Journal for Geometry and Graphics
Volume 11 (2007), No. 1, 59–71.

Controllable Simulation of Deformable Objects Using Heuristic Optimal Control

Hongjun Jeon, Min-Hyung Choi *

Department of Electrical and Computer Engineering
University of Colorado. Boulder, CO 80309-0425, USA
email: jeon@colorado.edu
Deptartment of Computer Science and Engineering
University of Colorado. Denver, CO 80217, USA
email: min.choi@cudenver.edu

Abstract. Physically-based simulation techniques have been widely used in computer graphics because it creates highly realistic animation. However, due to the limitation of passive simulation and simplified modeling methods, it is very difficult to control the behavior of deformable objects directly. Ability to control the behavior of deformable objects is a very important feature for an effective simulation. In this paper, we present a novel interactive method and interface techniques for controlling the behavior of physically-based simulation of deformable objects. In our approach, an animator can select any part of the deformable structure and drag it to the desired location then our system automatically generates the motion path using a heuristic optimal method. Animators can focus on the final pose of the controlled object, without worrying about how to achieve the goal pose. Based on the displacement distance and the previous trajectory of the intended node, the optimal path generator computes the required control parameters that steer the intended node to the desired goal position while preserving the style of the original motion and minimizing the energy to achieve the goal pose. The goal oriented control scheme enables users to interactively control and redirect the motion of deformable objects and guarantees that the edited motion is physically conforming.

Key Words: Deformable objects simulation, physically-based simulation, motion control, optimal control

MSC: 68U05

1. Introduction

Forward dynamics systems are widely used in both rigid and deformable object animation because they produce highly realistic animation. However, direct and precise control of the

ISSN 1433-8157/\$ 2.50

behavior and trajectories of objects is difficult because it is based on a passive simulation model where the initial parameters and external forces are set upfront to compute the future motion. Existing approaches in deformable object simulation are mostly based on the passive dynamic simulation model. Under the current passive dynamic simulation paradigm, a small adjustment of the initial parameters can drastically affect the subsequent motion and result in different behaviors in the end. It is very difficult to predict if the final behavior would be what was desired for the simulation and it is even more challenging to control the dynamic behavior only by manipulating the initial conditions.

Another important cause of the difficulties in predicting and controlling the behavior of a deformable structure is that the widely used physical modeling techniques involve too much approximation and simplification to enhance the efficiency and programmability. The outcome of a simulation, using both the popular mass spring model and the finite element method (FEM), is heavily dependent to the combination of an element structure, the topology of connectivity, the granularity of elements, and coefficients that define the mechanical properties of the target deformable objects. If the granularity of the element structure is fine enough and all involved physical phenomena are precisely modeled, more realistic simulation can be achieved, but still it is insufficient to guarantee an exactly matched motion that an animator wants. Therefore, due to the limitation of the simplified modeling methods, certain behavior is somewhat unobtainable unless with a long, tedious trials and errors of parameter tweaking. Finding a correct combination of all contributing factors is often very difficult or almost practically unachievable. As long as we use approximated models like mass-spring or FEM, localized motion editing is inevitable to get the exactly intended and optimal behavior and therefore an intuitive, interactive controllable simulation is important and desirable especially in many production studios.

In many cases, animators would like to edit the end result or intermediate status of a simulation, just like using an inverse kinematics based editing tool for an articulated figure posing. A conventional way to edit a deformable structure simulation is to use a well-known key-frame interpolation and blending. However, it requires an animator to define a physically plausible path of a node, and therefore the quality of the motion is totally dependent to the animator's skill. Furthermore, each node of a deformable structure is connected to many neighboring nodes and they are mutually affecting all nodes' behavior. Manually generating realistic paths for all affected nodes in a deformable object is very difficult.

Controlling the dynamic behavior to a user's liking, especially in an interactive fashion, is very challenging since it requires sophisticated algorithms that steer the simulation by managing space and time dependent control forces over the neighboring structures on the fly. The thrust of this work is to automatically generate a set of appropriate control forces from user interaction metaphors, such as stopping a simulation and dragging a part of the object or interactively defining the trajectories of nodes. We adopt the two-phase simulation model: a pilot first-phase simulation with tentative coefficients is conducted and let the user edit the behavior of target nodes directly so that a desired behavior can be achieved in the second-phase simulation with added control parameters. Instead of repeating the initial parameter tweaking, we propose to use a set of time-encoded localized control parameters to steer the simulation. The proposed path generation algorithm automatically computes the motion paths and the required control parameters for the designated node and surrounding affected nodes, so that the goals can be satisfied in the second phase.

An attractive feature of our system is that the animator is able to directly edit the motion at any time without adjusting the underlying physical parameters. The effect of control and behavioral changes are localized to the intended time and space without altering the material properties or fundamental conditions of the simulation. After generating the motion paths, the user has an opportunity to fine tune the trajectories by moving control points of the trajectories. In editing the generated path, we use the interpolation polynomial in Newton's form [19] to set up the path for an object motion with 4 control points (Figure 3 (c)). The main contribution of our heuristic optimal method is substantial performance improvement and ease of implementation for the path generation by removing expensive gradient computation. Typically the gradient computation has been the bottleneck for computing control parameters because each parameter required its own derivative computation at every time steps.

2. Related works

Motion sketching and relevant interactive technique for controlling and manipulating rigid body simulation are introduced in [26, 27]. To achieve a desired motion, they changed initial physical parameters so the result of editing affects the simulation globally due to the tweaked fundamental parameters for the entire simulation. This method works well for rigid bodies but it is difficult to apply the same technique to deformable objects because of the large number of degrees of freedom and associated computational complexity in a deformable object. Motion synthesis techniques that generate motions by cutting and pasting motion capture data [1, 2], parameterization of motions [20] and path-based editing of existing motion data [14] were successfully applied to human motion synthesis. Similarly an inverse kinematics system, based on a learned model of human poses, was introduced in [15] but it is also limited to articulated figures like human body. James et al. [16] used pre-computation based data driven tabulation of state space for quick computation of shape and appearance of deformable objects. Their method permits real-time hardware synthesis of nonlinear deformation and real-time user interaction. But the pre-computing process to encompass possible future motions requires a heavy computation and it only covers a small portion of frequently animated modes. Constrained dynamic schemes have been widely studied in robotics and computer graphics for motion control. Putting a proper set of geometric constraints over a dynamic system is often used to control the behavior of dynamic system [3, 4, 8, 30]. Constraint-based methods work well for enforcing geometric constraints but it is very difficult to generate particular motions since the desired motion is in a higher abstract form and it should be parameterized with a set of proper constraints. Realistic simulation and motion control of smoke and water has drawn a lot of attention from the special effects industry. These phenomena require high-level control mechanisms for physics-based fluid simulations; for example, keyframe fluid simulation [22, 29], target-driven simulation [10], and level-set method [9, 11, 21, 23]. While these studies were successful in achieving the desired configuration of fluid, still the generation of optimal paths to form the configuration and an interactive editing in the middle of a simulation remain a significant challenge.

3. Motion control

3.1. Modeling of physical systems

Mass-spring system is a popular choice for modeling deformable objects in many computer graphics applications. We use a simple mass-spring-damper model with linear springs and dampers. Structural springs are connected in a rectangular format and shear springs are connected in diagonal direction. Primary and secondary bending spring structures are similar to [7].

We write our system of equations using $3N$ generalized coordinates, $\boldsymbol{q}$ which is the state vector of the system, where N is the number of discrete masses, and the generalized coordinates are simply the Cartesian coordinates of the discrete masses.

$$\boldsymbol{q} = [x_1\ y_1\ z_1\ x_2\ y_2\ z_2\ \ldots\ x_N\ y_N\ z_N]^T \tag{1}$$

Simulation of physical systems begins with an initial state $\boldsymbol{q}_0$ and repeatedly applies a sequence of operations $\boldsymbol{F}_i$ to the state, so that $\boldsymbol{q}_{i+1} = \boldsymbol{F}_i(\boldsymbol{q}_i,\ \dot{\boldsymbol{q}},\ \Delta t)$ for all $i \geq 0$, thus advancing the state through time. The function $\boldsymbol{F}_i$ is derived from the Newton's law and Δt is a step-size parameter.

3.2. Control parameters

We define a control vector, $\boldsymbol{u}$, which encodes all the external influences the system has over simulation. The physical system equation of movement of discrete masses with external control forces is

$$\boldsymbol{M}\,\ddot{\boldsymbol{q}} + \boldsymbol{K}_d\,\dot{\boldsymbol{q}} + \boldsymbol{K}_s\,\boldsymbol{q} = \boldsymbol{u} \tag{2}$$

Equation (2) expresses Newton's second law of motion for discrete masses. $\boldsymbol{M}$, $\boldsymbol{K}_d$, and $\boldsymbol{K}_s$ are $3N \times 3N$ matrixes containing discrete masses, damping coefficients, and spring stiffness constants. $\boldsymbol{u}$ is a $3N \times 1$ vector including external control forces. We can rewrite equation (2) in the following form:

$$\boldsymbol{M}\,\ddot{\boldsymbol{q}} = \boldsymbol{F}^A + \boldsymbol{u} \tag{3}$$

where $\boldsymbol{F}^A$ are applied, gravitational, damping, and spring forces acting on the discrete masses. With the Euler method, the equation of motion (equation (3)) along with the kinematic relationship between $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ is discretized as

$$\dot{\boldsymbol{q}}_{i+1} = \dot{\boldsymbol{q}}_i + \Delta t \boldsymbol{M}^{-1}\boldsymbol{F}^A\left(\boldsymbol{q}_i,\ \dot{\boldsymbol{q}}_i,\ t\right) + \Delta t\,\boldsymbol{M}^{-1}\boldsymbol{u}_i \tag{4}$$

$$\boldsymbol{q}_{i+1} = \boldsymbol{q}_i + \Delta t\,\dot{\boldsymbol{q}}_{i+1} \tag{5}$$

Substituting $\dot{\boldsymbol{q}}_{i+1}$ from equation (4) into equation (5) we obtain

$$\boldsymbol{q}_{i+1} = \boldsymbol{q}_i + \Delta t\,\dot{\boldsymbol{q}}_i + \Delta t\left[\Delta t\,\boldsymbol{M}^{-1}\boldsymbol{F}^A\left(\boldsymbol{q}_i,\ \dot{\boldsymbol{q}}_i,\ t\right) + \Delta t\,\boldsymbol{M}^{-1}\boldsymbol{u}_i\right] \tag{6}$$

$$\boldsymbol{u}_i = \boldsymbol{M}\left\{\boldsymbol{q}^*_{i+1} - \left[\boldsymbol{q}^*_i + \Delta t\,\dot{\boldsymbol{q}}^*_i + \Delta t^2\boldsymbol{M}^{-1}\boldsymbol{F}^A\left(\boldsymbol{q}^*_i,\ \dot{\boldsymbol{q}}^*_i,\ t\right)\right]\right\}/\Delta t \tag{7}$$

Our system tries to generate the new motion path $\boldsymbol{q}^*$ using the heuristic optimal method, computed by the control forces described in equation (7). These control forces should guarantee the minimal energy consumption, preservation of original motion style and meeting the goal poses.

3.3. Problem definition

The goal of our research is to provide users an ability to control the motion which appears physically correct, preserves the style of the original motion, and satisfies goals for a deformable object. In most motion control task, the first problem is to reach the desired position. To meet the goal, the control vector $\boldsymbol{u} = [\boldsymbol{u}_0\,\boldsymbol{u}_1\ \cdots\ \boldsymbol{u}_{n-1}]$ that guarantees meeting the goal positions within a finite time frames must be calculated. There are n unknown control forces but we only have one given condition ($\boldsymbol{q}_n$ is equal to the desired position). It is not enough condition to solve the problem and infinite number of solutions are possible so additional conditions are required. We create the cost function $\boldsymbol{J}$ which measures required energy and evaluates the goal satisfaction criteria. In addition the time range n over which the control forces are engaged should be defined. Another condition is that the new path should preserve the style of the original motion. If the objective is to minimize the energy only, then it may generate a jerky motion with sharp turns and uneven velocity distribution, resulting unnatural motion.

3.4. Cost function minimization

Given the generalized system equations

$$\boldsymbol{X}_{i+1} = \boldsymbol{A}\boldsymbol{X}_i + \boldsymbol{B}\boldsymbol{u}_i \quad \text{where} \quad \boldsymbol{X}_i = [\boldsymbol{q}_i \; \dot{\boldsymbol{q}}_i] \tag{8}$$

the optimal control problem determines control vector $\boldsymbol{u}_i$ that minimizes the cost function

$$\boldsymbol{J} = \sum_{i=0}^{n-1} \left[\boldsymbol{X}_i^T \boldsymbol{q} \boldsymbol{X}_i + \rho \boldsymbol{u}_i^2\right] \tag{9}$$

where $\boldsymbol{q} = \boldsymbol{q}^T \geq 0$ which is a positive semi-definite matrix and $\rho > 0$. $\boldsymbol{q}$ is often diagonal where the diagonal entries are chosen to weight different states by different amounts. The first term on the right-hand side of equation (9) can be modeled to measure how closely the simulation reaches the desired conditions. The second term accounts for the expenditure of the energy of the control forces. $\boldsymbol{q}$and ρ determine the relative importance of the error and the expenditure of this energy. There are many standard numerical methods for minimizing the cost function [5, 12, 13, 18, 25]. Conventional methods to find the control vector that minimizes the cost function typically use the Riccati equation [18, 25] or a set of Lagrange multipliers [5, 12, 24]. Since these numerical methods are a derivative based optimization, we must not only evaluate $\boldsymbol{J}$, but also compute its gradient $d\boldsymbol{J}/d\boldsymbol{u}_i$ at every time step. This gradient computation for deformable structure is very expensive and becomes a bottleneck because usually a large number of nodes are involved and each parameter requires its own derivative computation. The proposed heuristic optimal control method eliminates this bottleneck completely to improve the performance.

3.5. Motion path generation using the heuristic optimal control

The formulation of an optimal control problem requires a mathematical model of the process to be controlled [6, 17], a statement of the physical constraints, and specification of a performance criterion. Equation (2) is our mathematical model which expresses Newton's second law of motion. We represent motion path $\boldsymbol{q}_{ij}$

$$\boldsymbol{q}_{ij} = \left[\boldsymbol{x}_{ij} \; \boldsymbol{y}_{ij} \; \boldsymbol{z}_{ij}\right]^T \tag{10}$$

where i is the time steps and j denotes the selected node to be moved. From now on, j will be omitted so $\boldsymbol{q}_i$ means $\boldsymbol{q}_{ij}$. The next step is to define the physical constraints on the state and control values but the time range T_r which a user wants to control must be defined in advance. A user selects one node which a user wants to move at current time T_c and simply

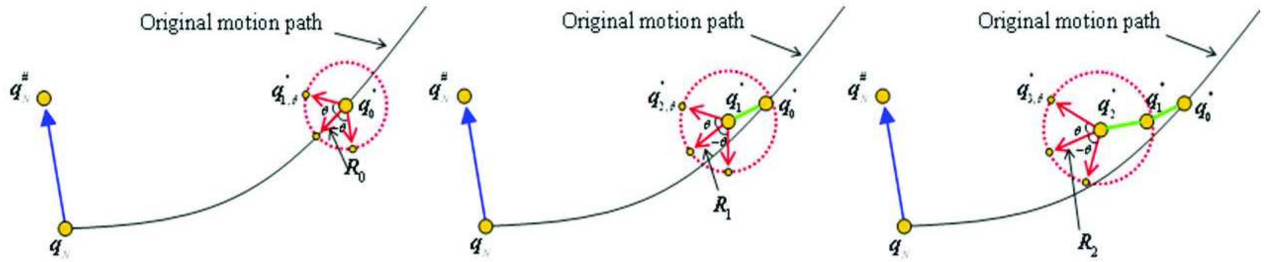


Figure 1: These three figures show the heuristic optimal path generation process using the multistage decision process. The red circle is the region to find the locally minimum optimal position. $\boldsymbol{R}_i$ is the radius vector that determines the direction of control force. Green line is the newly generated heuristic optimal trajectory.

drag it to a desired location $\boldsymbol{q}_n^\sharp$ (i=n means $i = T_c$). We set a time range to be directly proportional to the length of change between the current position $\boldsymbol{q}_n$ and the desired position $\boldsymbol{q}_n^\sharp$ (the blue line in Figure 1).

$$T_r = \alpha \|\boldsymbol{q}_n^\sharp - \boldsymbol{q}_n\|^2 \tag{11}$$

where α is a scale factor. If T_s ($T_s = T_c - T_r$) is the starting time to put control forces and T_c is the time which a user want to change the scene, then

$$\boldsymbol{q}_0^* = \boldsymbol{q}_0,\ \dot{\boldsymbol{q}}_0^* = \dot{\boldsymbol{q}}_0, \text{ and } \boldsymbol{q}_n^* = \boldsymbol{q}_n^\sharp \quad (i = 0 \text{ means } i = T_s) \tag{12}$$

In addition, if the maximum control force is $\boldsymbol{F}_{max}$, then the control forces must satisfy

$$\|\boldsymbol{u}_i\|^2 \le \boldsymbol{F}_{max} \quad for\ T_s \le i < T_c \quad or \quad 0 \le i < n \tag{13}$$

The cost function $\boldsymbol{J}$ measures how closely the affected node reaches the desired position and also penalizes the system for using too much control forces. These two conditions can be written as:

$$\boldsymbol{J} = \frac{1}{2}\sum_{i=0}^{n-1} \left[\beta\|\boldsymbol{q}_n^\sharp - \boldsymbol{q}_{i+1}^*\|^2 + \gamma\|\boldsymbol{u}_i\|^2\right] \tag{14}$$

$$\text{subject to: } \|\boldsymbol{q}_n^\sharp - \boldsymbol{q}_{i+1}^*\|^2 \subseteq \boldsymbol{D} \text{ and } \boldsymbol{u}_i \le \boldsymbol{F}_{max} \text{ for } 0 \le i < n$$

where $\beta \ge 0$ and $\gamma \ge 0$ are weighting factors. Large γ puts more emphasis on the amount of control force while β concerns about the goal satisfaction. The optimal control problem is to find an admissible control forces $\boldsymbol{u}$ which causes the system to follow an admissible trajectory $\boldsymbol{q}^*$.

Instead of solving the problem with the conventional nonlinear optimization method that involves expensive gradient computation, our algorithm tries to find local radial vectors only for the new control force direction that can improve the cost function at every iteration step independently. This multistage decision process to get a locally optimal path is shown in Figure 1. Let us denote the scalar ratio $\boldsymbol{L}$ between the length of original path and the length of desired path from T_s to T_c;

$$\boldsymbol{L} = \delta\frac{\|\boldsymbol{q}_0 - \boldsymbol{q}_n^\sharp\|^2}{\|\boldsymbol{q}_0 - \boldsymbol{q}_n\|^2} \tag{15}$$

where δ is a scale factor. Also we define the radius vectors $\boldsymbol{R}_i$;

$$\boldsymbol{R}_i = \boldsymbol{L}\left[\boldsymbol{q}_{i+1} - \boldsymbol{q}_i\right] \quad \text{for } 0 \leq i < n \tag{16}$$

Then the algorithm initially sets a seek angle range θ and searches for a condition that reduces the cost function by altering the search angle at a discrete interval, from $-\theta$ to θ. $\boldsymbol{q}_{i+1,\theta}$ is the new point after the rotation $\boldsymbol{R}_i$ for θ degrees centered at $\boldsymbol{q}_i^*$.

$$\boldsymbol{q}_{i+1,\theta} = \boldsymbol{q}_i^* + \mathbf{Rot}(\theta) \cdot \boldsymbol{R}_i \tag{17}$$

where $\mathbf{Rot}(\theta)$ is a rotation matrix. Control forces $\mathbf{u}_{i,\theta}$ for every point $\boldsymbol{q}_{i+1,\theta}$ on the circle with the radius $\boldsymbol{R}_i$ can be calculated by equation (7) for $0 \leq i < n$.

$$\boldsymbol{u}_{i,\theta} = \boldsymbol{M}\left\{\boldsymbol{q}_{i+1,\theta} - \left[\boldsymbol{q}_i^* + \Delta t\, \dot{\boldsymbol{q}}_i^* + \Delta t^2 \boldsymbol{M}^{-1} \boldsymbol{F}^A\left(\boldsymbol{q}_i^*,\, \dot{\boldsymbol{q}}_i^*,\, t\right)\right]\right\} / \Delta t \tag{18}$$

By repeating this process until the seek angle reaches the predefined boundary, locally optimal control forces $\boldsymbol{u}$ to minimize the cost function can be found and the heuristic motion path $\boldsymbol{q}^*$ can be generated.

$$\min\left(\boldsymbol{J}_{i,\theta}\right) = \min\left[\frac{1}{2}\left(\beta\|\boldsymbol{q}_{i+1,\theta} - \boldsymbol{q}_n^*\|^2 + \gamma\|\boldsymbol{u}_{i,\theta}\|^2\right)\right] \tag{19}$$

The solutions of equation (19) are our newly obtained motion path $\boldsymbol{q}^*$. Another goal that must be satisfied concurrently is to generate the path that reaches the desired position while preserving the style of the current motion. Finding the locally optimal control force, obtained in the radial vector form within the seek circle, is particularly convenient for preserving the style of the original motion since the arc length of new step is proportional to the step length of the original motion. Since the style of a motion is mostly dictated by the magnitude and direction of each step, radial vectors with proportionally scaled arc lengths, combined with minimally deviated seek angles, can help to conserve the style of the original motion.

3.6. User defined spartial and temporal control

Even though the optimal path generation method minimizes the weighted sum of required energy to achieve the desired end result and to maintain the smoothness of the trajectory, the generated path may not be an exactly intended path both in the spatial and temporal sense. Our system allows a user to manually edit the path by moving control points of a curve duplicated from the generated path. This manual intervention process may negate the optimal path generation process and it could possibly involve physically non-conforming motion, and therefore it is restricted only for a slight spatial and temporal adjustment from the automatically generated optimal path. The interpolation polynomial in Newton's form [19] is used for blending the user defined path and the generated optimal path. When the new node position is remotely displaced, the overall speed of motion is changed as well, since the arc length of the new motion is altered from the original motion. In most cases the optimal path generation algorithm disperses the speed discrepancies naturally but sometime further temporal fine tuning is desirable. In order to satisfy the temporal goals, the distance function can be expressed as:

$$d_{i+1} = \|\boldsymbol{q}_{i+1} - \boldsymbol{q}_i\|^2,\ d_{i+1}^* = \|\boldsymbol{q}_{i+1}^* - \boldsymbol{q}_i^*\|^2 \quad \text{for } 0 \leq i < n \tag{20}$$

where d_i is original arc-length and d_i^* is arc-length of the desired motion path at time i. d_0 and d_0^* are zero. From equation (20), total lengths of an original path and a desired path can be written as d_{Tot} and d_{Tot}^*.

$$d_{Tot} = \sum_{i=0}^{n-1} d_{i+1}, \ d_{Tot}^* = \sum_{i=0}^{n-1} d_{i+1}^* \tag{21}$$

We also define a scale vector $\boldsymbol{S}$ which is a ratio between a total moving length of original path and a length of each time step within the interval $[T_s, \ T_c]$. $\boldsymbol{S}^*$ represents a relationship between total arc-length of a desired motion path and a length of each time step:

$$\begin{aligned} \boldsymbol{S} \ &= [s_1 \ s_2 \ \cdots \ s_n], \ s_{i+1} = d_{i+1}/d_{Tot} \\ \boldsymbol{S}^* &= [s_1^* \ s_2^* \ \cdots \ s_n^*], \ s_{i+1}^* = d_{i+1}^*/d_{Tot}^* \end{aligned} \tag{22}$$

where $0 \leq i \leq n-1$. Total sum of scale vector elements is one.

$$\sum_{i=1}^{n} s_i = 1, \quad \sum_{i=1}^{n} s_i^* = 1 \tag{23}$$

The temporal control method for motion control allows users to change the speed of new motions similar to [28]. To maintain the characteristics of the original simulation, we proportionally divide a desired path compared with an original path. It involves following 4 steps. First, $\boldsymbol{S}^*$ is set to $\boldsymbol{S}$ then the distance vector of the desired motion path, $\boldsymbol{d}^* = [d_1^*, \ d_2^*, \ \cdots \ , \ d_n^*]$, is calculated by equation (22).

$$d_{i+1}^* = s_{i+1} d_{Tot}^* = d_{i+1} d_{Tot}^* / d_{Tot} \tag{24}$$

Third, $\boldsymbol{q}_i^*$ is obtained using distance vector $\boldsymbol{d}^*$. Finally the control force vector u can be computed by equation (7). The lower part of figure 3 shows the automatically generated optimal path in blue color, user directed path in orange color and the final blended path in the cyan line (dotted line).

4. Multiple paths control

A deformable object involves many particles and in order to control the behavior of the object, all affected particles need to be handled at the same time. Controlling multiple individual particles may incur conflicting control forces, or they may affect each other in unexpected ways since they are all connected in the mesh structure. To address the conflicting control forces issue we use a time-stamped sequential control force application for the overlapped control range of multiple paths control. For instance, when there are two particles' paths to be edited, the time range of the optimal path generation is redefined (mostly expanded) to deal with the possible interplay. Let's assume the time range of the first particle A is $[t_1, \ t_2]$ and $[t_3, \ t_4]$ for the particle B. If t_3 is earlier than t_1 the particle B's newly edited behavior will affect the particle A's force computation at timet_3. To avoid this conflict,t_1 is redefined to t_3 so that the two particles' mutual influences can be considered in the optimal paths generation. Figure 2 shows two optimally generated paths with overlapped time range.

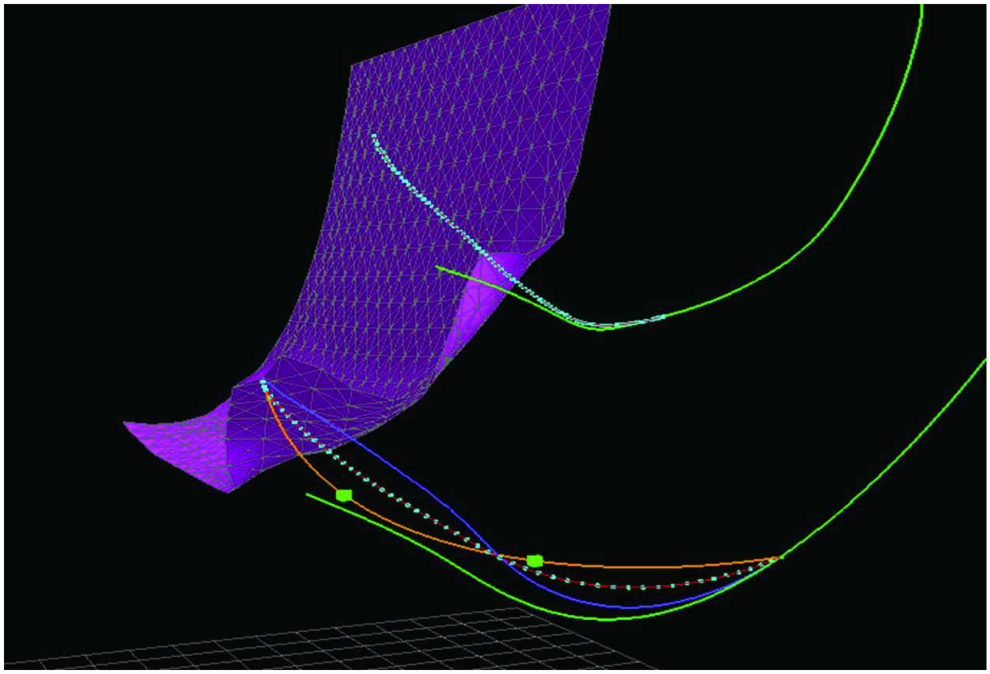


Figure 2: This figure shows that the multiple goals are simultaneously satisfied and it displays two optimally generated paths. The lower part of the figure indicates how to edit the path by moving control points and blending the user defined path and the generated optimal path.

5. Experiments

We have performed three sample cloth simulations shown in Fig. 3, Fig. 4, and Fig. 5. The example in Fig. 3 uses a mesh with 20×20 nodes to illustrate overall procedures of motion control and final editing. It is the screen capture of a live simulation of free falling cloth patch. In this example, a user can select any part of the patch, move it to the desired position (Fig. 3(a)), and the system generates the path using our algorithm (Fig. 3(b)). If a user wants to edit the generated path further, the path can be edited by moving control points and then blend two paths (Fig. 3(c)). Path generation can be accumulated for multiple node paths control (Fig. 3(e)). The example in Fig. 4 shows another motion path generation with collisions in the middle of the editing time range. After the collision, a corner of the cloth is excessively curled in the initial simulation but a simple relocation of a node corrects the problem while keeping the physically realistic behavior intact. Two examples in Fig. 3 and Fig. 4 are point based control. In those two example, the user selected one control node and moved it to the disired position. The example in Fig. 5 shows how to generate the user defined target pose. In this example, a user can select the scene to be controlled then choose the control region and define the target pose by changing the curl and rotate angle. Those two angles decide the shape of the target pose.

Even if the motion can be generated automatically, the process involves several user-defined coefficients such as β for the evaluation of the goal satisfaction, and γ for the admissible amount of control force. β and γ are dependent on the magnitude of distance error and control force. If γ is too large, that signifies small control forces applied to the system, the final position may not reach to the desired position. If β exceeds a threshold, the generated path tends to be the shortest path, and the route could be unnaturally straight. To avoid this, we limit the seek angle θ within minimal range at initial steps and gradually increase

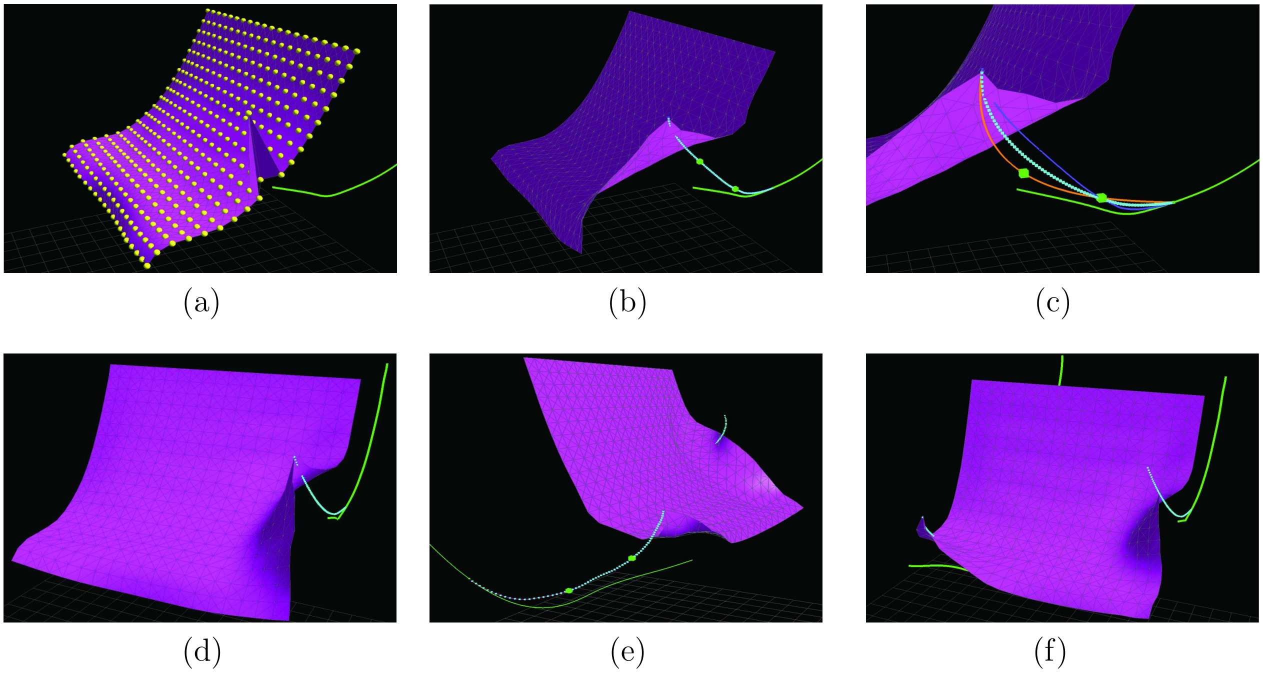


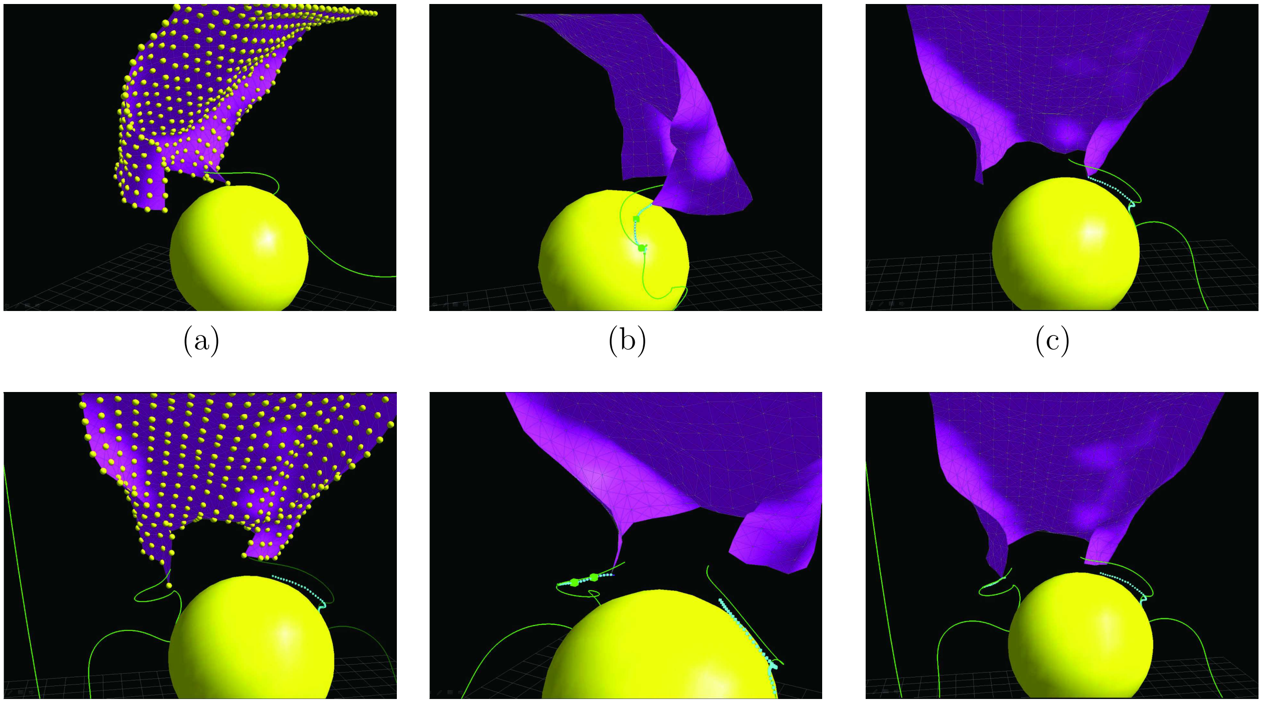
Figure 3: These series of figures show how to control the behavior of a cloth patch. (a) A user can define a final pose of a cloth by stopping the simulation and moving a node to the desired position. (b) Heuristic optimal path generator produces a new path (cyan line) that satisfies the new goal position. The green line is the original path. (c) A user can further fine tune the newly generated path by moving control points (green cubes) and blending with the user defined path (red line). (d) The edited motion that satisfies the desired position is shown (e) A user can control multiple nodes' behaviors simultaneously. The second node is moved to the other desired position and another optimal path is generated without violating the first node's optimal path conditions. (f) The final motion that satisfies both control conditions is shown.

the range. Since appropriate coefficients are problem dependent, they can be adjusted by the system designer to reflect the intended motion.

6. Conclusions and discussion

This paper reports a new heuristic optimal control method to control the behavior of physically-based simulation of deformable objects. We have shown that the motion path and the required control parameters can be automatically generated and the new simulation in the second phase satisfies all user defined goals by the control parameters. Since most existing optimal control methods are gradient based, they require expensive evaluation of a cost function J and its gradient dJ/du. The gradient computation should be calculated at every time steps for each parameter, so traditionally it has been the bottleneck of the deformable objects motion control. We cast the problem to a combinatorial discrete optimization to avoid the expensive gradient computation. The proposed heuristic optimal control method is simple and substantially faster. It also helps us in preserving the style of the original simulation in more straightforward manner.

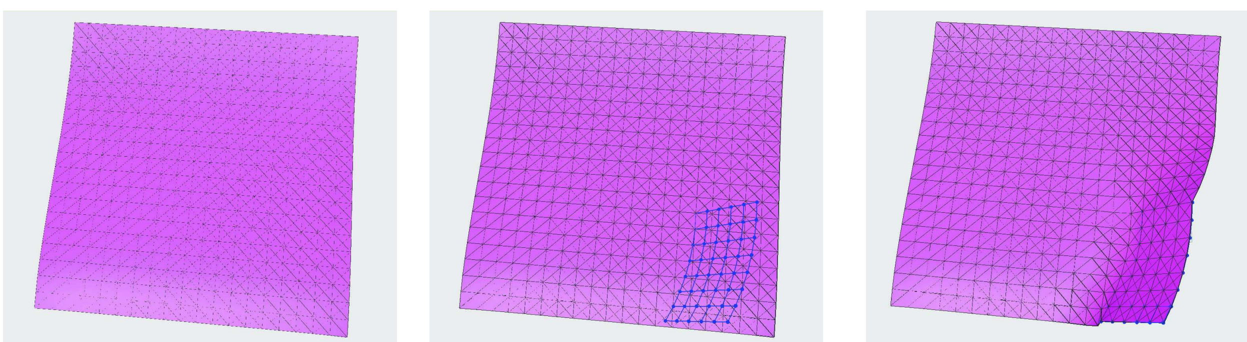
In the future, to optimize the search space and to accommodate the physical conditions of unreachable regions, new algorithms to detect the safe regions (physically possible regions)



(a) (b) (c)

(d) (e) (f)

Figure 4: These series of figures show how to control the unexpected behavior when a cloth patch is hitting a ball.



(a) (b) (c)

Figure 5: These series of figures show how to control the target motion. (a) A user selects the scene to be controlled. (b) A user can define the target pose by selecting the control region and changing the curl and rotate angles. (c) The final motion that satisfies the user defined target pose.

and a new scheme to incorporate this condition in the heuristic optimal path search needs to be further investigated. More studies on user interface to inform the user about the unreachable regions or accumulated collisions along the path is needed. User interaction to define a new desirable path could be further improved by a pattern and example based motion control and editing.

References

[1] O. Arikan, D.A. Forsyth: *Interactive motion generation from examples*. Proc. SIGGRAPH 2002, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2002, pp. 483–490.

[2] O. Arikan, D.A. Forsyth, J.F. O'Brien: *Motion synthesis from annotations*. Proc. SIGGRAPH 2003, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2003, pp. 402–408.

[3] D. Baraff, A. Witkin: *Large steps in cloth simulation*. Proc. SIGGRAPH 1998, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 1998, pp. 1–12.

[4] R. Barsel, A.H. Barr: *A modeling system based on dynamic constraints*. Proc. SIGGRAPH 1988, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 1988, pp. 179–188.

[5] W.L. Brogan: *Modern control theory*. Prentice-Hall, Inc. 1991.

[6] C.T. Chen: *Linear system theory and design*. 3rd edition. Oxford Univ. Press. 1999.

[7] K.J. Choi, H.S. Ko: *Stable but responsive cloth*. Proc. SIGGRAPH 2002, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2002, pp. 604–611.

[8] M.H. Choi, H. Min: *Implicit constraint enforcement for stable and effective control of cloth behavior*. Technical report, Univ. of Colorado at Denver, Computer Science and Engineering Dept. TR-03-01 (2004).

[9] D. Enright, S. Marschner, R. Fedkiw: *Animation and rendering of complex water surfaces*. Proc. SIGGRAPH 2002, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2002, pp. 736–744.

[10] R. Fattal, D. Lischinski: *Target-driven smoke animation*. Proc. SIGGRAPH 2004, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2004, pp. 441–448.

[11] N. Foster, R. Fedkiw: *Practical animation of liquids*. Proc. SIGGRAPH 2001, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2001, pp. 23–30.

[12] G.F. Franklin, J.D. Powell, M. Workman: *Digital Control of Dynamic Systems*. Addison-Wesley, 1998.

[13] R. Franz, M. Milam, J. Hauser: *Applied receding horizon control of the Caltech ducted fan*. Proc. American Control Conference 2002, pp. 3735–3740.

[14] M. Gleicher: *Motion path editing*. ACM Symposium on Interactive 3D Graphics 2001.

[15] K. Grochow, S.L. Martin, J. Popovic, A. Hertzmann: *Style-based inverse kinematics*. Proc. SIGGRAPH 2004, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2004, pp. 522–531.

[16] D.L. James, K. Fatahalian: *Precomputing interactive dynamic deformable scenes*. Proc. SIGGRAPH 2003, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2003, pp. 879–887.

[17] J.H. Juang: *Applied system identification*. Prentice-Hall, Inc. 1994.

[18] T. Kailath: *Linear systems*. Prentice-Hall, Inc. 1980.

[19] D. Kincaid, W. Cheney: *Numerical analysis: Mathematics of scientific computing*. 3rd edition. Brooks/Cole. 2002.

[20] L. Kovar, M. Gleicher: *Automated extraction and parameterization of motions in large data sets*. Proc. SIGGRAPH 2004, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2004, pp. 559–568.

[21] F. Losasso, F. Gibou, R. Fedkiw: *Simulating water and smoke with an octree data structure*. Proc. SIGGRAPH 2004, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2004, pp. 457–462.

[22] A. McNamara, A. Treuille, Z. Popovic, J. Stam: *Fluid control using the adjoint method*. .. Proc. SIGGRAPH 2004, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2004, pp. 449–456

[23] D. Nauyen, R. Fedkiw, H. Jensen: *Physically based modeling and animation of fire*. Proc. SIGGRAPH 2002, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2002, pp. 736–744.

[24] K. Ogata: *Discrete-Time Control Systems*. 2nd edition. Prentice-Hall. 1995.

[25] K. Ogata: *Modern Control Engineering*. 3rd edition. Prentice-Hall. 1997.

[26] J. Popovic, S.M. Seits, M. Erdmann, Z. Popovic, A. Witkin Interactive manipulation of rigid body simulations Proc. SIGGRAPH 2000, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2000, pp. 209–218

[27] J. Popovic, S. M. Seits, M. Erdmann: *Motion sketching for control of rigid body simulations*. ACM Transactions on Graphics **22**, No. 4, 1034–1054 (2003).

[28] S.S. Snibbe: *A direct manipulation interface for 3D computer animation*. Computer Graphics Forum **14** (3), 271–283 (1995).

[29] A. Treuille, A. McNamara, Z. Popovic, J. Stam: *Keyframe control of smoke simulations*. Proc. SIGGRAPH 2003, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 2003, pp. 716–723.

[30] A. Witkin, M. Gleicher, W. Welch: *Interactive Dynamics*. Proc. SIGGRAPH 1990, ACM Press/ACM SIGGRAPH, Comput. Graphics Proc. 1990, pp. 11–21.

Received August 7, 2006; final form July 5, 2007