Numerical Stability and Convergence Analysis of Geometric Constraint Enforcement in Dynamic Simulation Systems

Hongjun Jeon[1] Min-Hyung Choi[2] Min Hong[3]

[1]*Dept. of Electrical and Computer Engineering, Campus box 425*
University of Colorado, Boulder, Colorado 80309-0425, USA
jeon@colorado.edu
[2]*Dept. of Computer Science and Engineering, University of Colorado at Denver,*
Campus Box 109, PO Box 173364, Denver, CO 80217, USA
minchoi@acm.org
[3]*Bioinformatics, University of Colorado Health Sciences Center,*
4200 E. 9th avenue Campus Box C-245, Denver, CO 80262, USA
Min.Hong@UCHSC.edu

Abstract

The geometric constraint enforcement using Lagrange Multipliers is one of the popular methods to control the behavior and trajectories of dynamically simulated entities. Therefore, effective and efficient enforcement and proper integration of the geometric constraints is the key to a successful constraint management. This paper describes the formulation and integration of geometric constraints in a dynamic simulation and provides a guideline to choose a proper constraint enforcement method. In addition, the numerical stability and convergence analysis of two explicit and implicit constraint enforcement techniques are reported with respect to the step sizes of differential equations. Experiments show that our new implicit constraint enforcement technique demonstrates a superior stability over large time steps and fast system response compared to the explicit Baumgarte method. The implicit constraint enforcement method uses the future time step to estimate the correct magnitude of the constraint forces and doesn't require problem dependent stabilization parameters as a second order state feedback term.

1. Introduction

Due to the well-defined structures and the ease of programming, forward dynamics systems are widely used to simulate the dynamics of both rigid and deformable bodies in computer graphics, robotics, medical visualization and interactive applications. Dynamic simulations are essential components of many modern computer graphics and visualization applications because it produces highly realistic animations. However, the direct and precise control of the behavior and trajectories of objects are difficult because it's based on a passive simulation model where we set the initial conditions and external forces to compute the future motion. For example, a small adjustment of the input parameters can drastically affect the subsequent motion [10]. The inverse dynamics techniques could generate correct initial conditions from well-defined trajectories but usually they are very expensive to compute and sensitive to the numerical errors. Therefore, putting a proper set of geometric constraints over a dynamic system is widely used to control the behavior of dynamic system. By employing the geometric constraints as a control term, the overall scenario of a dynamic simulation can be designed in advance by defining the trajectories of a set of objects. The user interactions can also be interpreted as a control commands during a simulation, giving effective environments to direct and steer the simulation. In addition, geometric constraints have wide applications beyond the behavior control such as the formulation of robotic joints, joint limits, the definition of a valid variable scope, collision detection and contact resolution, etc. Despite this advantages, one of the main reasons that the geometric constraints are not used extensively is the error accumulation and numerical error drift over time when it's used in conjunction with the dynamic system. Several methods have been suggested for the stabilization of differential algebraic equations [1, 4, 18, 19]. Such methods often include constraint differentiation and problem stabilization. Because of its simplicity, one of the popular methods is Baumgarte constraint stabilization [2, 22] that employs a second order stabilization term, but the proper choice of coefficients dictates the stability and convergence. A physical explication of this method is that we are adding additional correction forces, proportional to the error in the velocity and position constraints, to counteract drift. The main difficulty of using Baumgarte stabilization is that it is not always easy to find an appropriate value for the coefficients.

Constraints have to conserve the geometric characteristics such as distance between the nodes, angle between the adjacent surface elements, during simulation. Constrained dynamics is concerned with making objects' behavior consistent with the forces of constraint. In a simple mass-spring case, the main idea of constrained dynamics is that our dynamic description includes not only discrete masses and forces, but restrictions on the way the masses are permitted to move. For example, we might constrain a mass to move along a specified curve, or require two masses to remain a specified distance apart. The problem of constrained dynamics is to make the mass obey Newton's laws, and at the same time obey the geometric constraints.

This paper shows a comparison between implicit and explicit constraint and provide a guideline to choose an appropriate constraint enforcement method. Geometric constraint enforcement technique shows good stability over large time steps and quick convergence. It is particularly effective to enforce a hard constraint over a large time step. It also presents dynamic model in state space, which describe differential equations, and state feedback to relocate the eigenvalues of a given system.

2. Background

Some of the most natural and graceful motion in computer graphics has been achieved by simulating the physical behavior of objects. Dynamic simulation of physical systems has become an important approach to computer graphics and computer animation. The modeling of physical systems often leads to partial or directly to ordinary differential equations. The solution of these equations usually is a major part of the total computational costs for a simulation or animations.

2.1 Dynamic Model in State Space

The idea of state space comes from the state-variable method of describing differential equations. In this method the differential equations describing a dynamic system are organized as a set of first-order differential equations in the vector-valued state of the system [6], and the solution is visualized as a trajectory of this state vector in space.

Use of Newton's law typically leads to second-order differential equations, that is, equations that contain the second derivative. Differential equations can be described by a set of simultaneous first-order differential equations [3, 12]. For the convenience of the description, we use a simple mass-spring-damper model but the constraint enforcement technique presented here can be applied to any dynamic system with holonomic constraints. We write our system of equation using $3\mathbf{N}$ generalized coordinates, $\boldsymbol{q}$, where $\mathbf{N}$ is the number of discrete masses, and the generalized coordinates are simply the Cartesian coordinates of the discrete masses.

$$\boldsymbol{q} = \begin{bmatrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & \ldots & x_N & y_N & z_N \end{bmatrix}^T \quad (1)$$

The physical system equation of movement of discrete masses with external control force, u called control reference, is

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{K}_d\dot{\boldsymbol{q}} + \boldsymbol{K}_s\boldsymbol{q} = \boldsymbol{K}_c u \quad (2)$$

The purpose of the control reference is to allow us to assign a set of pole locations for the closed-loop system that will correspond to satisfactory dynamic response in terms of rise time and other measures of transient response. Equation (2), the well known equation describing the damped oscillator [9], expresses Newton's second law of motion for discrete masses. $\boldsymbol{M}$ is a $3\mathbf{N} \times 3\mathbf{N}$ diagonal matrix containing discrete nodal masses, $\boldsymbol{K}_d$ is a $3\mathbf{N} \times 3\mathbf{N}$ matrix containing nodal damping coefficients, $\boldsymbol{K}_s$ is a $3\mathbf{N} \times 3\mathbf{N}$ matrix containing nodal spring stiffness constants, and $\boldsymbol{K}_c$ is a $3\mathbf{N} \times 1$ positive gain constant vector. Let us define state variables $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q} & \dot{\boldsymbol{q}} \end{bmatrix}^T$. Then we obtain

$$\dot{\boldsymbol{Q}} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\boldsymbol{M}^{-1}\boldsymbol{K}_s & -\boldsymbol{M}^{-1}\boldsymbol{K}_d \end{bmatrix}\boldsymbol{Q} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{M}^{-1}\boldsymbol{K}_c \end{bmatrix} u \quad (3)$$

$$\boldsymbol{P} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}\boldsymbol{Q} \quad (4)$$

Equation (3) is a state equation and equation (4) is an output equation for our system ($\boldsymbol{P}$ is the position vector). Equation (3) and (4) are in the standard form:

$$\dot{\boldsymbol{Q}} = \boldsymbol{A}\boldsymbol{Q} + \boldsymbol{B}u \quad (5)$$

$$\boldsymbol{P} = \boldsymbol{C}\boldsymbol{Q} \quad (6)$$

The entries of $\boldsymbol{Q}$ are called state variables. $\boldsymbol{Q}$ is a $6\mathbf{N} \times 1$ state vector, $\boldsymbol{A}$ is a $6\mathbf{N} \times 6\mathbf{N}$ system matrix, $\boldsymbol{B}$ is a $6\mathbf{N} \times 1$ input vector, and $\boldsymbol{C}$ is a $1 \times 6\mathbf{N}$ output vector.

2.2 State Feedback

If we were solving the differential equation exactly, this procedure would keep the particle exactly on the geometric constraint, provided we began with valid initial conditions. In practice, we know often the numerical solutions to ODE drifts. We must add an extra feedback term to prevent this numerical drift. The feedback force needs to be added in after the constraint

force calculation, or else the constraint force will dutifully cancel it.

One of the first applications of state space methods to linear systems was that of using feedback of the state variables to relocate the eigenvalues of a given system, which achieves better system response characteristics. The reason for adding feedback is to improve the system characteristics in some sense. Feedback control refers to an operation that, in the presence of disturbances, tends to reduce the difference between the output of a system and some reference input and that does so on the basis of this difference. Here only unpredictable disturbances are so specified, since predictable or known disturbances can always be compensated for within the system.

We consider the 6N-dimensional single-variable state equation (5) and (6). The first step in the state-space design method is to find the control reference as feedback of a linear combination of the state variables that is $u = -\boldsymbol{KQ}$. This equation tells us that the system has a constant matrix in the state-vector feedback path. For an 6Nth-order system there will be 6N feedback gains and since there are 6N roots of the system, it is possible that there are enough degrees of freedom to select arbitrarily any desired root location by choosing the proper values of $\boldsymbol{K}$.

Substituting the feedback reference given by $u = -\boldsymbol{KQ}$ into the system described by equation (5) yields

$$\dot{\boldsymbol{Q}} = \boldsymbol{AQ} - \boldsymbol{BKQ} = \left(\boldsymbol{A} - \boldsymbol{BK}\right)\boldsymbol{Q} \tag{7}$$

We wish to modify the given system by the use of state-variable feedback so as to obtain a new system with specified eigenvalues [11]. The characteristic equation of this closed-loop system is

$$\det\left[s\mathbf{I} - \left(\boldsymbol{A} - \boldsymbol{BK}\right)\right] = 0 \tag{8}$$

The control reference design then consists of picking the gain $\boldsymbol{K}$ so that the roots of equation (8) are in desirable locations. For now we assume that the desired locations are known, say $s = s_1, s_2, \cdots, s_n$. Then the corresponding desired characteristic equation is

$$\alpha_c(s) = (s - s_1)(s - s_2)\cdots(s - s_n) = 0 \tag{9}$$

Hence the required elements of $\boldsymbol{K}$ are obtained by matching coefficients in equation (8) and (9). This forces the system's characteristic equation to be identical to the desired characteristic equation and the closed-loop poles to be placed at the desired locations.

There are some advantages of feedback control. First, the system output can be made to follow or track the specified input function in an automatic fashion. Second, system performance is less sensitive to variations of parameter values. Third, system performance is less sensitive to unwanted disturbances. Fourth, use of feedback makes it easier to achieve the desired transient and steady-state response. The advantages of feedback are gained at the expense of certain disadvantages. First, the possibility of instability is introduced and stability becomes a major design concern. Actually, feedback can either stabilize or destabilize a system. Second, there is a loss of system gain, and additional stages of amplification may be required to compensate for this. Third, additional components of high precision are usually required to provide the feedback signals.

3. Constrained Dynamics

One of the most popular methods for integrating a geometric constraint into a dynamic system is to use Lagrange multipliers and constraint forces [13, 14, 20, 21]. The constraint-based formulation using Lagrange multipliers results in a mixed system of ordinary differential equations and algebraic expressions. Let $\boldsymbol{\Phi(q,t)}$ be the constraint vector made up of m components each representing an algebraic constraint. The constraint vector is represented mathematically as

$$\boldsymbol{\Phi}\left(\boldsymbol{q},t\right) = \left[\boldsymbol{\Phi}^1(\boldsymbol{q},t) \quad \boldsymbol{\Phi}^2(\boldsymbol{q},t) \;\ldots.\; \boldsymbol{\Phi}^m(\boldsymbol{q},t)\right]^T \tag{10}$$

where the $\boldsymbol{\Phi}^i$ are the individual scalar algebraic constraint equations. The total force acting on the system is the sum of the external force and the constraint forces. We write the system of equations

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{\Phi}_q^T\boldsymbol{\lambda} = \boldsymbol{F}^A \tag{11}$$

$$\boldsymbol{\Phi}\left(\boldsymbol{q},t\right) = \mathbf{0} \tag{12}$$

where $\boldsymbol{F}^A$ are applied, gravitational and spring forces acting on the discrete masses, $\boldsymbol{\lambda}$ is a $\mathbf{M} \times 1$ vector containing the Lagrange multipliers and $\boldsymbol{\Phi}_q$ is the $\mathbf{M} \times 3\mathbf{N}$ jacobian matrix.

3.1 Explicit Constraint Enforcement

Baumgarte [2, 7] described a technique in which the differentiated constraint equations are augmented with terms that are analogous to feedback applied to a classical second order system. Let me explain more in detail. Equation (12) is an implicit function for the constraint. If $\boldsymbol{q}$ is a legal position, then legal velocities and legal accelerations are all those that satisfy

$$\dot{\Phi} = \mathbf{0} \quad \text{(legal velocity)} \tag{13}$$

$$\ddot{\Phi} = \mathbf{0} \quad \text{(legal acceleration)} \tag{14}$$

Let us define state variables $\Theta = \begin{bmatrix} \Phi & \dot{\Phi} \end{bmatrix}^T$. Then using the state feedback gain $K = \begin{bmatrix} \beta^2 & 2\alpha \end{bmatrix}$ as we describe section 2.2, we obtain

$$\dot{\Theta} = \frac{d}{dt}\begin{bmatrix} \Phi \\ \dot{\Phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\beta^2 & -2\alpha \end{bmatrix} \cdot \begin{bmatrix} \Phi \\ \dot{\Phi} \end{bmatrix} \tag{15}$$

From equation (15), the constraint equation with state feedback can be written by

$$\ddot{\Phi} + 2\alpha\dot{\Phi} + \beta^2\Phi = \mathbf{0} \tag{16}$$

The parameters α and β determine the character of the transient as $\Phi \to \mathbf{0}$. Equation (16) may be expressed in a more convenient form by differentiating the constraint vector as follows

$$\dot{\Phi}(q,t) = \frac{d\Phi(q,t)}{dt} = \frac{\partial\Phi(q,t)}{\partial t} + \frac{\partial\Phi(q,t)}{\partial q}\frac{dq}{dt} \tag{17}$$

or

$$\dot{\Phi}(q,t) = \Phi_t + \Phi_q \dot{q} \tag{18}$$

Continuing in this fashion we arrive at the result

$$\Phi_q \ddot{q} = \hat{\gamma} \tag{19}$$

where

$$\begin{aligned} \hat{\gamma} = &-\left(\Phi_q \dot{q}\right)_q \dot{q} - 2\Phi_{qt}\dot{q} - \Phi_{tt} \\ &- 2\alpha\left(\Phi_q \dot{q} + \Phi_t\right) - \beta^2\Phi \end{aligned} \tag{20}$$

and the subscripts q and t indicate partial differentiation with respect to q and t, respectively. $\ddot{q}$ may be isolated from equation (11)

$$\ddot{q} = M^{-1}F^A - M^{-1}\Phi_q^T\lambda \tag{21}$$

this may be substituted into equation (20) to obtain the linear system that must be solved for the Lagrange multipliers

$$\begin{aligned} &\Phi_q(q,t)M^{-1}\Phi_q^T(q,t)\lambda \\ &\quad = -\hat{\gamma} + \Phi_q(q,t)\,M^{-1}F^A(q,t) \end{aligned} \tag{22}$$

This equation along with equation (11) can be solved with a variety of explicit methods. Solution with explicit methods constrains the time step to be within a stability limit. An additional difficulty is that selection of the parameters α and β in such a way as to force satisfaction of the constraint to a lower error tolerance decreases the time increment allowed from stability considerations. The detailed analysis is described in section 4.

3.2 Implicit Constraint Enforcement

Choi et al. proposed implicit constraint enforcement [5, 8]. The implicit method [15, 16, 17] is a backward first order and is implemented the following way. The equation of motion (equation (11)) along with the kinematic relationship between q and $\dot{q}$ are discretized as

$$\begin{aligned} \dot{q}(t+\Delta t) = \dot{q}(t) &- \Delta t M^{-1}\Phi_q^T(q,t)\lambda \\ &+ \Delta t M^{-1}F^A(q,t) \end{aligned} \tag{23}$$

$$q(t+\Delta t) = q(t) + \Delta t\,\dot{q}(t+\Delta t) \tag{24}$$

The constraint equations written at new time thus they are treated implicitly

$$\Phi(q(t+\Delta t),t+\Delta t) = \mathbf{0} \tag{25}$$

Equation (25) is now approximated using a truncated, first-order Taylor series

$$\begin{aligned} \Phi(q,t) + \Phi_q(q,t)(q(t+\Delta t) - q(t)) \\ + \Phi_t(q,t)\Delta t = \mathbf{0} \end{aligned} \tag{26}$$

Again note that the subscripts q and t indicate partial differentiation with respect to q and t, respectively. Substituting $\dot{q}(t+\Delta t)$ from equation (23) into equation (24) we obtain

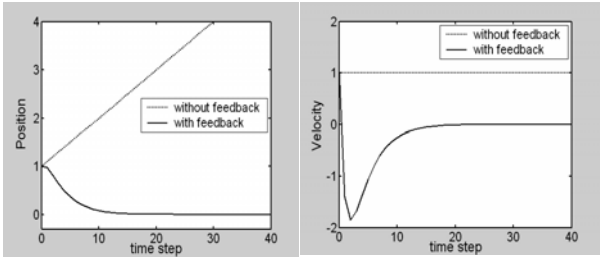
$$\begin{aligned} q(t+\Delta t) = q(t) &+ \Delta t\,\dot{q}(t) \\ &- \Delta t\left\{\Delta t M^{-1}\Phi_q^T(q,t)\lambda + \Delta t M^{-1}F^A(q,t)\right\} \end{aligned} \tag{27}$$

Substitution of this result into equation (26) thus eliminating $q(t+\Delta t)$ results in the following linear system with λ the remaining unknown.

$$\begin{aligned} \Phi_q(q,t)M^{-1}\Phi_q^T(q,t)\lambda = &\frac{1}{\Delta t^2}\Phi(q,t) + \frac{1}{\Delta t}\Phi_t(q,t) \\ &+ \Phi_q(q,t)\left(\frac{1}{\Delta t}\dot{q}(t) + M^{-1}F^A(q,t)\right) \end{aligned} \tag{28}$$

Note that the coefficient matrix for this implicit method is the same as the coefficient matrix for the Baumgarte method. This system is solved for the Lagrange multipliers then equations (23) and (24) are used to update the generalized coordinates and velocities.

This constraint-based scheme has several advantages. The scheme is simple, the constraints are satisfied without the time lag and oscillations associated with the Baumgarte method. Perhaps the most important advantage of this scheme is that the constraint enforcement does not place a stability limit on the time step while factoring the same linear system as the Baumgarte method requires. Just as with the Baumgarte method, constraints are easily added or removed from the overall dynamic system enabling the use of adaptive constraint activation.



(a) (b)

Figure 1: We use $\alpha = \beta = 4$ so state feedback gain $K = \begin{bmatrix} 16 & 8 \end{bmatrix}$. Initial position $\Phi(0) = 1$ (a) and initial velocity $\dot{\Phi}(0) = 1$ (b)

4. Stability Analysis of Explicit Constraint

There are two main issues in applying constraint forces to deformable object simulations. First, constraints might not be met initially ($\Phi(0) \neq 0$, $\dot{\Phi}(0) \neq 0$). Second, numerical errors could accumulate over time especially if we're using a large time step. Baumgarte uses a second order state feedback term to handle both problems. If initial values of constraints are not on legal positions and legal velocities, the system might blow up. Figure 1 shows that state feedback puts the initial state, which is not meet legal positions and legal velocities, back on the legal states.

As we describe in section 3.1, in stead of solving for $\ddot{\Phi} = 0$, we solve for

$$\ddot{\Phi} = -2\alpha\dot{\Phi} - \beta^2\Phi \qquad (29)$$

where α and β are stabilization factors. Equation (29) can be reduced to first order system which is the state space equation (15). The differential equation (15) is approximated by a difference equation with the explicit Euler method and a fixed step size *h*:

$$Y(t+h) \approx Y(t) + h\dot{Y}(t) = (\mathrm{I} + hA) \cdot Y(t) \qquad (30)$$

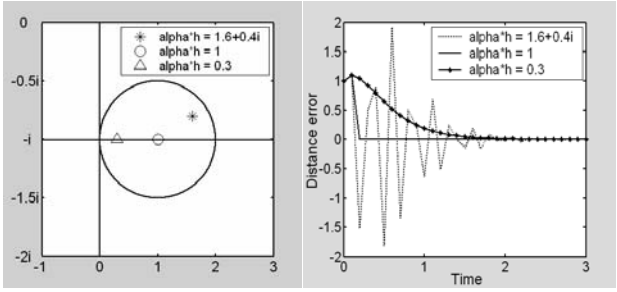
The *n*-th point of the numerical solution is given by:

$$\begin{aligned} Y(n+1) &= (\mathrm{I} + hA)^n Y_0 \\ &= \begin{bmatrix} 1 & h \\ -\beta^2 h & 1 - 2\alpha h \end{bmatrix}^n Y_0 = B^n Y_0 \end{aligned} \qquad (31)$$

with initial condition $Y_0 = \begin{bmatrix} \Phi(0) \\ \dot{\Phi}(0) \end{bmatrix} = \begin{bmatrix} y_0 \\ \dot{y}_0 \end{bmatrix}$. This is stable if and only if the magnitude of the eigenvalues of B are less than 1. If we assume $\beta = \alpha$ as a simple case, the eigenvalues satisfy:

$$\det(s\mathrm{I} - B) = \left(s - (1 - \alpha h)\right)^2 = 0 \qquad (32)$$

and the stability region is $|1 - \alpha h| < 1$. We define the stability region as shown in figure 1. For stability, αh must be inside the circle in the figure 2(a). It is important to note that different system behavior is expected with different pole locations within stability region. If it's very close to the center, it converges well without oscillation. The beta term determines the initial slope of the curve and high damping alpha term mitigates the oscillation. Note that if we want fast convergence and no oscillation, we can use a big value for beta but it makes the system unstable for a big time step, requiring a small time step. The Baumgarte method uses ordinary differential equations to force the constraints to zero. This necessarily results in time constants associated with the differential equations. Implicit constraint enforcement does not have time constants because we're solving algebraic equations directly within the truncated first-order Talyor series error bound. Therefore, there is no stability limit with respect to the time step for this approach. Another important advantage of implicit constraint enforcement is insensitivity to the initial conditions. By the time of constraint activation, initial conditions may not be perfectly met or too costly to guarantee at every instance. This is particularly important for many actual implementations and applications since the preparation for the correct initial conditions could be very time consuming and tedious.



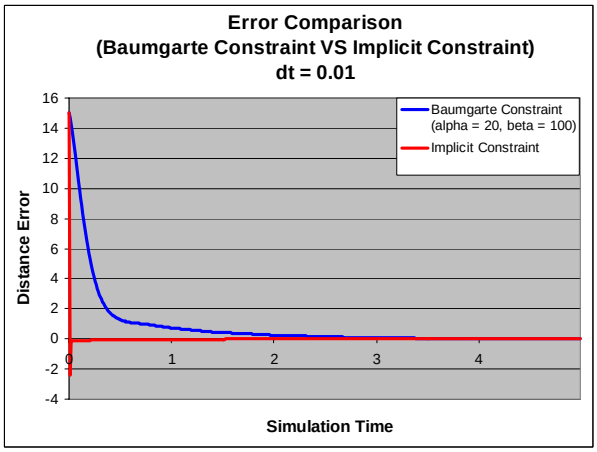
(a) (b)

Figure 2: Selected choices of parameters in the stability region (a) and their distance error over time (b)

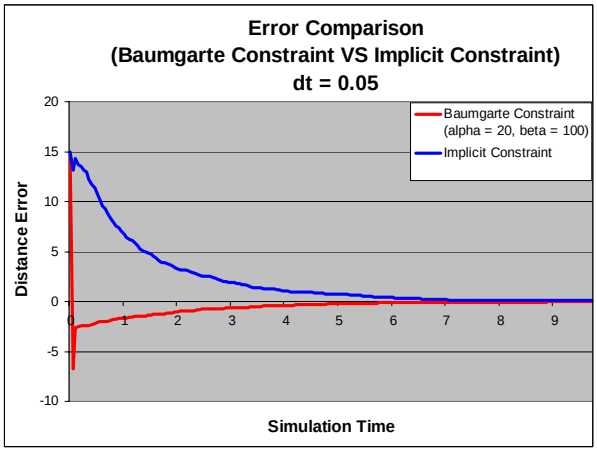
5. Comparison between Implicit and Explicit Constraint

To prove the efficiency of implicit constraint, we measure the constraint distance errors using a simple bead-on-a-wire simulation where a particle is geometrically constrained for maintaining a desired distance from the center and initially we set up the illegal position and velocity to simulate. We arrange a particle has 65 lengths from the center of circle but desirable distance is 50 and apply the initially velocity 10% away from tangential direction with relatively big velocity (50 to off tangential direction). The particle should follow a circle orbit under relatively difficult and initially illegal situation. Figure 3 illustrates the distance error comparison with different time step sizes (dt = 0.01 (a), dt = 0.05 (b), and dt = 0.1(c)). Although we select ideal alpha and beta term for feedback of Baumgarte constraint approach, figure 3 (a) and (b) show implicit constraint enforcement is much quickly and efficiently converge to a solution than Baumgarte method. For dt = 0.1 (c), Baumgarte method blows up the simulation system but implicit constraint enforcement still converges to solution and it allows time step size until 2.0.

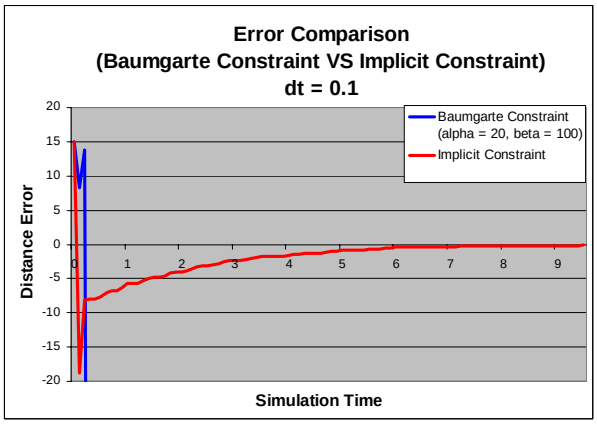
Our experimental results show that implicit constraint is superior to Baumgarte constraint in terms of convergence and stability under illegal position and velocity. The immanent artificial damping effect of implicit constraint enforcement causes fast convergence to the solution. Asymptotic computational complexities are same for both methods but implicit constraint enforcement has additional constant factor. This small disadvantage can be easily overcome by employing a bigger time step. In addition, we can utilize big time step size for implicit constraint enforcement without any cost for finding correct alpha and beta feedback coefficients.



(a)



(b)



(c)

Figure 3: Constraint distance errors with different time step size

7. Conclusion

The formulation of implicit constraint enforcement, a stability analysis, and integration of geometric constraints in a dynamic simulation has been described. To control the behavior and trajectories of dynamically simulated entities we have suggested the geometric constraint enforcement using Lagrange Multipliers. This paper has reported numerical stability and convergence analysis of geometric constraint enforcement technique that is stable over a large time step and doesn't require problem dependent feedback terms. Combined with an adaptive constraint activation scheme, the implicit constraint enforcement method can substantially enhance the controllability of the dynamic simulation of deformable objects.

Our new implicit constraint enforcement method used the future time step to estimate the correct magnitude of the constraint forces, resulting in better stability over bigger time steps. Using an implicit constraint method is better for the minimization of numerical drift and error accumulation, instead of using second order feedback term. It is easy to implement and integrate into the existing explicit ODE dynamic system.

Downside of this method is that it's still the first order backward Euler for the constraint enforcement so if we need more precise constraint handling, the higher order system is desirable. Since we're comparing two different constraint enforcement schemes under the framework of Lagrange Multipliers method, the inherent limitations of Lagrange Multipliers method still remains. As the number of constraint grows, solving the linear system to convert geometric constraints into constraint forces grows as well.

8. References

[1] David Baraff and Andrew Witkin, "Large steps in cloth simulation", *SIGGRAPH 98 Conference Proceedings*, pages 43-54, July 1998.

[2] Baumgarte J, "Stabilization of Constraints and Integrals of Motion in Dynamical Systems", *Computer Methods in Applied Mechanics*, 1:1-36, 1972.

[3] Chi-Tsong Chen, *Linear System Theory and Design*, Third edition, Oxford university press, 1999.

[4] Michael Hauth, "Numerical Technique for Cloth Simulation", *SIGGRAPH 2003 Course #29*, July, 2003

[5] Choi Min-Hyung, Hong Min, and Samuel Welch, "Implicit Constraint Enforcement for Stable and Effective Control of Cloth Behavior", *A technical report form Univ. of Colorado at Denver*, Computer Science and Engineering Department TR-03-01, 2004.

[6] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini, *Feedback Control of Dynamic Systems*, Third edition, Addison-Wesley, 1995.

[7] Hang E. J., *Computer Aided Kinematics and Dynamics of Mechanical Systems*, Volume I: Basic Method, Allyn and Bacon, 1989.

[8] Hong Min, Choi Min-Hyung, and Ravi Yelluripati, "Intuitive Control of Deformable Object Simulation using Geometric Constraints", *Conference on Imaging Science, Systems, and Technology (CISST)*, 2003.

[9] Jer-Nan Juang, *Applied System Identification*, Prentice Hall, 1994.

[10] Jovan Popovic, Steven M. Seitz, Michael Erdmann, Zoran Popovic and Andre Witkin. "Interactive Manipulation of Rigid Body Simulations", *In Computer Graphics (Proceedings of SIGGRAPH 2000), ACM SIGGRAPH, Annual Conference Series*, 209-217, 2000.

[11] Katsuhiko Ogata, *Modern Control Engineering*, Third edition, Prentice Hall, 1997.

[12] Thomas Kailath, *Linear Systems*, Prentice Hall, 1980.

[13] G. Strang, *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, MA, 1986.

[14] D. Terzopoulos and H. Qin, "Dynamics nurbs with geometric constraints for interactive sculpting", *ACM Transactions on Graphics*, 13:103-136, 1994.

[15] D. Terzopoulos and K. Fleischer, "Deformable Models", *Visual Computer*, 4:306-331, 1988

[16]] D. Terzopoulos and K. Fleischer, "Modeling inelastic deformation: Viscoelasticity, plasticity, fracture", *In Computer Graphics (Proc. SIGGRAPH)*, volume 22, pages 269-278, ACM, August 1988.

[17] D. Terzopoulos, J.C. Platt, and A.H. Barr, "Elastically deformable models", *Computer Graphics (Proc. SIGGRAPH)*, 21:205-214, 1987.

[18] Uri M. Ascher, Hongsheng Shin, and Sebastian Reich, "Stabilization of DAEs and invariant manifolds", *Numerische Mathematik, Springer Verlag*, Numer. Math. 67: 131-149, 1994.

[19] Uri M. Ascher and L. Petzold, "Stability of computational methods for constrained dynamics systems", *SIAM J. Numer. Anal.* 28, 1097-1120, 1993.

[20] John C. Platt and Alan H. Barr, " Constraint methods for flexible models", *Computer Graphics*, Volume 22: 279-288, 1988.

[21] Andrew Witkin, Michael Gleicher, and William Welch, "Interactive Dynamics", *In Proceedings 1990 Symposium on Interactive 3d Graphics*, volume 24, pages 11–21, March 1990.

[22] Michael B. Cline and Dinesh K. Pai, "Post-stabilization for rigid body simulation with contact and constraints", *In Proceeding IEEE International Conference on Robotics and Automation*, 2003.