seek the set of control forces $\mathbf{u}$ that yields the minimal ϕ, simultaneously minimizing the keyframe deviation and the use of external controls in a least-squares sense. In a typical application, most keyframes will have a weighting W_n of zero, and only a few frames will have non-zero weights.

We also consider generalizations of (2). For example, in figure 1, 200 agents form a box. To obtain this result, we replaced the squared distance to a specific keyframe state with the sum of squared distances from the box over agents outside the target box. The agents created an evenly-spaced formation within the goal shape because the flocking simulation prefers that agents do not come too close to one another. The *smoothness* term in the objective function penalizes any excessive control force that may violate this preference.

4.1. Review of the Adjoint Method

The *adjoint method*, excellently summarized by McNamara *et al*. [MTPS04], efficiently calculates the gradient of ϕ,

$$\frac{d\phi}{d\mathbf{u}} = \frac{\partial\phi}{\partial\mathbf{Q}}\frac{d\mathbf{Q}}{d\mathbf{u}} + \frac{\partial\phi}{\partial\mathbf{u}} \tag{3}$$

for use as part of a minimization routine. The gradient as written in (3) is excessively expensive to calculate, because it requires computation of the full $d\mathbf{Q}/d\mathbf{u}$ matrix of derivatives of all state variables with respect to all possible controls, subject to the condition that the state variables $\mathbf{Q}$ obey the imposed simulation conditions. In the notation used by McNamara *et al*. [MTPS04], the simulation requires $\mathbf{Q} = \mathbf{F}(\mathbf{Q},\mathbf{u})$, where $\mathbf{F}$ encapsulates all of the time-step formulae between the $\mathbf{Q}$ values at different times. Alternatively, differentiation of this constraint with respect to $\mathbf{u}$ gives

$$\frac{d\mathbf{Q}}{d\mathbf{u}} = \frac{\partial\mathbf{F}}{\partial\mathbf{Q}}\frac{d\mathbf{Q}}{d\mathbf{u}} + \frac{\partial\mathbf{F}}{\partial\mathbf{u}} \quad \text{or} \quad \left(\mathbf{I} - \frac{\partial\mathbf{F}}{\partial\mathbf{Q}}\right)\frac{d\mathbf{Q}}{d\mathbf{u}} = \frac{\partial\mathbf{F}}{\partial\mathbf{u}}. \tag{4}$$

The crux of the adjoint method for rapid evaluation of the gradient comes from replacing the vector-matrix product containing $d\mathbf{Q}/d\mathbf{u}$ in (3) with an equivalent product in terms of the more-easily calculated adjoint of $\mathbf{Q}$. That is, while we seek a specific vector-matrix product $\mathbf{g}^T\mathbf{B}$, subject to a matrix-matrix product constraint $\mathbf{AB} = \mathbf{C}$, we can replace this vector-matrix product with $\mathbf{s}^T\mathbf{C}$ such that $\mathbf{A}^T\mathbf{s} = \mathbf{g}$. This can be easily verified by direct substitution: $\mathbf{s}^T\mathbf{C} = \mathbf{s}^T\mathbf{AB} = (\mathbf{A}^T\mathbf{s})^T\mathbf{B} = \mathbf{g}^T\mathbf{B}$. This replacement is advantageous in many control settings, including the present case, where the matrix $\mathbf{B} = d\mathbf{Q}/d\mathbf{u}$ is costly to compute while the alternative calculation includes sparse $\mathbf{A} = (\mathbf{I} - \partial\mathbf{F}/\partial\mathbf{Q})$ and $\mathbf{C} = \partial\mathbf{F}/\partial\mathbf{u}$:

$$\frac{d\phi}{d\mathbf{u}} = \hat{\mathbf{Q}}^T\frac{\partial\mathbf{F}}{\partial\mathbf{u}} + \frac{\partial\phi}{\partial\mathbf{u}}, \quad \hat{\mathbf{Q}} = \left(\frac{\partial\mathbf{F}}{\partial\mathbf{Q}}\right)^T\hat{\mathbf{Q}} + \left(\frac{\partial\phi}{\partial\mathbf{Q}}\right)^T. \tag{5}$$

The simulation specifics encoded in $\partial\mathbf{F}/\partial\mathbf{Q}$ therefore define the relations for calculating the adjoint state variables $\hat{\mathbf{Q}}$. If $\mathbf{F}$ represents a multistep integration scheme for simulating the physical states, then the constraint on $\hat{\mathbf{Q}}$ in (5) typically reduces to a similar multistep scheme for the adjoints going backwards in time. The adjoint method thus proceeds by first computing an entire simulation with a given control vector $\mathbf{u}$, saving the state $\mathbf{q}$ at each time and calculating the objective function ϕ. The gradient $d\phi/d\mathbf{u}$ is then found by running backwards through a series of operations based on (5) to calculate the $\hat{\mathbf{Q}}$ adjoint states and $d\phi/d\mathbf{u}$ gradient.

The adjoint operations correspond to the functions that advance our system through time. They consist of: (i) the adjoint of the function that maps the control vector $\mathbf{u}$ to the state $\mathbf{q}$, (ii) the adjoint of our objective function (equation 2) and (iii) the adjoint of the functions that advance our physical simulation. In our system, the adjoint functions (i) and (ii) are the same as those of McNamara *et al*. [MTPS04]. However, our adjoint functions (iii) are very different, as we simulate particle systems with pair-wise interactions (solving ordinary differential equations) where they simulated fluid (solving partial differential equations). Additionally, we will see in the section below that implicit schemes for simulating cloth introduce additional issues. Having established our forward and backward calculations, the objective function and its gradient are used in a gradient-based minimization routine (e.g., [ZBLN95]) for the controls, $\mathbf{u}$.

4.2. Adjoint Particle System Dynamics

Now that we have explained the need for a set of adjoint operations corresponding to the functions that advance our physical simulation, let us discuss the specifics of our particle system dynamics. Temporarily focusing our attention on forward Euler integration of the differential equation $\mathbf{q}' = \mathbf{V}(\mathbf{q})$, $\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta t\mathbf{V}(\mathbf{q}_n)$, the right equation in (5) reduces to a simple rule for the evolution of the adjoint states:

$$\hat{\mathbf{q}}_n = \hat{\mathbf{q}}_{n+1} + \Delta t\left(\left.\frac{\partial\mathbf{V}}{\partial\mathbf{q}}\right|_n\right)^T\hat{\mathbf{q}}_{n+1} + \left(\frac{\partial\phi}{\partial\mathbf{q}_n}\right)^T \tag{6}$$

where $\partial\mathbf{V}/\partial\mathbf{q}|_n$ signifies the derivative $\partial\mathbf{V}/\partial\mathbf{q}$ evaluated at time step n. Therefore, once the full set of physical states $\mathbf{q}$ have been calculated for a given set of control forces, the adjoint states $\hat{\mathbf{q}}$ can be calculated by iterating backward in time according to the above formula.

Because the scheme presented in (1) can be similarly rewritten as a one-step explicit form $\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta t\mathbf{V}(\mathbf{q}_n)$, albeit for a different $\mathbf{V}$ which includes additional Δt factors, the derivation of the rule for evolving the adjoint states similarly reduces to proper use of $\partial\mathbf{V}/\partial\mathbf{q}|_n$ quantities. These quantities can be identified from the differentials of (1):

$$\begin{aligned} d\mathbf{v}_{n+1} &= d\mathbf{v}_n + \Delta t\mathbf{M}^{-1}\left(\left.\frac{\partial\mathbf{f}}{\partial\mathbf{v}}\right|_n d\mathbf{v}_n + \left.\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right|_n d\mathbf{x}_n\right) \\ d\mathbf{x}_{n+1} &= d\mathbf{x}_n + \Delta t d\mathbf{v}_{n+1} \end{aligned} \tag{7}$$

where, e.g., $\partial \mathbf{f}/\partial \mathbf{v}|_n$ indicates the Jacobian evaluated with $\mathbf{x}_n$ and $\mathbf{v}_n$ values at time t_n. Substitutions of the above representations of the partial derivatives then lead to

$$\begin{aligned}\hat{\mathbf{v}}_n &= \left(\mathbf{I} + \Delta t \mathbf{M}^{-1} \left.\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right|_n\right)^T (\hat{\mathbf{v}}_{n+1} + \Delta t \hat{\mathbf{x}}_{n+1}) + \left(\frac{\partial \phi}{\partial \mathbf{v}_n}\right)^T \\ \hat{\mathbf{x}}_n &= \hat{\mathbf{x}}_{n+1} + \Delta t \mathbf{M}^{-1} \left(\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_n\right)^T (\hat{\mathbf{v}}_{n+1} + \Delta t \hat{\mathbf{x}}_{n+1}) + \left(\frac{\partial \phi}{\partial \mathbf{x}_n}\right)^T\end{aligned} \tag{8}$$

We emphasize that the above equations explicitly give the $\hat{\mathbf{q}}_n$ adjoint states in terms of those at time t_{n+1}, $\hat{\mathbf{q}}_{n+1}$, because the partial derivatives at time t_n are in terms of physical states already calculated by the corresponding forward simulation.

Using these functions to update the adjoints $\hat{\mathbf{q}}$ and calculate the $d\phi/d\mathbf{u}$ gradient, we can control our particle system.

4.3. Adjoint Flocking

We use the flocking forces described in section 3 as our $\mathbf{f}$ forces in equation 8. The collision avoidance and flock centering forces are purely position-dependent, so their $\partial \mathbf{f}/\partial \mathbf{v}$ contribution is zero, while their $\partial \mathbf{f}/\partial \mathbf{x}$ has symmetric nonzero entries whenever two particles exert one of these forces on each other. Similarly, the velocity matching force is purely velocity-dependent, so its $\partial \mathbf{f}/\partial \mathbf{x}$ Jacobian is zero, while $\partial \mathbf{f}/\partial \mathbf{v}$ will be symmetric. The wander force is neither dependent on $\mathbf{x}$ nor $\mathbf{v}$; indeed, in order to make the objective function consistent from one iteration to the next, we precompute and store all of the random wander contributions for the full simulation. That is, while each particle feels these random forces throughout the simulation, each wander force is consistent across different simulation iterations.

In figure 3, we show a flock of 100 agents of equal mass following a pre-determined path, obtained by placing special center-of-mass key frames along the curve and generalizing the distance term in (2) to $||W_n(\mathbf{q}_n^{cm} - \mathbf{q}_n^{cm*})||^2$, where $\mathbf{q}_n^{cm*}$ and $\mathbf{q}_n^{cm}$ are the desired and computed centers of mass at time step n, and W_n is the weight of this key frame. The adjoint of this generalized objective involves incrementing every element of the adjoint state by the derivative with respect to $\mathbf{q}$ — in this case, $W_n(\mathbf{q}_n^{cm} - \mathbf{q}_n^{cm*})/p$ for p particles in the flock (from $q^{cm} = \sum_{i=1}^{p} \mathbf{q}_i/p$).

5. Cloth

For our cloth simulations, we use the physical model described in [CK02], selected in part because of its good handling of physical buckling, though our work is not restricted to this model. Because simple explicit integrators are unstable for stiff springs, cloth animators typically use either a more complicated explicit scheme or an implicit scheme. Explicit schemes, including explicit Runge-Kutta or Newmark integrators, require small time steps, necessitating a huge amount of memory for all the saved states in a control setting. The derivation of the adjoint operations for these higher-order integrators are more complicated than for forward Euler, but are nevertheless straightforward and

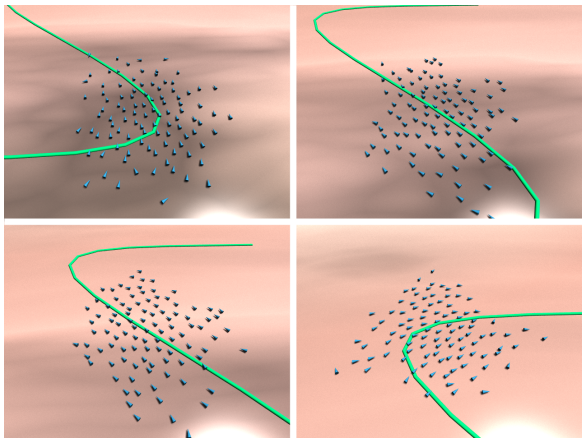

Figure 3: *We use center-of-mass key frames to constrain this flock along a path.*

standard. In contrast, naive implementation of the adjoint method for control of implicitly-integrated cloth introduces significant complications. To illustrate this, we consider the backward Euler scheme:

$$\begin{aligned}\mathbf{v}_{n+1} &= \mathbf{v}_n + \Delta t \mathbf{M}^{-1} \mathbf{f}_{n+1} \\ \mathbf{x}_{n+1} &= \mathbf{x}_n + \Delta t \mathbf{v}_{n+1}\end{aligned} \tag{9}$$

where $\mathbf{f}_{n+1}$ is the spring force evaluated at the *next* time step. This scheme is implicit because $\mathbf{f}_{n+1}$ is unknown at the moment of calculating the positions and velocities at time t_{n+1}. While we thus face generically nonlinear sets of equations, Baraff and Witkin [BW98] showed that these can be simplified in cloth simulations by use of the first-order Taylor series approximation:

$$\mathbf{f}_{n+1} \approx \mathbf{f}_n + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_n \Delta \mathbf{x} + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right|_n \Delta \mathbf{v} \tag{10}$$

where $\Delta \mathbf{x} = (\mathbf{x}_{n+1} - \mathbf{x}_n)$, and $\Delta \mathbf{v} = (\mathbf{v}_{n+1} - \mathbf{v}_n)$ (the same approximation is used by Choi and Ko [CK02] to linearize the BDF2 scheme). Replacing $\mathbf{f}_{n+1}$ with its first-order Taylor polynomial, equation 9 then gives

$$\begin{aligned}\Delta \mathbf{v} &= \Delta t \mathbf{M}^{-1} \left(\mathbf{f}_n + \tfrac{\partial \mathbf{f}}{\partial \mathbf{x}}|_n \Delta \mathbf{x} + \tfrac{\partial \mathbf{f}}{\partial \mathbf{v}}|_n \Delta \mathbf{v}\right) \\ \Delta \mathbf{x} &= \Delta t (\mathbf{v}_n + \Delta \mathbf{v})\end{aligned} \tag{11}$$

which can be rearranged as a linear system $\mathbf{A} \cdot \Delta \mathbf{v} = \mathbf{b}$, with

$$\mathbf{A} = \mathbf{M} - \Delta t \left.\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right|_n - \Delta t^2 \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_n, \quad \mathbf{b} = \Delta t (\mathbf{f}_n + \Delta t \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_n \mathbf{v}_n) \tag{12}$$

(noting this $\mathbf{A}$ is different from the general notation in Section 4.1). We solve this system by a modified conjugate gradient technique [BW98], update the particle positions and velocities, and repeat this process for every time increment.

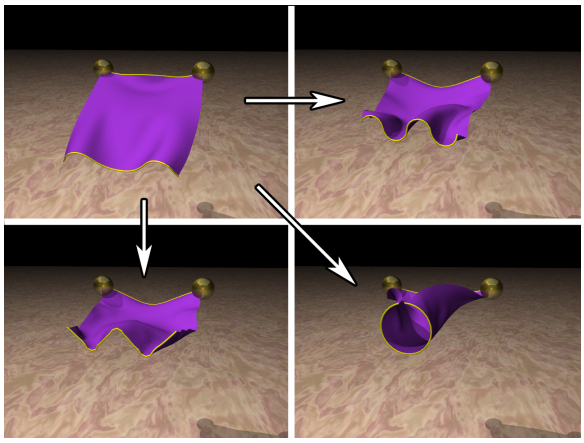

Figure 4: *We control these animations to create shapes with the cloth trimming.*

6. Cloth Control

Our flocking simulations use explicit integration with a straightforward explicit evolution of the associated adjoint state from time t_{n+1} to t_n. In contrast, the cloth simulations described above use an implicit, linearized backward Euler scheme (11). We would like to create adjoint states corresponding to this system, but, as we demonstrate below, this becomes excessively complicated computationally. We thus choose to develop an approximate system for evolving the adjoint states in order to efficiently calculate the adjoint states and resulting gradients.

One can naively derive the adjoint operations for our linearized implicit mass-spring system by following the same steps used to reach equation 8. However, such calculation leads immediately to derivatives of the terms in equation 11, including derivatives of the $\partial\mathbf{f}/\partial\mathbf{x}$ and $\partial\mathbf{f}/\partial\mathbf{v}$ Jacobians, reducing to second derivatives of the $\mathbf{f}$ forces with respect to both positions and velocities. The $\mathbf{f}$ forces have $3p$ components at each time step for a three-dimensional system of p particles, and each Jacobian (with respect to $\mathbf{x}$ and $\mathbf{v}$, respectively) is a $3p \times 3p$ matrix. Taking additional derivatives yields a trio of $3p \times 3p \times 3p$ tensors — a dimensional explosion. Meanwhile, where our explicit time stepping for flocking yielded an explicit evolution for the adjoint states, the implicit integration of cloth similarly leads to implicit adjoint calculations. Even given the relative sparseness of the second-derivative tensors, the required linear solves for this adjoint evolution are significantly more expensive than those in the forward simulation, so even ignoring the resultingly cumbersome bookkeeping, this does not appear to yield a computationally feasible control strategy.

Our solution to this problem explores the distinction between full backward Euler integration (9) and the linearized scheme (11) used for efficient cloth simulation, taking advantage of a specific feature of the adjoints of implicit systems. That is, since the adjoint method involves computing an entire simulation first, we already know every physical state of the forward simulation when calculating the adjoint states. The adjoint evolution corresponding to the generically nonlinear backward Euler integration of the physical states then reduces to a linearly implicit calculation for the adjoints, as we show below. We will therefore avoid the dimensional explosion of calculating adjoints corresponding to (11) by instead approximating these adjoint states by those corresponding to (9).

The adjoint evolution corresponding to the backward Euler scheme, $\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta t\mathbf{V}(\mathbf{q}_{n+1})$, is found by substitution into equation 5:

$$\hat{\mathbf{q}}_n = \hat{\mathbf{q}}_{n+1} + \Delta t \left(\frac{\partial \mathbf{V}}{\partial \mathbf{q}}\bigg|_n \right)^T \hat{\mathbf{q}}_n + \left(\frac{\partial \phi}{\partial \mathbf{q}_n} \right)^T \tag{13}$$

Where the backward Euler evolution of the physical states generally involves solving nonlinear implicit systems (motivating the linearization), the corresponding adjoint evolution is linearly implicit because the $\partial\mathbf{V}/\partial\mathbf{q}|_n$ Jacobian is already known from the forward simulation of the physical state. Applying the backward Euler particle-system evolution (9), equation 13 becomes

$$\begin{aligned} \hat{\mathbf{v}}_n &= \hat{\mathbf{v}}_{n+1} + \Delta t\mathbf{M}^{-1} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\bigg|_n \right)^T \hat{\mathbf{v}}_n + \Delta t\hat{\mathbf{x}}_n + \left(\frac{\partial \phi}{\partial \mathbf{v}_n} \right)^T, \\ \hat{\mathbf{x}}_n &= \hat{\mathbf{x}}_{n+1} + \Delta t\mathbf{M}^{-1} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\bigg|_n \right)^T \hat{\mathbf{v}}_n + \left(\frac{\partial \phi}{\partial \mathbf{x}_n} \right)^T \end{aligned} \tag{14}$$

Notationally absorbing $\partial\phi/\partial\mathbf{q}_n$ terms into the $\hat{\mathbf{q}}_{n+1}$ in the above equation, substitution and rearrangement gives $\mathbf{A}^T\hat{\mathbf{v}}_n = \mathbf{c}$, where

$$\mathbf{A} = \mathbf{M} - \Delta t \left. \frac{\partial \mathbf{f}}{\partial \mathbf{v}} \right|_n - \Delta t^2 \left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right|_n, \quad \mathbf{c} = \mathbf{M}\left(\hat{\mathbf{v}}_{n+1} + \Delta t\hat{\mathbf{x}}_{n+1}\right) \tag{15}$$

Solution of this linear system gives $\hat{\mathbf{v}}_n$, then yielding $\hat{\mathbf{x}}_n$ explicitly from (14). Because the matrices $\mathbf{M}$, $\frac{\partial \mathbf{f}}{\partial \mathbf{v}}|_n$, and $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}|_n$ are all symmetric, this $\mathbf{A}$ is, conveniently, exactly the same matrix as in the linear system for the forward simulation (12). Therefore, very little additional implementation work needs to be done, and this adjoint computation is just as fast as the forward simulation. We note that we approximate the implicit forward integrator with another implicit integrator (as opposed to an explicit integrator), because the adjoint equations are stiff.

We emphasize that our cloth control simulations are not as would be obtained by use of automatic adjoint-code generation based on automatic differentiation (e.g., [GK98]). Automatic adjoint generation from the linearized backward Euler time-stepping would generate the dimensional explosion described above. Rather, we have avoided such expensive derivative calculations by evaluating approximations to the adjoint states, in this case in terms of a different time step scheme. While the general notion of using adjoint states that

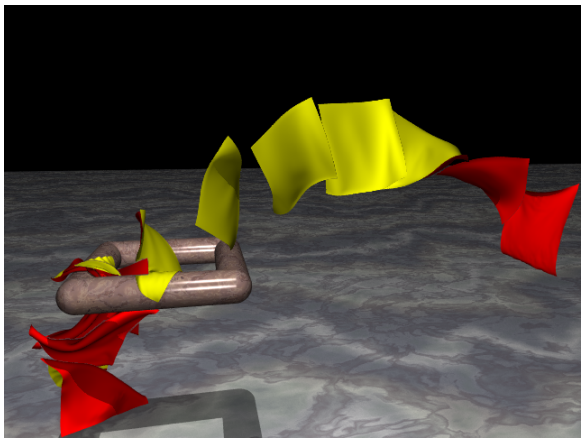

Figure 5: *We force a sheet of cloth to land in a small ring using key-framed control. Our system only optimized over the initial conditions in this example; no additional control forces were applied.*

only approximately correspond to the physical state variables has been used successfully in the meteorological community for data assimilation [SW95] and sensitivity studies [ST97], we know of no previous application of such integrator-swapping to turn what would be a seemingly computationally intractable problem into an approximate calculation that runs at the same cost as the forward simulation.

We note that our methodology does not require any specific restrictions on the types of forces in the system, only that it is possible to compute $\partial\mathbf{f}/\partial\mathbf{x}$ and $\partial\mathbf{f}/\partial\mathbf{v}$. This idea of using the adjoints corresponding to a slightly different integration scheme can then be extended to other systems. Meanwhile, it is important to recognize that this is an approximation, because better optimization convergence can sometimes be obtained by improving the accuracy of the adjoints and the resulting gradient. Our substitution of the linearized backward Euler scheme with the adjoints corresponding to the full backward Euler integrator introduces $O(\Delta t^3)$ errors; each integrator is only first-order accurate in time, but they are the same as each other through second order. We note that the linearized BDF2 scheme of [CK02] could be similarly approximated by adjoints derived for full BDF2 steps, with the same $O(\Delta t^3)$ accuracy. Taking advantage of the improved accuracy afforded by smaller time steps, we have in some instances had good success by first optimizing over simulations with large time steps and using these results to generate the initial guess of a rapid but more detailed optimization over smaller time step simulations.

6.1. Cloth Collisions

In order to model collision response in our cloth, we apply stiff repulsive springs between the two colliding bodies. This is the same method that Choi and Ko [CK02] use to prevent self-collisions. We use the Jacobian of this spring force in the adjoint calculation in order to add some predictive ability to our optimizer. Collision response adds discontinuities into our objective function, so our gradient-based optimization may have trouble with collision-heavy animations. Even so, we have successfully controlled cloth in the presence of multiple collisions. In figure 6, for example, the cape collides with the character and with itself hundreds of times during the optimization iterations.

7. Results

We have produced a series of examples illustrating control in both flocking and cloth simulations. As mentioned earlier, we have created flocks that are constrained to a path or a series of shapes. In figure 2, we show a small flock racing towards a goal. The highlighted character progresses from the back of the pack to the front, while the group still adheres to the flocking rules. This outcome was achieved by constraining the entire flock to a straight path with center-of-mass key frames, while the highlighted agent was constrained to be in the front of the flock at the end of the animation with an individual particle key frame. The adjoint control of flocks seems to be very effective, and we have tested it with as many as one thousand agents in a flock. Each flocking example took less than 250 iterations of our flocking simulator. In our most expensive example, our method took 3.5 hours for 200 agents to match two shapes on the ground (figure 1). We believe our timings could be much better had we implemented spatial data structures for such large flocks, in order to avoid the quadratic scaling of particle interactions.

We have also created several examples of key-framed cloth control. In figure 5, we optimized over only the initial conditions of a cloth animation (one velocity per particle at the start of the animation) in order to stylistically toss a piece of cloth through a small ring. We use a center-of-mass constraint to force the cloth to travel through the ring, while we enforce additional position constraints throughout the simulation in order to make the cloth flip and spin in desired ways. In all other cloth examples, our control vector consisted of one force per particle per time step in each cloth animation. We show pieces of cloth swinging into user-defined shapes in figure 4, and we show a piece of cloth matching the shape of an arch in figure 1.

In figure 6, we show three cloth animations of a character wearing a cape. In the left animation, a “stunt double” falls from a height, and the resulting cloth motion is rather chaotic. The right animation shows the cape of an actor walking normally. We stitch these two animations together with our cloth-control technique, and the resulting animation is shown in the center column with the synthesized blend highlighted in orange. The left animation is treated as the initial condition, and the right animation is treated as the goal state (our system matches both the desired positions and ve-

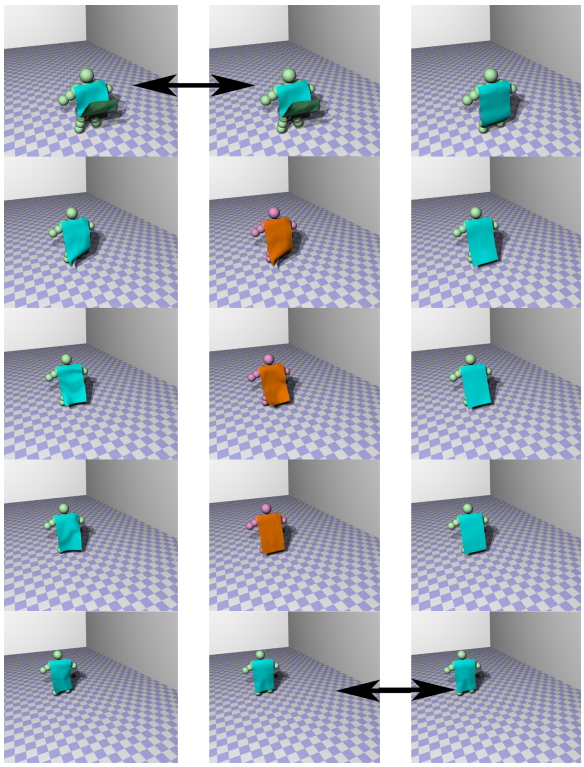

Figure 6: *The left and right columns of this image show two existing animations of a character wearing a cape. The middle column shows an animation that begins with the same conditions as the left animation and ends with the same conditions as the right animation. The cloth in the center is a synthesized animation that blends between the two states using our technique for controlling cloth animations.*

locities for every particle in the simulation). This boundary value problem in figure 6 was our most expensive example, taking almost two days to converge. The rest of the cloth examples ran for a few hours on average, though the optimizer found an acceptable solution very early before it converged to the optimal solution. The resolutions of our cloth ranged from 16×26 (figure 6) to 50×50 (figure 5), though our method will work with much higher resolutions.

8. Discussion and Future Work

We have presented methods for controlling particle systems that are governed by ordinary differential equations, with applications of this control to both flocking and cloth simulations, and have produced several animations to illustrate our method. As part of this effort, we have explored a novel extension to the adjoint method allowing for efficient calculation of approximate adjoint states for the linearized implicit integrators commonly used for animation of cloth and other stiff systems.

In section 6, we introduced an approximation of the adjoint state by swapping a backward Euler scheme with a linearized backward Euler scheme. The error induced, locally $O(\Delta t^3)$, was only problematic in very complicated control problems, such as the boundary value problem in figure 6, where every particle in the simulation must exactly match a given position and velocity. The optimizer found a very reasonable answer, but did not hit the desired key frame perfectly. In order to find a more accurate solution, we used the result from this imperfect optimization as a starting point in a new simulation with smaller time steps.

There is still future work to be done with the cloth control problem. For example, collision detection and response are inherently discontinuous operations. Because we use a gradient-based optimization technique to find the optimal set of control forces in our animations, collisions present a problem. Nevertheless, we have noticed that the optimization software [ZBLN95] is surprisingly resilient, and has managed to converge to the correct solution even in the presence of several collisions. In addition, almost all of our flocking forces have sharp discontinuities in them beyond a certain distance, but they did not pose much of a problem for our optimizer. Perhaps this is to be expected, as McNamara *et al*. [MTPS04] were able to control level sets even though their heapsort operation was highly discontinuous. In order to handle collisions more robustly, we would like to investigate the use of a random search [PSE*00], or smoothing out the collision operations in early optimization stages in order to avoid local minima. Robust cloth control in the presence of a large number of collisions is considered future work.

9. Acknowledgements

We would like to thank Elaine M. Huang for recording the voice-over in our video. We would also like to thank Adrien Treuille for helping us understand the adjoint method. Chris Wojtan was supported by an NSF Graduate Fellowship.

References

[AMC03] ANDERSON M., MCDANIEL E., CHENNEY S.: Constrained animation of flocks. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), pp. 286–297.

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 594–603.

[BHW96] BARZEL R., HUGHES J. F., WOOD D. N.: Plausible motion simulation for computer graphics animation. In *Proceedings of the Eurographics workshop on Computer animation and simulation '96* (New York, NY, USA, 1996), Springer-Verlag New York, Inc., pp. 183–197.

[BMF03] BRIDSON R., MARINO S., FEDKIW R.: Simulation of clothing with folds and wrinkles. In *SCA '03: Proceedings of the*

2003 ACM SIGGRAPH/Eurographics symposium on Computer animation (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 28–36.

[BTH*03] BHAT K. S., TWIGG C. D., HODGINS J. K., KHOSLA P. K., POPOVIć Z., SEITZ S. M.: Estimating cloth simulation parameters from video. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 37–51.

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Computer Graphics Proceedings, Annual Conference Series* (1998), SIGGRAPH, pp. 43–54.

[BWK03] BARAFF D., WITKIN A., KASS M.: Untangling cloth. *ACM Trans. Graph. 22*, 3 (2003), 862–870.

[CF00] CHENNEY S., FORSYTH D. A.: Sampling plausible solutions to multi-body constraint problems. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 219–228.

[CGW*05] CUTLER L. D., GERSHBEIN R., WANG X. C., CURTIS C., MAIGRET E., PRASSO L., FARSON P.: An art-directed wrinkle system for cg character clothing. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM Press, pp. 117–125.

[CK02] CHOI K.-J., KO H.-S.: Stable but responsive cloth. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 604–611.

[DSB99] DESBRUN M., SCHRÖDER P., BARR A.: Interactive animation of structured deformable objects. In *Graphics Interface* (1999), pp. 1–8.

[Dun02] DUNCAN J.: Ring masters. *Cinefex 89* (April 2002), 64–131.

[FL04] FATTAL R., LISCHINSKI D.: Target-driven smoke animation. *ACM Trans. Graph. 23*, 3 (2004), 441–448.

[For03] FORDHAM J.: Neo realism. *Cinefex 95* (October 2003), 94–127.

[GK98] GIERING R., KAMINSKI T.: Recipes for adjoint code construction. *ACM Transactions on Mathematical Software 24* (December 1998), 437–474.

[GKJ*05] GOVINDARAJU N. K., KNOTT D., JAIN N., KABUL I., TAMSTORF R., GAYLE R., LIN M. C., MANOCHA D.: Interactive collision detection between deformable models using chromatic decomposition. *ACM Trans. Graph. 24*, 3 (2005), 991–999.

[KKiA05] KONDO R., KANAI T., ICHI ANJYO K.: Directable animation of elastic objects. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM Press, pp. 127–134.

[LCF05] LAI Y.-C., CHENNEY S., FAN S.: Group motion graphs. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), pp. 281–290.

[MTPS04] MCNAMARA A., TREUILLE A., POPOVIĆ Z., STAM J.: Fluid control using the adjoint method. *ACM Trans. Graph. 23*, 3 (2004), 449–456.

[Pro95] PROVOT X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface '95* (1995), Davis W. A., Prusinkiewicz P., (Eds.), Canadian Human-Computer Communications Society, pp. 147–154.

[PSE*00] POPOVIĆ J., SEITZ S. M., ERDMANN M., POPOVIĆ Z., WITKIN A.: Interactive manipulation of rigid body simulations. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 209–217.

[PSE03] POPOVIĆ J., SEITZ S., ERDMANN M.: Motion sketching for control of rigid-body simulations. *ACM Transactions on Graphics 22*, 4 (October 2003), 1034–1054.

[Rey87] REYNOLDS C. W.: Flocks, herds and schools: A distributed behavioral model. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (1987), ACM Press, pp. 25–34.

[ST97] SIRKES Z., TZIPERMAN E.: Finite difference of adjoint or adjoint of finite difference? *Monthly Weather Review 125* (December 1997), 3373–3378.

[SW95] SCHILLER A., WILLEBRAND J.: A technique for the determination of surface heat and freshwater fluxes from hydrographic observations, using an approximate adjoint ocean circulation model. *Journal of Marine Research 53* (May 1995), 433–451.

[SY05a] SHI L., YU Y.: Controllable smoke animation with guiding objects. *ACM Trans. Graph. 24*, 1 (2005), 140–164.

[SY05b] SHI L., YU Y.: Taming liquids for rapidly changing targets. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2005), ACM Press, pp. 229–236.

[SYWC05] SHI L., YU Y., WOJTAN C., CHENNEY S.: Controllable motion synthesis in a gaseous medium. *The Visual Computer 21*, 7 (2005), 474–487.

[TMPS03] TREUILLE A., MCNAMARA A., POPOVIĆ Z., STAM J.: Keyframe control of smoke simulations. *ACM Trans. Graph. 22*, 3 (2003), 716–723.

[TO99] TREMBLAY M., ONO H.: Multiple creature choreography on star wars. SIGGRAPH 99 AnimationSketch. In Conference Abstracts and Aplications, page 205, August 1999.

[VMT00] VOLINO P., MAGNENAT-THALMANN N.: Implementing fast cloth simulation with collision response. In *Computer Graphics International* (2000), pp. 257–.

[WK88] WITKIN A., KASS M.: Spacetime constraints. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1988), ACM Press, pp. 159–168.

[ZBLN95] ZHU C., BYRD R. H., LU P., NOCEDAL J.: *L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization*. Tech. Rep. NAM12, Northwestern University, Evanston, IL, 1995.