CPSC533B Course Project: Spacetime Constraints

Lin Zhong
University of British Columbia

Abstract

Computer animation has two methods of modeling: one is physical-based modeling; the other is example-based modeling. Although the example-based modeling now is the leading direction, physical-based modeling has its important as the basement in the area of computer animation. Spacetime constraints are a physical-based method for creating character animation. It specifies what the character should do-- "moving from here to there, " how the motion should be performed, for instance "minimize waste energy", the character's physical structure—the geometry, mass, connectivity, etc. of the parts; and the physical resources available to the character to accomplish the motion, for instance the character's muscles, a floor to push off from, etc. The requirements contained in this description, together with Newton's laws, comprise a problem of constrained optimization. The solution to this problem is a physically valid motion satisfying the "what" constraints and optimizing the "how" criteria. I experiment as examples a simple accelerate particle and discuss the solution for a swing pendulum and a luxo lamp. These realistic motions conform to such principles of traditional animation as anticipation, squash-and-stretch, follow-through, and timing. This result shown that using efficiency as the criterium for selecting among the physically valid ways a creature can accomplish a task yields visually pleasing results.

Introduction and Previews Work

Computer animation has made enormous strides in the past several years and the example-based modeling now is the leading direction. Physical-based modeling could solve simple problem much more satisfied than example-based modeling. However, in practical, it is really hard to solve complexes models currently. Not only the difficulties in analyzing the model, but also tough in deriving the required differentiations of those complicate algebraic expressions by code or by hand. The main object that I choose the spacetime constraints, as my course project for animation is to get some basic and canonical knowledge of physical-based modeling solution.

Although it is hard, researchers still have done plenty of progress on the problem. Pixar's Luxo, Jr. [10] marked a turning point as perhaps the first computer-generated work to compete seriously with works of traditional animation on every front. Although Luxo, Jr. showed us that the team of animator, keyframe system, and renderer can be a powerful one, the responsibility for defining the motion remains almost entirely with the animator. Then, there has been considerable interest in incorporating physics into animation using simulation methods. [7, 14, 15, 1, 13, 4, 6] The appeal of physical simulation as an animation technique lies in its promise to produce realistic motion automatically by applying the same physical laws that govern real objects' behavior. Unfortunately, the realism of simulation comes at the expense of control. Simulation methods solve initial value problems: the course of a simulation is completely determined by the objects'

initial positions and velocities, and by the forces applied to the objects along the way. An animator, however, is usually concerned as much with where the objects end up and how they get there as where they begin. Problems cast in this form are not initial value problems.

For instance, while simulating a bouncing ball is easy enough, making the ball bounce to a particular place requires choosing just the right starting values for position, velocity, and spin. Making these choices manually is a painful matter of trial and error. Problems such as this one, in which both initial and final conditions are partially or completely constrained, are called two-point boundary problems, requiring more elaborate solution methods than forward simulation.[3] Character animation poses a still more difficult problem. Animals move by using their muscles to exert forces that vary as a function of time. Calculating the motion by simulation is straightforward once these time-dependent force functions are known, but the difficult problem is to calculate force functions that achieve the goals of the motion. Specifying these functions by hand would be hopeless, equivalent to making a robot move gracefully by manually varying its motor torques. In an effort to reconcile the advantages of simulation with the need for control, several researchers [1, 7] have proposed methods for blending positional constraints with dynamic simulations. The idea behind these methods is to treat kinematic constraints as the consequences of unknown "constraint forces," solve for the forces, then add them into the simulation, exactly canceling that component of the applied forces that fights against the constraints. Constraint force methods permit parts, such as a character's ands or feet, to be moved along predefined keyframed trajectories, but provide no help in defining the trajectories, which is the central problem in creating character animation. While allowing a character to be dragged around manually like a marionette, constraint forces sidestep the central issue of deciding how the character should move.[17].

These shortcomings led us to adopt a new formulation of the constraint problem, whose central characteristic is that we solve for the character's motion and time-varying muscle forces over the entire time interval of interest, rather than progressing sequentially through time. Because we extend the model through time as well as space, we call the formulation spacetime constraints. The spacetime formulation permits the imposition of constraints throughout the time course of the motion, with the effects of constraints propagating freely backward as well as forward in time. Constraints on initial, final, or intermediate positions and velocities directly encode the goals of the motion, while constraints limiting muscle forces or preventing interpenetration define properties of the physical situation. Additionally, Newtonian physics provides a constraint relating the force and position functions that must hold at every instant in time. Subject to these constraints we optimize functions that specify how the motion should be performed, in terms of efficiency, smoothness, etc. Solving this constrained optimization problem yields optimal, physically valid motion that achieves the goals specified by the animator.

A spacetime particle

As a concrete example, I did a simple implement of an accelerating particle, influenced by gravity, and equipped with a "jet force". With no restrictions on the forces exerted by its engine, the particle can move in arbitrary trajectory. The problem here is that of making the particle fly from a given starting point to a given destination in a fixed period of time, with minimal fuel consumption. This experimental problem is too simple to

produce any really interesting motion, but it exhibits all the key elements of the method, and it is the basement of further extension.

1 Problem formulation

Let the particle's position as a function of time be y(t), and the time-varying jet force be f(t). Suppose for simplicity that the mass of the fuel is negligible compared to that of the particle, so the total mass may be treated as a constant, with a constant gravitational force. Then the particle's equation of motion is

$$m\ddot{y} - f - mg = 0 \quad (1)$$

where $\ddot{y}$ is the second time derivative of position. Given the function f(t) and initial values for y and $\ddot{y}$ x at some time t0, the motion y(t) from t0, could be obtained by integrating equation 1 to solve the initial value problem. Instead we wish to make the particle fly from a known point ***a*** to a known point ***b*** in a fixed period of time. Suppose for simplicity that the rate of fuel consumption is $|f|^2$. In that case, we have constraints $y(t_0) = a$ and $y(t_1) = b$ subject to the objective function

$$R = \int_{t_0}^{t_1} |f(t)|^2 dt$$

which must be minimized. The problem then is to find a force function, defined on the interval (t0, t1), such that the position function obtained by solving equation (1) satisfies the boundary constraints, and such that the objective function R is a constrained minimum. There exist a variety of standard approaches to solving problems of this form. I used the optimal control, which solve the initial value problem within each iteration, using the equations of motion to obtain the position function from the force function. ([12]) It represents the functions y(t) and f(t) independently. The equation of motion then enters as a constraint that relates the two functions, to be satisfied along with the other constraints during the solution process. Each function is discredited, that is, represented as a sequence of values, with time derivatives approximated by finite differences. This approach leads to a classical problem in constrained optimization, for which a variety of standard solution algorithms are available.

Let the discredited functions y(t) and f(t) be represented by sequences of values y_i and f_i, $0 \le i \le n$, with h the time interval between samples. To approximate the time derivatives of $y(t)$, we use the finite difference formulas:

$$\dot{y} = \frac{y_i - y_{i-1}}{h} \quad (2)$$

$$\ddot{y} = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \quad (3)$$

Substituting these relations into equation 1 gives n "physics constraints" relating the y_i's to the f_i's,

$$p_i = m\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} - f_i - mg = 0, \qquad 1 < i < n$$

(4)

In addition we have the two boundary constraints

$$c_a = y_1 - a = 0$$

and

$$c_b = y_n - b = 0$$

Assuming that f(t) is constant between samples, the objective function R becomes a sum

$$R = h\sum_i |\, f_i \,|^2 \qquad (5)$$

which is to be minimized subject to the constraints. The discretized objective and constraint functions are now expressed in terms of the y_i's and f_i's which are the independent variables to be solved for.

2 Numerical Solution

Then we need a numerical solution to find the correct answer to those equations. We know constraints y_1, y_n and motion equation (4) and objective function (5), they are known as Constraints factor. The scalar independent variables S_i, which we don't know, are the positions $y_2 \cdots y_{n-1}$ and the forces $f_2 \cdots f_{n-1}$.

In these terms, the standard constrained optimization problem is "Find S_i that minimizes R subject to C_i". The solution method I choose to use is a variant of Sequential Quadratic Programming (SQP), described in detail in [3]. The method computes a second-order Newton- Raphson step in R, and a first-order Newton-Raphson step in the Ci's, and combines the two steps by projecting the first onto the null space of the second (that is, onto the hyperplane for which all theCi's are constant to first order.)

Because it is first-order in the constraint functions and second-order in the objective function, the method requires that we be able to compute two derivative matrices: the Jacobian of the constraint functions, given by

$$J_{ij} = \frac{\partial C_i}{\partial S_j}$$

and the Hessian of the objective function,

$$H_{ij} = \frac{\partial^2 R}{\partial S_i \partial S_j}$$

The SQP step is obtained by solving two linear systems in sequence. The first,

$$-\frac{\partial R}{\partial S_i} = \sum_j H_{ij}(S_j' - S_j)$$

yields a step S_j' that minimizes a second-order approximation to R, without regard to the constraints. The second,

$$-C_i = \sum_j J_{ij}(S_j'' - S_j')$$

yields a step $S_j^{''}$ that drives linear approximations to the Ci's simultaneously to zero, and at the same time projects the optimization step Sj onto the null space of the constraint Jacobian. The final update is $S_j^{''}$. The algorithm reaches a fixed point when Ci=0 and when any further decrease in R requires violating the constraints.

3 Linear system solving

The choice of a method for solving these linear systems is critically important, because the matrices here although is large, it is sparse and symmetric. So I choose to use the adapted sparse conjugate gradient (CG) algorithm described in [11], which is O_n2_ for typical problems. The CG algorithm solves the matrix equation a =Mb by iteratively minimizing $| a - Mb |^2$, giving a least-squares solution to overconstrained problems. Provided that a zero starting-point is given for b, the solution vector is restricted to the null-space complement of M.

4 Matrix evaluation.

Then the most important factor , which used in solving the equation by applying the SQP algorithm of the moving particle example, is Jacobian of the motion equation and Hessian of the objective function.

Those evaluations are straightforward. The Jacobian of the physics constraint is given by

$$\frac{\partial p_i}{\partial x_j} = \begin{cases} -2m/h^2 & i = j \\ m/h^2 & i = j \pm 1 \\ 0 & otherwise \end{cases}$$

$$\frac{\partial p_i}{\partial x_j} = \begin{cases} 1 & i = j \\ 0 & otherwise \end{cases}$$

The Jacobians of the boundary constraints are trivial. The gradient of R is

$$\frac{\partial R}{\partial f_j} = 2 f_i$$

and the Hessian is

$$\frac{\partial^2 R}{\partial f_i \partial f_j} = \begin{cases} 2 & i = j \\ 0 & otherwise \end{cases}$$

Although it happens that the toy problem we chose constrains initial and final positions, nothing in the solution approach depends on this configuration: initial and final conditions could be left free, and constraints at arbitrary internal points could be added. Moreover, arbitrary constraints of the form F(Si)= 0, not just position constraints, may be added provided that the constraint functions and their derivatives can be evaluated.

The linear system for optimization of objective function is

$$\begin{bmatrix} -2f_1 = 2(f_1' - f_1) & \cdots & 0 \\ \vdots & & \\ -2f_i = & 2(f_i' - f_i) & \\ \vdots & & \\ -2f_{n-1} = 0 & \cdots & 2(f_{n-1}' - f_{n-1}) \end{bmatrix}$$

And the constraints on the motion equation is

$$\begin{bmatrix} -p_1 = (f_1'' - f_1') & \cdots & 0 \\ \vdots & & \\ -p_i = & (f_i'' - f_i') & \\ \vdots & & \\ -p_{n-1} = 0 & \cdots & (f_{n-1}'' - f_{n-1}') \end{bmatrix}$$

$$\begin{bmatrix} -p_1 = -\frac{2m}{h^2}(y_1' - y_1) & \frac{m}{h^2}(y_2' - y_2) & & 0 \\ \vdots & & & \\ -p_i = & \frac{m}{h^2}(y_{i-1}'' - y_{i-1}) & -\frac{2m}{h^2}(y_i'' - y_i) & \frac{m}{h^2}(y_{i+1}'' - y_{i+1}) \\ \vdots & & & \\ -p_{n-1} = 0 & & \frac{m}{h^2}(y_{n-2}'' - y_{n-2}) & -\frac{2m}{h^2}(y_{n-1}'' - y_{n-1}) \end{bmatrix}$$

Then iteratively until $p_1 \cdots p_2 = \{0 \cdots 0\}$.

5 Extension to complex models

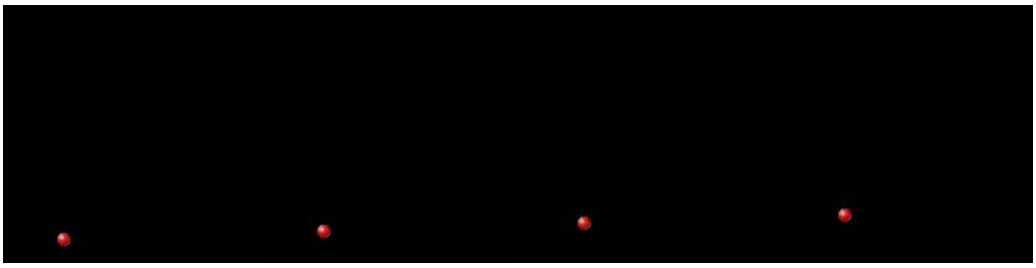
After the simple experiment, we have the sense of how to solve the simple models by spacetime constraints. Fisrt we need to figure out it physical motion equation and then set some constraints on it. Second, we need to know it's Jaccobian and Hessian of the functions. At last is to solving the linear system. As extension, first let us discuss the model- swing Stick, then mention some of the Luxo model.

5.1 Swing Stick

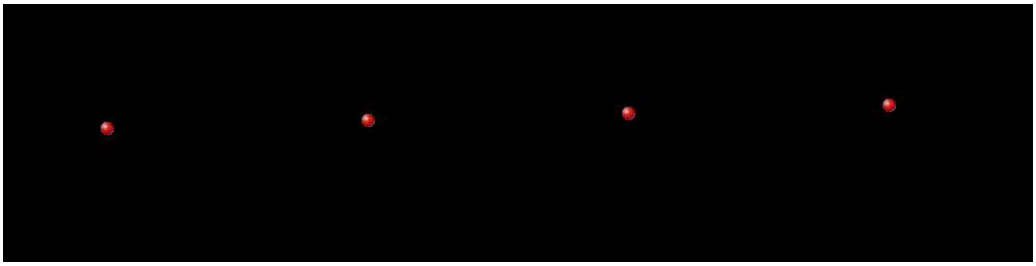
For the swing stick(one side of the stick is fixed on a point), we set the scalar independent variables Si as the angle. If the stick's length is L, and weight m, the stick's angle is α (the angle between stick and perpendicular direction). Then the motion equation is the relative to f, which perpendicular with the stick and the angle and the gravity of the stick. And the objective function R is same as the moving particle. Once we set the motion equation, we could figure out the linear system, then we could compute the angle's during the interval t0 and t1.

5.2Luxo

For Luxo, the situation is much more complicated. But as we know the motion equation for each segment of the lamp, depending on the angle of the adjacent two segments, the linear system equation still could be set up.



frame1-frame4



frame15-frame 30

Result

As the above picures showed, the start point is $y_0 = 10$ and $y_{n-1} = 200$. Genearate 30 frames during the 3 seconds, and use Slideshow Moviemaker to combine the avi file. The results show that spacetime methods are capableof producing realistic, complex and coordinated motion given only minimal kinematic constraints. The principle advantage of spacetime methods over simple keyframing is that they do much of the work that the animator would otherwise be required to do, and that only a skilled animator can do. Motions that would require highly detailed keyframe information may be sketched out at the level of "start here" and "stop there." This is a profoundly different and more economical means of control than conventional keyframing affords, an advantage that easily outweighs the greater mathematical complexity and computational cost of the method. Of the new opportunities for motion control, perhaps the most exciting is the selection of optimization criteria to affect the motion globally, an area we have only begun to explore. With a little thought, it is clear that a magic "right" criterion, whether based on smoothness, efficiency or some other principle, is unlikely to emerge and would in any case be undesirable. This is because the "optimal" way to perform a motion, as with any optimization, depends on what you're trying to do. The advantage of the method of spacetime constraints is the result is realistic, and under the control of the designer, and the problem is it is hard to generate the motion equation for the common complex practical models.

References

[1] Ronen Barzel and Alan H. Barr. A modeling system based on dynamic constaints. Computer Graphics, 22:179–188, 1988.
[2] Kurt Fleischer and Andrew Witkin. A modeling testbed. In Proc. Graphics Interface, pages 127–137, 1988.
[3] Phillip Gill, Walter Murray, and Margret Wright. Practical Optimization. Academic Press, New York, NY, 1981.
[4] Michael Girard and Anthony A. Maciejewski. Computational Modeling for the Computer Animation of Legged Figures. Proc. SIGGRAPH, pages 263–270, 1985.
[5] Herbert Goldstein. Classical Mechanics. Addision Wesley, Reading, MA, 1950.
[6] David Haumann. Topics in Physically Based Modeling, Course Notes, volume 16, chapter Modeling the Physical Behavior of Flexible Objects. SIGGRAPH, 1987.
[7] Paul Issacs and Michael Cohen. Controlling dynamic simulation with kinematic constraints, behavior functions and inverse dynamics. Computer Graphics, 21(4):215–224, July 1987. Proc. SIGGRAPH'87.
[8] Charless Klein and Ching-Hsiang Huang. Review of Pseudoinverse Control for Use With Kinematically Redundant Manipulators. IEEE Trans. SMC, 13(3), 1983.
[9] Johen Lasseter. Principles of traditionial animation applied to3Dcomputer animation. ComputerGraphics, 21(4):35–44, 1987.
[10] Pixar. Luxo, Jr., 1986. film. [11] W.H. Press, B.P. Flannery, S. A. Teukolsky, andW. T. Vetterling. Numerical Recipes in C. Cambridge University Press, Cambridge, England, 1988.
[12] Robert S. Stengel. Stochastic Optimal Control. John Wiley and Sons, New York, New York, 1986.
[13] Demetri Terzopoulos, John Platt, Alan Barr, and KurtFleischer. Elastically deformable models. *Computer Graphics*, 21(4), July 1987. Proc. SIGGRAPH '87.
[14] Jane Wilhelms and Brian Barsky. Using dynamic analysis to animate articulated bodies such as humans and robots. *Graphics Interface*, 1985.
[15] Andrew Witkin, Kurt Fleischer, and Alan Barr. Energy constraints on parameterized models. *Computer Graphics*, 21(4):225–232, July 1987.
[16]Andrew Witkin, Michael Kass. Spacetime Constraints.1988