Continuous Contact Simulation for Smooth Surfaces

PAUL G. KRY and DINESH K. PAI
University of British Columbia

Dynamics simulation of smooth surfaced rigid bodies in contact is a critical problem in physically based animation and interactive virtual environments. We describe a technique which uses reduced coordinates to evolve a single continuous contact between smooth piece-wise parametric surfaces. The incorporation of friction into our algorithm is straightforward. The dynamics equations, though slightly more complex due to the reduced coordinate formulation, can be integrated easily using explicit integrators without the need for constraint stabilization. Reduced coordinates confine integration errors to the constraint manifold, thereby permitting a wide choice of step sizes with visually acceptable results. We demonstrate these results using Loop Subdivision surfaces with parametric evaluation.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Physically based modeling*; *Splines*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*

General Terms: Algorithms

1. INTRODUCTION

One of the difficult aspects of simulating rigid bodies is the case where there is contact. In this case the motion of the bodies must be constrained so that they do not interpenetrate. The combination of smooth surfaces, friction, and multiple contacts can make this a challenging problem.

Dynamics simulation with contact is difficult since the contact points impose constraints on motion which must be maintained during the simulation. These contact constraints are intimately related to the representation of the local geometry at the contact points. Polyhedral surfaces impose simple constraints and are often used to approximate smooth shapes, but can cause either large errors in motion, or require dense meshes. An attractive alternative is to directly work with smooth surfaces (e.g., [Baraff 1990; Nelson and Cohen 1998]).

We can conceptually divide the computations of handling contact constraints into two types:

—The *contact transition problem* is to determine possible discontinuous changes in contact, for instance, caused by the collision of two objects (or different parts

This work was supported in part by scholarships from NSERC and BC-ASI, and grants from NSERC and IRIS. Blue Moon Rendering Tools were used for Figures 1 and 6.
Authors' address: Department of Computer Science, University of British Columbia, 201-2366 Main Mall, Vancouver, BC, V6T 1Z4, e-mail: {pgkry|pai}@cs.ubc.ca.

Fig. 1. Example frames from some of our simulations.

of objects that are already in contact). Algorithms for this problem must be "global," that is, they must be able to detect contacts anywhere on the two bodies. This problem is essentially a two part problem as it involves both detecting collision (or computing a separation distance) and computing a collision response. This approach has received the most attention in graphics, computational geometry, and robotics (e.g., [Gilbert et al. 1988; Baraff 1990; Lin and Canny 1991; Mirtich and Canny 1995]).

—The *contact evolution problem* is to determine how a contact point evolves continuously over time, given the external forces on and dynamics of contacting bodies. Algorithms for this problem are "local," that is, they determine how the contact changes if no global contact transitions occur. To be useful, they still require a global algorithm to detect collisions. These algorithms come into effect when the collision response determines that the surfaces are in continuous contact (e.g., [Goyal 1989; Anitescu et al. 1996; Nelson et al. 1999]).

During typical contact simulations, a large proportion of time is spent in contact evolution computations, with a few contact transitions. For instance, Figure 1 shows examples of Loop subdivision surfaces in continuous contact, simulated with our system. For such motions, it pays to solve the contact evolution problem efficiently.

In principle, contact evolution problems can be handled in the same way as contact transition problems; indeed this the most common approach (e.g., [Baraff 1990; Mirtich and Canny 1995]). Once the contact forces are determined, the motion of the objects in 3D is computed by numerical integration. However, since only second order (acceleration) constraints are imposed, the two bodies can interpenetrate or separate after each integration step. The constraint then has to be reimposed (or *stabilized*) by an additional step after integration, which is similar to the contact transition problem. While this has the advantage of reusing some software used for contact transitions, it can be inefficient. Stabilizing the constraint usually requires finding extreme points between the separated or interpenetrating surfaces to compute constraint gradients. This can be awkward for smooth surfaces, particularly parametric surfaces, though it is possible to use iterative methods and coherence to improve performance [Baraff 1990].

This paper investigates the dynamics of piece-wise smooth surfaces in single surface-surface continuous contact. Our solution to this case of the contact evolution problem formulates an ordinary differential equation (ODE) in the coordinates of the contact patches. By parameterizing the system in reduced coordinates, the

contact constraint is automatically satisfied and therefore we avoid constraint stabilization. There is a price, however, as the reduced coordinate formulation is more complex and can take just as long if not longer to integrate per step. The advantage is that a large range of step sizes can be taken with visually acceptable results, since the integration error conforms to the constraint.

It is not difficult to extend our method to multi-body systems where the contacts exist either in a chain or tree structure, since both the composite rigid body method and articulated body method can be applied directly (our contacts are essentially generalized joints in a kinematic system, see [Pai et al. 2000]). Kinematic loops can also be treated, such as with loop closure constraints [Featherstone 1987]; however, in this case, the extra constraints are likely to prevent an efficient reduction of the system to an ODE (so stabilization of the extra constraints becomes necessary). These more complicated systems also require a more thorough treatment of breaking contact, which is a tricky problem in the presence of multiple contacts and friction (though solvable as a linear complementarity problem [Anitescu and Potra 1997]). In addition, for multiple contacts between just two rigid bodies, the advantage of the reduced coordinate parameterization is not clear. But this problem is more general, and different from the problem wherein our method has the greatest benefit, specifically, smooth surfaces in contact exhibiting long periods of rolling and sliding. For example, simulations including automatic synthesis of contact sounds benefit from this direct treatment of rolling and sliding since the smooth and contiguous contact forces do not produce audible artifacts [van den Doel et al. 2001].

Reduced coordinate formulations such as the one we use here are well studied [Featherstone 1987; Rabier and Rheinboldt 1993]. While Baraff [1996] provides good motivation for the use of Lagrange multipliers in lieu of reduced coordinates, the parameterization of smooth surface contact is not as hard as he suggests. Our parameterization and kinematics equations, similar to others, are inspired by the elegant derivation of contact kinematics equations for orthogonal nets shown by Montana [1988]. Examples of other derivations are found in Nelson et al. [1999], which considers equations more suitable for haptics, and in Anitescu et al. [1996], which includes equations for curve-surface and curve-curve contact. Our contact evolution algorithm is also inspired by the analysis of rigid body methods in Ascher et al. [1997].

Subdivision surfaces and free-form parametric surfaces such as NURBS are popular surface representations for modeling and design. Because of their popularity, simulating the contact dynamics of these kinds of surfaces is important. Our contact evolution technique, being ideal for piece-wise parametric and parameterizable surfaces (such as Loop, Catmull-Clark, and other stationary subdivision schemes for which parameterizations are known), provides a useful component for simulations involving smooth surface contact.

2. PRELIMINARIES

In this paper we use a notation for spatial dynamics which is similar to others [Featherstone 1987; Jain 1991; Montana 1988; Murray et al. 1994; Pai et al. 2000].

The motion of a rigid body $i = 1, ..., n$ is described using a reference frame labeled

i attached to the body. Unless otherwise noted, we will let this body reference frame be located at the center of mass and with its axes aligned with the principal axes of rotation.

The homogeneous coordinates of frame i with respect to another frame j is given by the 4×4 matrix ${}^j_i\mathsf{E}$. We will always use leading subscripts and superscripts to indicate frames. The homogeneous coordinates of a three dimensional vector x in frame i are denoted ${}^ix$. The homogeneous coordinates of this vector in frame j are given by left multiplying by ${}^j_i\mathsf{E}$.

The spatial velocity $\phi(j,i)$ describes the relative motion of frame i with respect to frame j. In (non-homogeneous) coordinates of frame i it is given by the 6×1 matrix ${}^i\phi(j,i) = ({}^i\omega^T,\ {}^iv^T)^T$, where ω is the angular velocity and v is the linear velocity of the point at the origin of frame i. Spatial forces, called wrenches, are represented as $f = (f_r^T, f_t^T)^T$ where f_r is the (rotational) torque and f_t is the (translational) force.

It is useful to note that the spatial velocity vector parameterizes a twist that can be written in homogeneous coordinates as the 4×4 matrix,

$$\mathcal{O}\phi = \begin{pmatrix} [\omega] & v \\ 0 & 0 \end{pmatrix}. \tag{1}$$

Here, $[\omega]$ denotes the skew symmetric 3×3 matrix equivalent to the cross product $\omega\times$, that is,

$$[\omega] = \begin{pmatrix} 0 & -\omega^z & \omega^y \\ \omega^z & 0 & -\omega^x \\ -\omega^y & \omega^x & 0 \end{pmatrix}, \tag{2}$$

and $\mathcal{O}$ denotes the linear expansion operation which converts the 6×1 matrix into a 4×4 matrix representation. We will let $\mathcal{V}$ denote the linear operation which performs extraction of the 6 parameters, thus,

$$\phi = \begin{pmatrix} \omega \\ v \end{pmatrix} = \mathcal{V}\begin{pmatrix} [\omega] & v \\ 0 & 0 \end{pmatrix}. \tag{3}$$

If $\mathcal{V}$ is applied to a matrix whose upper 3×3 matrix is not skew symmetric then we can assume that $\mathcal{V}$ acts as a projection onto $so(3)$.

The time derivative of the change of coordinates from frame i to frame j is a twist when written in coordinates of frame i, that is,

$${}^i\phi(j,i) = \mathcal{V}({}^i_j\mathsf{E}\ {}^j_i\dot{\mathsf{E}}). \tag{4}$$

Spatial velocities and spatial wrenches transform according to the Adjoint transformation ${}^j_i\mathsf{Ad}$. Spatial velocities being contravariant quantities transform by left multiplying, ${}^j\phi = {}^j_i\mathsf{Ad}\ {}^i\phi$. Because we write spatial wrenches as column vectors, these covariant quantities are transformed by left multiplying with the inverse transpose, ${}^jf = {}^i_j\mathsf{Ad}^T\ {}^if$. The 6×6 Adjoint matrix is defined as,

$${}^j_i\mathsf{Ad} = \begin{pmatrix} \Theta & 0 \\ [p]\Theta & \Theta \end{pmatrix}, \quad \text{where} \quad {}^j_i\mathsf{E} = \begin{pmatrix} \Theta & p \\ 0 & 1 \end{pmatrix}.$$

Here Θ is a 3×3 rotation matrix and p is a 3×1 displacement.

Lastly, we define the spatial cross product of $\phi = \left(\omega^T, \mathsf{v}^T \right)^T$ as the linear operator with coordinate matrix

$$[\phi] = \begin{pmatrix} [\omega] & 0 \\ [v] & [\omega] \end{pmatrix}.$$

3. CONTACT EVOLUTION FORMULATION

We consider rigid bodies whose boundaries are defined by a collection of parametric surface patches. Although the surface patches will typically be polynomial or rational polynomial functions, this does not need to be the case. We only require the ability to evaluate the function and its derivatives at a given point.

A single point of contact between two bodies will involve one patch from each body. We ignore that adjacent patches will overlap along their boundary curves as it is the location of the contact that we want to describe. If the contact lies on a boundary then we can choose either of the patches sharing the boundary to describe the contact location.

3.1 Contact Coordinates

Suppose body 1 and body 2 are in contact. Let the shape of the contacting patch on body 1 be described by the function ${}^{1}\mathsf{c} : (s,t) \to \mathbb{R}^3$, and on body 2 by ${}^{2}\mathsf{d} : (u,v) \to \mathbb{R}^3$. Note that the functions have a preceding superscript denoting that they define the surface shape in the coordinates of their respective body's reference frame. We drop this superscript when the coordinate frame is clear from context.

We use the notation $\mathsf{c}_{,s} \stackrel{\text{def}}{=} \frac{\partial \mathsf{c}}{\partial s}$. The orthonormal contact frame with coordinates (s,t) can be defined in frame *1*, the body frame, as the coordinate frame with origin at $\mathsf{c}(s,t)$ and coordinate axes[1]

$$\mathsf{x} = \frac{\mathsf{c}_{,s}}{\|\mathsf{c}_{,s}\|}, \quad \mathsf{y} = \frac{(\mathsf{c}_{,s} \cdot \mathsf{c}_{,s})\mathsf{c}_{,t} - (\mathsf{c}_{,s} \cdot \mathsf{c}_{,t})\mathsf{c}_{,s}}{\|(\mathsf{c}_{,s} \cdot \mathsf{c}_{,s})\mathsf{c}_{,t} - (\mathsf{c}_{,s} \cdot \mathsf{c}_{,t})\mathsf{c}_{,s}\|}, \quad \mathsf{z} = \frac{\mathsf{c}_{,s} \times \mathsf{c}_{,t}}{\|\mathsf{c}_{,s} \times \mathsf{c}_{,t}\|}. \tag{5}$$

Coordinates are in frame *1* (superscripts were omitted). The vectors x, y and z form orthonormal axes provided the surface function is regular (that is $\mathsf{c}_{,s} \times \mathsf{c}_{,t} \neq 0$ for all points in the domain). Thus the homogeneous coordinates of this contact frame with respect to frame *1* are given by

$${}^{1}_{1c}\mathsf{E} = \begin{pmatrix} {}^{1}\mathsf{x} & {}^{1}\mathsf{y} & {}^{1}\mathsf{z} & {}^{1}\mathsf{c} \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

This matrix is a function of s and t. The contact frame on body 2 is defined in the same way yielding the matrix ${}^{2}_{2c}\mathsf{E}$, a function of u and v in the domain of the patch $\mathsf{d}(u,v)$. Note that the c in the frame label *2c* stands for *contact* as opposed to representing patch c of body 1.

Because the contact frames are orthonormal and occur at the same location in

[1]Note that the given expression for y is less expensive than normalizing $\mathsf{c}_{,s} \times \mathsf{c}_{,t} \times \mathsf{c}_{,s}$ which is important since derivatives of this expression are needed in Section 3.2

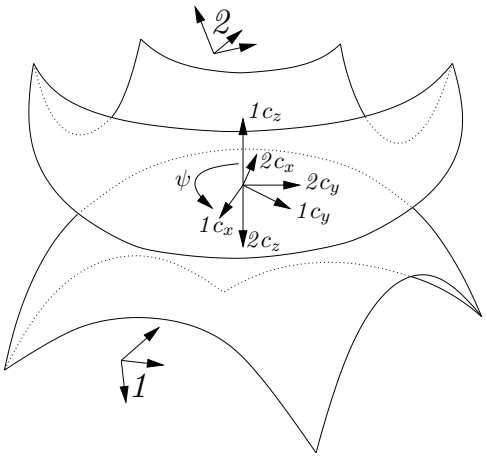

Fig. 2. Contact frames and contact coordinate ψ

space, they can be easily aligned by a rotation matrix (see Figure 2).

$$ {}^{1c}_{2c}\mathsf{E} = \begin{pmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ -\sin\psi & -\cos\psi & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \stackrel{\text{def}}{=} \begin{pmatrix} R_\psi & 0 \\ 0 & 1 \end{pmatrix} \tag{7} $$

The matrix ${}^{1c}_{2c}\mathsf{E}$ is idempotent and R_ψ is its upper 3×3 rotation matrix.

We can describe any contact configuration between patches c and d of bodies 1 and 2 by the 2-dimensional location of the contact in the domain of each patch along with the angle of rotation ψ. Assembled in a column vector, we call these 5 independent variables the *contact coordinates* and we denote it q; that is,

$$ q = \begin{pmatrix} s & t & u & v & \psi \end{pmatrix}^T . \tag{8} $$

3.2 Kinematics Equations

The contact kinematics equations relate the relative motion between two smooth contacting bodies to a change in the contact coordinates. Because the contact coordinates are of reduced dimension, this system of equations is a linear transformation (a function of the contact coordinates) from the 5 dimensional space of contact coordinate velocities into a 5 dimensional subspace of the 6 dimensional space of spatial velocities. We give our own derivation here, suitable for any type of parametric surface.

As illustrated in Figure 3 we compute a matrix representing a change in coordinates from frame *1* to frame *2*. This composition is written as

$$ {}^{2}_{1}\mathsf{E} = {}^{2}_{2c}\mathsf{E}\ {}^{2c}_{1c}\mathsf{E}\ {}^{1c}_{1}\mathsf{E}. \tag{9} $$

The three matrices on the right hand side are functions of the contact coordinates. They are defined by Equations 6 and 7. Recall[2] that the relative spatial velocity

[2]See, for example, Murray et al. [1994].

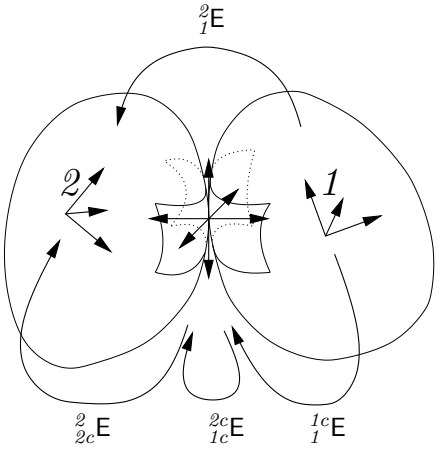

Fig. 3. Change of reference frames

can be defined by

$$ {}^{1}\phi(2,1) = \mathcal{V}({}^{1}_{2}\mathsf{E}\ {}^{2}_{1}\dot{\mathsf{E}}). \tag{10} $$

Taking the time derivative using the chain rule gives

$$ {}^{1}\phi(2,1) = \mathcal{V}\sum_{j=1}^{5} {}^{1}_{2}\mathsf{E}\ \ {}^{2}_{1}\mathsf{E}_{,q_j}\ \dot{q_j}. \tag{11} $$

Since the contact coordinates can change independently of each other, each 4×4 matrix in the sum will be a *twist* and can have the extraction operator applied directly, hence

$$ {}^{1}\phi(2,1) = \sum_{j=1}^{5} {}^{1}H_j\ \dot{q_j} \tag{12} $$

$$ \text{where} \quad {}^{1}H_j = \mathcal{V}({}^{1}_{2}\mathsf{E}\ \ {}^{2}_{1}\mathsf{E}_{,q_j}). \tag{13} $$

The column vector ${}^{1}H_j$ tells us the contribution of a change in the j^{th} component of the contact coordinates to the relative spatial velocity of the two objects. Letting ${}^{1}H$ be a matrix whose columns are ${}^{1}H_j,\ j = 1 \ldots 5$, we can write Equation 12 in matrix notation as

$$ {}^{1}\phi(2,1) = {}^{1}H\dot{q}. \tag{14} $$

This 6×5 matrix H relates coordinate velocities to relative spatial velocities. It can be transformed to any other convenient frame by left multiplying with an appropriate adjoint.

When written in a contact frame, H has a simpler form. The bottom row contains zeros because there does not exist a contact coordinate velocity corresponding to a nonzero velocity in the z direction (a nonzero z velocity involves breaking contact or penetration). Although the upper 5×5 sub-matrix of H is in general dense, in a contact frame it can have several zeros if one of the surfaces is flat with straight equiparameter lines.

If there is near conforming contact between the surfaces, H becomes poorly conditioned. When there is conforming contact the upper 5×5 sub-matrix becomes singular. Consider the example of a ball in a socket. If the surfaces have almost but not quite the same shape, then very large parameter velocities are needed to account for small spatial velocities. When the surfaces conform exactly, the contact has a lower number of degrees of freedom and needs a different parameterization (see Anitescu et al. [1996] for an example of a cylinder on a flat surface going from conforming contact to point contact). In general, this is a difficult problem for all multi-body methods.

Although the derivation shown in Equations 9 through 14 is quick for demonstration, it lacks the convenience of the finer details necessary for an implementation. A detailed derivation more suitable for implementation and emphasizing ease of computation can be found in the appendix.

3.3 Constrained Dynamics

Let body *1* be free to move in contact with body *2* which is fixed. The Newton-Euler equations for a rigid body (see [Murray et al. 1994; Pai et al. 2000]) can be written using spatial vectors as

$$f_{ext} + f_c = M\dot{\phi} - [\phi]^T M\phi \tag{15}$$

where f_{ext} is the external force, f_c is the constraint force, ϕ is the spatial velocity of the body relative to the world (inertial) frame, M is the spatial inertia. All coordinates in this section are with respect to the body fixed frame, frame *1*. Note that $\dot{\phi}$ is the time derivative of ϕ in the body frame.

We can combine this equation with the contact kinematics equations,

$$\phi = H\dot{q}, \tag{16}$$

as a non-holonomic constraint to get a differential algebraic equation. The most common method for solving these types of equations is to differentiate the constraint until the system can be expressed as an ordinary differential equation. Differentiating the constraint once, in the body frame, we get

$$\dot{\phi} = \dot{H}\dot{q} + H\ddot{q}. \tag{17}$$

Combining Equation 15 with the constraint Equation 17 (instead of Equation 16) gives us an ordinary differential equation instead of a differential algebraic equation.

In the frictionless case, the constraint force, f_c, will be orthogonal to the degrees of freedom since it must not do any work on the system. This can be expressed as $H^T f_c = 0$. In the contact frame, f_c will be zero except for the translational z component as the first five rows of H are linearly independent and the sixth row is zero.

We will write these equations in the matrix form below where all quantities are expressed in frame *1* coordinates.

$$\begin{pmatrix} I & M & 0 \\ 0 & I & H \\ H^T & 0 & 0 \end{pmatrix} \begin{pmatrix} f_c \\ -\dot{\phi} \\ \ddot{q} \end{pmatrix} = \begin{pmatrix} b \\ -a \\ 0 \end{pmatrix} \tag{18}$$

$$b = -[\phi]^T M\phi - f_{ext}$$
$$a = \dot{H}\dot{q}$$

Performing block row elimination on the last row produces

$$\begin{pmatrix} I & M & 0 \\ 0 & I & H \\ 0 & 0 & H^T MH \end{pmatrix} \begin{pmatrix} f_c \\ -\dot{\phi} \\ \ddot{q} \end{pmatrix} = \begin{pmatrix} b \\ -a \\ r \end{pmatrix}, \tag{19}$$

$$r = H^T(f_{ext} + [\phi]^T M\phi - M\dot{H}\dot{q}).$$

The last row yields an ordinary differential equation in the contact coordinates which can be integrated directly; it is equivalent to the ODE

$$\ddot{q} = (H^T MH)^{-1} H^T(f_{ext} + [\phi]^T M\phi - M\dot{H}\dot{q}). \tag{20}$$

Note that although $H^T MH$ is an invertible 5×5 matrix we use a matrix factorization instead of explicit inversion.

The ODE in Equation 20 is not stiff and can be solved with explicit integration methods. Implicit integration methods are also possible, though they can be more expensive to compute and unduly stable.

Note that $H^T MH$ can become poorly conditioned in areas of high curvature, but more importantly, our method is sensitive to the surface parameterization. Undesirable accuracy to time-step trade offs result for low curvature surfaces with parameterizations that have high curvature equiparameter lines (for simple rolling or sliding motions the contact will need follow a high curvature path in parameter space). Section 3.6 discusses this problem in more detail.

To summarize, we choose our constraint as the description of how two surfaces move when they are in contact. Because we constrain the motion of our objects to remain in contact, a continuous contact motion results when we integrate while applying an external force. Evolving the system in a set of reduced coordinates avoids stabilization since the coordinates parameterize the constraint manifold exactly.

Since the constraint is bilateral we monitor the constraint force to check for when the surfaces are breaking contact. The constraint force f_c is computed easily during back-substitution of Equation 19. Separation occurs when the z component of ${}^{1c}f_c$ is negative.

Note that it is not difficult to extend this method to multi-body systems when the contacts exist either in a chain or tree structure. Both the composite rigid body method and articulated body method can be applied directly since the contacts are essentially generalized joints in a kinematic system [Pai et al. 2000].

3.4 Friction

Friction is a necessary component in any physically-based animation. This section presents a simple way of incorporating dry Coulomb friction into our algorithm.

With only a single point of contact between two bodies and isotropic friction, we can make the assumption that the friction force opposes the relative velocity. When the relative translational velocity is nonzero, the system experiences *dynamic friction* (discussed in Section 3.4.1). For pure rolling motions we allow the contact to evolve in a *no-slip friction* mode (discussed in Section 3.4.2). In no-slip mode

the system has fewer degrees of freedom as the contact will move at the same speed across both surfaces. In Section 3.4.3, we also discuss a less useful no-spin friction mode that we have not implemented.

3.4.1 *Dynamic Friction.* For translational friction due to sliding at a single point of contact, the tangential friction force is proportional to the normal constraint force and opposes the *slip velocity*. That is,

$$f_t = -\mu \hat{v} |f_n|, \quad v \neq 0, \tag{21}$$

where f_t is the tangent translational force due to friction, f_n is the normal force, and $\hat{v}$ is the normalized relative translational velocity. Summing tangent and normal components gives us the translational component of the constraint force.

This equation becomes very simple when written in coordinates of a contact frame since the constraint wrench in a contact frame is already decomposed into a component normal to the constraint manifold and a tangential component due to friction. The normal component is simply the z translational component (the sixth entry of ${}^{1c}f_c$). The remaining components are due to friction, hence we can write Equation 21 as

$${}^{1c}f_{c_4} = -\mu \ \hat{v}_x \ \ {}^{1c}f_{c_6}, \tag{22}$$

$${}^{1c}f_{c_5} = -\mu \ \hat{v}_y \ \ {}^{1c}f_{c_6}. \tag{23}$$

Since the constraint wrench in Equation 15 is in coordinates of frame *1*, we need to left multiply by the appropriate adjoint. Using the adjoint and Equations 22 and 23, the new constraint equation in matrix form becomes,

$$\begin{pmatrix} I_{5\times5} & 0 \end{pmatrix} \ {}^{1}_{1c}\mathsf{Ad}^T \ \ {}^{1}f_c = -\begin{pmatrix} 0_{5\times5} & \boldsymbol{\mu} \end{pmatrix} \ {}^{1}_{1c}\mathsf{Ad}^T \ \ {}^{1}f_c, \tag{24}$$

$$\text{where} \quad \boldsymbol{\mu} = \begin{pmatrix} 0 & 0 & 0 & (\mu \ \hat{v}_x) & (\mu \ \hat{v}_y) \end{pmatrix}^T. \tag{25}$$

This corresponds to the third block row of Equation 18. For clarity, the size of selected sub-matrices are shown with a trailing subscript. Note that the first three entries of $\boldsymbol{\mu}$ are zero because there is no torsional friction. The left hand side of Equation 24 extracts the tangential component of the constraint force, while the right hand side computes the tangential component based on the normal restoring force and friction (Equations 22 and 23). There are only five rows in the matrix because we do not place any restrictions on the normal force. The normal force is determined solely by the condition that the surfaces must not interpenetrate. Essentially, this equation directs the constraint wrench slightly off normal, such that it opposes the relative motion of the bodies. If the normal component needs to be larger to prevent the bodies from interpenetrating, then the tangential portion will also become larger due to the larger normal force.

Bringing everything to the left hand side in Equation 24 gives,

$$F \ \ {}^{1}f_c = 0, \tag{26}$$

$$\text{where} \ \ F = \begin{pmatrix} I_{5\times5} & \boldsymbol{\mu} \end{pmatrix} \ {}^{1}_{1c}\mathsf{Ad}^T. \tag{27}$$

The 5×6 matrix F replaces H^T in Equation 18. Repeating the block row simplification performed in Section 3.3 on the modified matrix results in the following

equation which we use instead of Equation 20.

$$FMH\ddot{q} = F(f_{ext} + [\phi]^T M\phi - M\dot{H}\dot{q}) \tag{28}$$

Since F is a function of ϕ, a time explicit integration method will use the relative spatial velocity from the previous time step in Equation 28. Note that high friction can cause the matrix FMH to become poorly conditioned.

3.4.2 *No-Slip Friction.* For pure rolling motions, the contact location moves across each object at the same speed. The dynamic friction method, described in the previous section, can often generate reasonable approximations of these motions. When the slip velocity is small, and provided the conditions exist for the system to exhibit pure rolling, the slip velocity tends to make small oscillations which effectively prevents noticeable slip between the objects. Although this may be adequate for some applications, it has the undesirable drawback that objects will slowly drift over one another rather than reaching a static rest configuration. This section presents a solution for generating pure rolling motions by requiring the slip velocity to be zero.

The no-slip friction condition imposes a non-holonomic unilateral constraint on motion; it only places a restriction on the contact coordinate velocities. The five-dimensional contact coordinates are still necessary to keep track of the configuration of the contacting objects. The contact coordinate velocities, however, are no longer independent, since they only have three degrees of freedom.

In this section, unless otherwise specified, we let H be in a contact frame (say ${}^{1c}H$, in the contact frame *1c*). Let $H_{i..j,k..l}$ be the sub-matrix consisting of rows i through j and columns k through l. For example, $H_{4..5,1..2}$ is the portion of H which relates the contribution of the contact coordinate velocities $(\dot{s}, \dot{t})$ to the tangential translational velocity of the contact, (v_x, v_y).

When the two objects in contact are moving without slipping, the relative translational velocity of the bodies in the contact frame will be zero. This defines a relationship between $(\dot{s}, \dot{t})$ and $(\dot{u}, \dot{v})$. Because $\dot{\psi}$ does not contribute to the slip velocity we can write this relationship as

$$H_{4..5,1..2} \begin{pmatrix} \dot{s} \\ \dot{t} \end{pmatrix} = -H_{4..5,3..4} \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix}. \tag{29}$$

$$\text{Or,} \quad \begin{pmatrix} \dot{s} \\ \dot{t} \end{pmatrix} = D \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix}, \tag{30}$$

$$\text{where} \quad D = -H^{-1}_{4..5,1..2} H_{4..5,3..4}. \tag{31}$$

This leads us to a smaller version of H, which relates the relative spatial velocity to a set of independent contact coordinate velocities. We can write this as

$$\phi = H_{1..6,1..2}\ D \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} + H_{1..6,3..4} \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} + H_{1..6,5}\ \dot{\psi}, \tag{32}$$

which has the matrix form

$$\phi = H' \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{\psi} \end{pmatrix}, \tag{33}$$

$$\text{where} \quad H' = \begin{pmatrix} (H_{1..6,1..2}\ D + H_{1..6,3..4}) & H_{1..6,5} \end{pmatrix}. \tag{34}$$

The no-slip equations of motion are now simply the equations in Section 3.3 with all occurrences of H and $\dot{q}$ replaced with this 6×3 matrix H' and the *reduced contact coordinate* velocities, $(\dot{u}, \dot{v}, \dot{\psi})^T$, from Equation 33. Solving this smaller system provides the reduced contact coordinate accelerations. The original $\dot{q}$ must still be integrated to get the new contact configuration q. This is done at each derivative computation step by computing $\dot{q}$ with Equation 30 using the reduced contact coordinate velocities that resulted from the integration of the reduced contact coordinate accelerations.

Simulations that include periods of both sliding and pure rolling are possible by switching, when appropriate, between the methods presented in this section and the previous section. To switch between models, we monitor the constraint force and slip velocity. The system remains in no-slip friction mode when the inequality

$$\frac{|f_t|}{|f_n|} \leq \mu_{\text{static}} \tag{35}$$

is satisfied. Recall that the tangential and normal components are easily extracted from the constraint force when it is written in the contact frame. When in sliding friction mode, if the magnitude of the slip velocity falls below the threshold then we switch back to the no-slip mode. We used thresholds ranging from 2.5 mm/s to 5 mm/s (approximately 2 to 4 percent of the largest dimension of the object) for the simulation described in Section 4.2.

3.4.3 *No-Slip and No-Spin Friction.* The no-slip method described above can easily be extended to prevent any spin between contacting objects. However, since the no-spin formulation does not solve any noticeable problems (such as the slow drift avoided with the use of no-slip friction), we have not implemented this method.

Additionally restricting the free object from spinning reduces the degrees of freedom to only two. The no-spin condition requires ω_z to be zero in the contact frame, which can be written as

$$H_{3,1..2} \begin{pmatrix} \dot{s} \\ \dot{t} \end{pmatrix} + H_{3,3..4} \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} + H_{3,5}\ \dot{\psi} = 0. \tag{36}$$

Since ${}^{1c}H_{3,5} = 1$ (see Section 3.1 or the appendix for more details), and using D from Equation 30 we can write

$$\dot{\psi} = -(H_{3,1..2}\ D + H_{3,3..4}) \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix}. \tag{37}$$

Equation 33 with the additional requirement of no spin becomes,

$$\phi = H'' \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix}, \tag{38}$$

$$\text{where} \quad H'' = H_{1..6,1..2}\ D + H_{1..6,3..4} + H_{1..6,5}(-(H_{3,1..2}\ D + H_{3,3..4}). \tag{39}$$

This 6×2 matrix along with $\dot{u}$ and $\dot{v}$ replace H and $\dot{q}$ in our dynamics equations of Section 3.3. After integrating the computed $\ddot{u}$ and $\ddot{v}$ to get $\dot{u}$ and $\dot{v}$, we use Equations 30 and 37 to compute $\dot{q}$.

The no-spin condition suggests that the contact area is large enough to transmit frictional torques. In this case we can compare the torque about the z axis with the magnitude of the normal force. That is, if the inequality

$$\frac{|{}^{1c}f_{c_3}|}{|f_n|} \leq \mu_{\text{spin static}} \tag{40}$$

is violated then we can switch to no-slip or sliding friction.

3.5 Traversing Patch Boundaries

Since object models are commonly a collection of surface patches, a robust method of evolving a contact across patch boundaries is needed. For example, the bowl and table in Figure 6 consist of 14 and 60 surface patches respectively. We use Stam [1998] for parametric evaluation of Loop subdivision surfaces, which recognizes that these surfaces are equivalent to piece-wise quartic box splines almost everywhere (see [Zorin and Schroder 2000] for background on subdivision surfaces). Normally, this involves subdividing the mesh once to ensure that there is at most one extraordinary vertex[3] per patch, but larger patches are preferred since they result in fewer patch transitions. Instead of subdividing, we evaluate a patch having more than one extraordinary vertex using consistent reparameterizations of its four subpatches.

If two contacting patches are both part of a larger group of regular patches on their respective surfaces, then we can treat the collection of patches as a single surface function over a larger domain (much like we do for patches with more than one extraordinary vertex). Although the resulting functions would have discontinuous 3^{rd} derivatives at the original patch boundaries, they would allow a contact to be evolved over larger distances without requiring a boundary traversal. We do not do this because the non regular regions at extraordinary vertices make it difficult. Their domains, along with the domains of the regular patches surrounding the non regular patches, can not be assembled into a larger domain without distorting the parameter space. Instead, we avoid the problem entirely by evolving the contact very close to the boundary and then computing an equivalent state that we can evolve into the next patch.

Note that our traversal technique requires tangent plane continuity between adjacent patches. Sharp edges occur when there are tangent plane discontinuities, bringing up the possibility of curve-surface and curve-curve contact, which require different contact coordinates and contact kinematics equations [Anitescu et al. 1996].

We find the state of the system at the transition by taking successively smaller steps, and backing up every time the state crosses the boundary. When the step size falls below a specified minimum, the current parameter velocity is used to project the contact location onto the exact boundary.

Note that it is important to approach the traversal point from inside the patch rather than interpolate the current state with the result of the step that evolved the contact outside of the current patch. Interpolation introduces large errors into the system because the surface functions are polynomial and tend to grow quickly outside of their domains and in directions different from the adjacent surface.

[3]For triangle meshes with Loop subdivision, extraordinary vertices have valence not equal to 6.

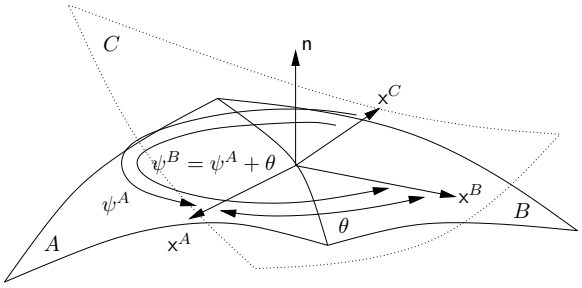

Fig. 4. To compute contact coordinates on patch B, the angle ψ has θ added to it. We add positive θ because $\mathsf{x}^A \times \mathsf{x}^B$ points in the same direction as the normal, n.

Assume that the domain of a patch is a polygon described by a counter clockwise list of points in the plane, $P_i,\ i = 1 \ldots n$. For every edge in the domain polygon we know both the adjacent patch and the corresponding domain edge. Given a point on the edge of a patch, we must be able to find the point on the corresponding domain edge of the adjacent patch mapping to the same location. In many cases, this relationship is quite simple.

In the case of Loop subdivision surfaces, the boundary curves (equivalent to quartic Bézier curves in most cases) of two adjacent patches are identical, and thus, a crossing at parameter α of one curve occurs at $1 - \alpha$ of the other, due to the counter clockwise order of the domain points.

Suppose that the contact moves from patch A to patch B on body 1, while the contact remains in patch C of body 2 (see Figure 4). When the state of the system is at a patch boundary we use a superscript to denote the patch whose coordinate system is being used. For example, q^A denotes the system configuration using patch A. Likewise, H^A denotes the contact kinematics equations using patch A.

Suppose the location of the contact is at a point α along edge i of patch A's domain polygon, that is, at $P^A_{i+1}\alpha + P^A_i(1-\alpha)$ in the domain of patch A. Assume edge j of patch B corresponds with edge i of patch A. We can write the state of the system on the boundary of patch B as

$$q^B = \begin{pmatrix} P^B_{j+1}(1-\alpha) + P^B_j\alpha \\ q^A_{3..4} \\ \psi^B \end{pmatrix} \tag{41}$$

where the new rotation ψ^B aligning the contact frames is yet to be determined. Note that the third and fourth components of the contact coordinate remain unchanged, since they only involve patch C.

As shown in Figure 4, we compute ψ^B by adding a correction to ψ^A to reflect the angle between the x axes of the contact frames at the crossing position of the adjacent patches. Since $\cos^{-1}(\mathsf{x}^A \cdot \mathsf{x}^B)$ is normally computed as an angle in the range $[0, \pi]$, we check to see if $\mathsf{x}^A \times \mathsf{x}^B$ points in the opposite direction of the surface normal, in which case we negate the correction angle. That is,

$$\psi^B = \psi^A + \mathrm{sgn}((\mathsf{x}^A \times \mathsf{x}^B) \cdot \mathsf{z})\ \cos^{-1}(\mathsf{x}^A \cdot \mathsf{x}^B). \tag{42}$$

With the complete contact coordinates known on both sides of the boundary, the

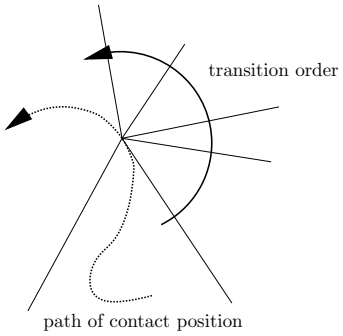

Fig. 5. Contact evolution through a corner and the resulting patch transitions.

destination patch contact coordinate velocity can be computed. Since the contact frame of body 2 is fixed, we equate the spatial velocities in coordinates of frame $2c$ as given by the contact kinematics equations. Since the spatial velocity is computed in a contact frame, we drop the z component (which is zero) yielding a system that is not over-constrained. Hence, we solve for $\dot{q}^B$ in

$$ {}^{2c}\hat{H}^A \; \dot{q}^A = \; {}^{2c}\hat{H}^B \; \dot{q}^B \tag{43}$$

where ${}^{2c}\hat{H}$ is a 5×5 matrix equivalent to the matrix ${}^{2c}H$ with the bottom (zero) row removed.

This completes the calculation of the system state in terms of patches B and C instead of A and C. Note that a transition on the other object, say from patch C into an adjacent patch D, is completely independent of our transition from A to B. The other transition can occur either before or after without changing the end result.

Consider the infrequent special case of a contact that evolves directly through the corner of a patch, as shown in Figure 5. Although the contact velocity may not point toward one of the adjacent patches (one sharing a boundary rather than just a vertex with our current patch), we still transfer the contact to an adjacent patch. Once in the adjacent patch we try again to advance the system to time $t + \Delta t$; however, the contact may evolve through the corner again (at time t). We let the process repeat until the contact evolves into the interior of the domain. If we have n patches meeting at a point, we may need to perform $n - 1$ transitions before finding the final destination patch (or detecting insufficient accuracy of the minimum step size), because we consistently choose counterclockwise transitions at the traversal point to prevent oscillating transitions between adjacent patches.

3.6 Surface Representation Problems

Subdivision surfaces are very convenient for describing object geometry. Their popularity in recent years is attributable to the relative ease with which they can be edited and implemented. As mentioned in Section 3.3, however, our method is sensitive to the parameterization of the surface. Problems can arise for parameterizations which have high curvature equiparameter lines even when the surface does not. Unfortunately, the natural parameterization given by Stam [1998] for Loop subdivision surfaces has this problem near extraordinary vertices, along with the

additional problem of non-regularities at the extraordinary vertices. Contact coordinate velocities are an essential part of the system state–the velocity is directly related to surface derivatives–so our contact evolution technique effectively breaks down in areas where the magnitude of the surface derivatives go to zero or infinity, such as when the contact point approaches an extraordinary vertex. Note that the non-regularity problem is not confined to subdivision surfaces. For example, the NURBS definition of a sphere has two non-regular points at its poles (though this too is avoidable, for example, with rational quartic Bézier triangles [Farin et al. 1988]).

In an attempt to circumvent this problem, Kry [2000] considers replacing the surface near extraordinary vertices with S-patches [Loop and DeRose 1989]. This is quite inefficient due to the expense involved in evaluating S-patches (see Table I). More recently, however, Peters [2001] shows that Loop surfaces can be approximated using only quartic Bézier patches. While ideal for our algorithm, we have not implemented this patching method, and instead we avoid the problem of irregularity by forcing the surfaces to separate when the contact evolves too close to extraordinary vertices. The objects then evolve by making small bounces, similar to a method described by Mirtich and Canny [1995], until the continuous contact can be restored on the other side of the problem area. The magnitude of the impulses used to resolve collisions during this period may need to be increased to prevent the objects from intersecting after advancing the system by the minimum step. Note that this change in simulation philosophy could make the incorporation of contact evolution into a multi-body setting tricky, since impulses resolving one contact may violate interpenetration constraints of others. For this reason, it would be desirable to just avoid the problem entirely with good surface parameterizations, such as those built using the surface repatching method described by Peters [2001].

3.7 Algorithm Summary

At this point it may be useful to review the entire algorithm. Everything is centered around the computation of $\ddot{q}$. For a single contact, the amount of work needed to do this computation does not vary.

While the simulation has not achieved the target time, we try to step the integrator by the current step size Δt, which may result in the contact crossing a patch boundary or breaking contact. To compute $(q, \dot{q})$ at $t + \Delta t$, the integrator requires us to compute the state derivative at a given state and time (this computation occurs multiple times for higher order methods). That is, for example, we need to produce $(\dot{q}, \ddot{q})$ at time t given $(q, \dot{q})$ at time t. This only involves computing $\ddot{q}$ as we can use the $\dot{q}$ provided in the current state. The whole process can be summarized briefly by the steps that follow.

—First we evaluate the surfaces at the current contact coordinates (q at time t). That is, we must compute the position of the contact point on each surface and all the partial derivatives of order up to three. As the basis functions of parametric surfaces are smooth, and due to the symmetry of mixed partials ($\mathsf{c}_{,uv} = \mathsf{c}_{,vu}$), only ten sets of basis functions are needed to do this evaluation[4].

[4]Note that Equation 20 uses $\dot{H}$, which contains third order partial derivatives.

—Next we compute the position and orientation of the bodies in world coordinates. This is necessary for computing any external forces, such as gravity. We do this by computing ${}^{2}_{1}\mathsf{E}$ from Equation 9 using the surface evaluations computed above.

—We then compute H with the method described in the appendix. With H in the free body reference frame, we calculate ϕ using $\dot{q}$ from the current state.

—Having everything required to compute $\ddot{q}$, we proceed by solving Equation 20 for the frictionless case, or by solving Equation 28 for the friction case. This is accomplished with an LU decomposition of $H^T M H$ in the frictionless case, or FMH in the friction case.

—Back substituting $\ddot{q}$ into the second row of Equation 19 gives the contact wrench f_c which we use to detect breaking contact (the constraint in this formulation is bilateral). When the z component of ${}^{1c}f_c$ becomes negative we allow the objects to separate.

—If the integrator computes a new contact coordinate outside the current domain boundary, then we revert the system to time t and reintegrate with a smaller time step, given the last step size was above the minimum step size. Otherwise, the state is projected onto the domain boundary using the current velocity and the new state on the other side of the boundary is computed as described in Section 3.5.

—Provided no boundary traversal or separation occurs, the time advances to $t+\Delta t$ and the computed values of q and $\dot{q}$ at $t+\Delta t$ become the current state. If the no-slip friction model is active during this last step, then the computed $\dot{q}$ only contains the reduced contact coordinate velocities. In this case the complete set of contact coordinate velocities is computed using Equation 30.

—Last, given a successful step, we also increase the current step size, though not beyond a previously specified maximum step size.

When the bodies are not in contact, such as after a separation, we evolve the bodies without constraints. If we observe transient collisions occurring in close succession and proximity, then we switch back to reduced coordinates. For collision detection, our implementation uses sphere trees [Quinlan 1994] built from polyhedral approximations of our models. In establishing a continuous contact, we first compute approximate contact coordinates from the minimum distance reported by the sphere tree. Before using these coordinates we apply a few Newton iterations, as described in [Nelson et al. 1999], to find a better approximation of the contact point on the smooth surfaces.

4. RESULTS AND DISCUSSION

We have implemented a rigid body simulator in Java that uses our contact evolution technique. When objects are not in contact we use a sphere tree [Quinlan 1994] for collision detection and an algebraic collision law due to Chatterjee and Ruina [1998] for collision response. Our simulations with Loop subdivision surfaces, such as those in Figures 1 and 6, run in real time on current hardware and in near real time on our reference machine used for timing specific computations in the following section.

Table I. Timings for calculating the ten partial derivatives necessary for the algorithm. Timings are with HotSpot optimizations, run on a 350 MHz Pentium II. Times are an average of ten thousand successive computations and hence may appear smaller than they should due to caching effects.

surface type	evaluation time
Loop regular	0.2413 ms
Loop extraordinary (degree 3)	0.1683 ms
Loop extraordinary (degree 5)	0.1933 ms
Loop extraordinary (degree 7)	0.2223 ms
Loop extraordinary (degree 10)	0.2654 ms
S-patch (3 sides)	0.1632 ms
S-patch (4 sides)	0.4466 ms
S-patch (5 sides)	1.1006 ms
S-patch (10 sides)	20.0248 ms
Bi-cubic Bézier	0.0801 ms

4.1 Timings

If running the HotSpot Java virtual machine on a 350 MHz Pentium II machine, the entire computation of $\ddot{q}$ takes about 1.2 ms. With this computation time the simulation can run at 15 frames per second without using all available CPU cycles. The bulk of the time is spent in Java3d code to draw the system.

Table I shows the time necessary to evaluate various types of surface functions along with all the necessary partial derivatives for the algorithm. We performed all our tests on a 350 MHz Pentium II running Java 1.3. All times reported in this section are measured as an average of ten thousand computations to give the HotSpot virtual machine sufficient opportunity to optimize our code. Although timings in the table may be smaller than in practice due to caching effects, all other times reported in this section do not have this bias as they were measured during an actual simulation.

We found that evaluating the surface functions takes up a substantial portion of the simulator's time. In our implementation, these functions have quite a bit of room for optimization. Surface representations which can be evaluate quickly are thus preferred for our algorithm. For two contacting cubic Bézier surfaces, the program computes H in 0.3485 ms on our test machine. Note that Table I reveals that half of this time is spent on surface evaluations of the two patches. The routine which computes $\ddot{q}$ takes about 0.235 ms (this includes the LU factorization and back substitution). The total computation time for the derivatives comes to about 0.9 ms in the Bézier on Bézier case. The time which is unaccounted for comes from checking boundaries, computing object positions, velocities, and external forces.

4.2 Example Simulation and Results

To evaluate the benefits of our contact evolution algorithm, we look at the simulations which result when we change the upper limit on the step size.

Figure 6 shows the simulation of a bowl thrown on a table, repeated three times. Loop subdivision surfaces define the boundaries of both objects. The table surface is approximately 40 cm square while the bowl is approximately 13 cm across, 11 cm high, and weighs a little less than 1 kg. For all three simulations the minimum step size is fixed at 1 ms, while the maximum step size is set to 1 ms, 16 ms, and 64 ms seconds respectively. We integrate using an explicit fourth order Runge-Kutta

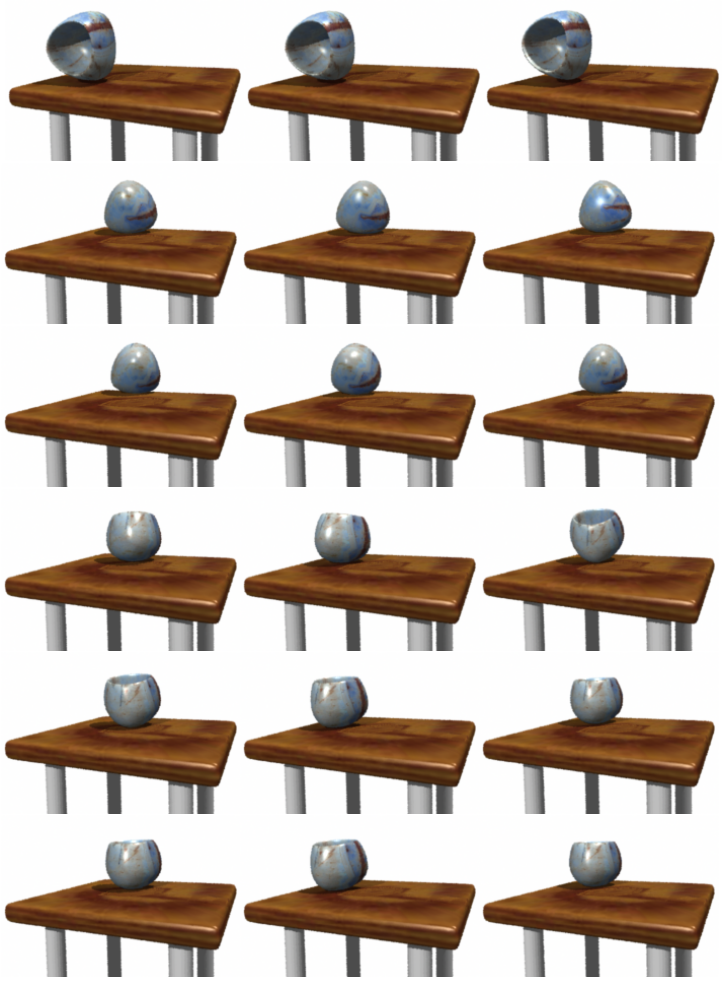

Fig. 6. The three columns show sequences of the bowl thrown on the table. Note that the table edges are rounded. The six frames in each sequence are in half second intervals. The maximum step size, from left to right, is 1 ms, 16 ms and 64 ms respectively. All three settings yield plausible motion for the bowl on the table.

method.

Figure 7 shows the trajectories of the bowl's center of mass for these three cases. The down pointing triangles mark where the bowl establishes continuous contact with the table. With larger maximum step sizes the integration error increases dramatically (as expected for any method due to larger truncation errors). What we notice here and in Figure 6 is that the bowl behaves in a relatively similar manner for all three cases.

In each simulation, the bowl collides with the table several times before establishing a continuous contact. All three simulations show similar periods of both pure rolling and sliding as the bowl gradually comes to rest on the table. The slip velocity threshold for entering no-slip friction is 5 mm/s for all but the 1 ms max-

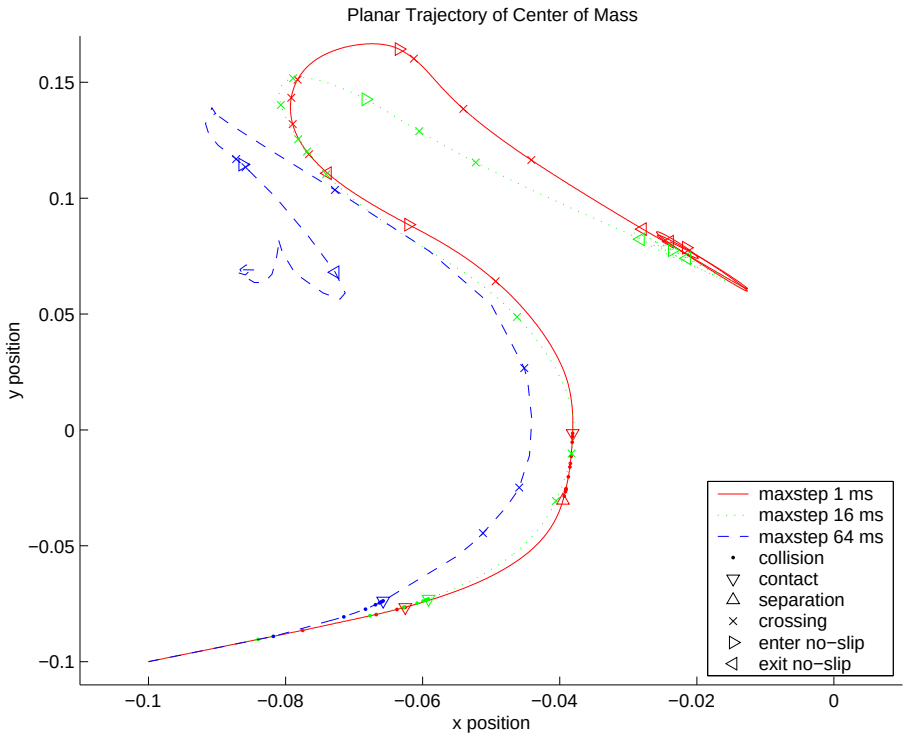

Fig. 7. Planar position of the center of mass of a bowl thrown on a table. Trajectories for the step sizes limited to 1 ms, 16 ms, and 64 ms are shown in solid red, dotted green and dashed blue respectively. Each trajectory has additional markings to show collisions, transitions between patches, transitions between dynamic and no-slip friction and the transitions to and from continuous contact.

imum step size simulation where a threshold of 2.5 mm/s was used giving better accuracy.

Although the three trajectories follow different paths, they are all plausible motions for the bowl thrown onto the table. Thus, one can sacrifice accuracy for speed, without sacrificing visual plausibility. This is a major advantage of the method, however, there does comes a point when the error is too large. In this case, with a maximum step size of 128 ms, the slip velocity of the bowl on the table in the resulting simulation does not fall below the 5 mm/s threshold necessary to enter no-slip friction, which results in the bowl wobbling and sliding until it falls off the table.

The CPU times for the 3 second simulations in Figure 7 are shown in Table II along with the times for other settings of the maximum step size. This includes the collision detection computations during the brief period that the bowl is not in contact, as well as the contact evolution computations for the periods of dynamic friction and no-slip friction. Because the large smooth areas of the bowl and the table are described with a small number of patches, there exist long periods where larger steps can be taken without crossing a patch boundary. This is seen in the

Table II. Timings for a 3 second simulation for the bowl on the table with different maximum step sizes. These timings were taken on a machine with a 1.8 GHz Xeon processor. Note that this particular simulation with our unoptimized implementation requires steps larger than 2 ms for real time computation.

maximum step size	simulation time
1 ms	6.542 s
2 ms	3.328 s
4 ms	1.766 s
8 ms	0.974 s
16 ms	0.521 s
32 ms	0.370 s
64 ms	0.286 s

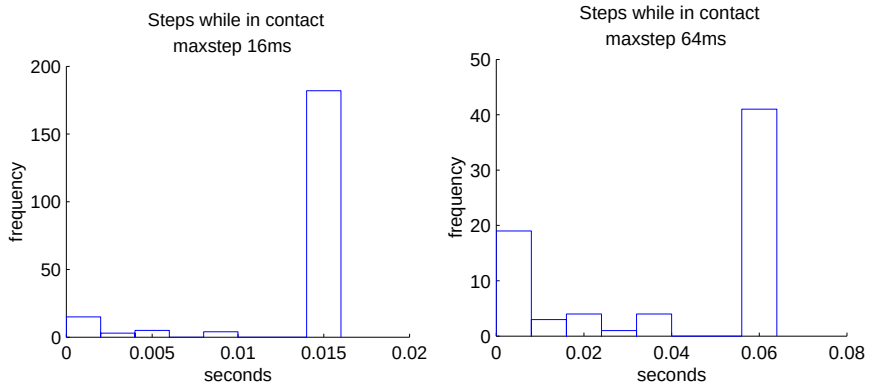

Fig. 8. Histograms of the successful step sizes used while in contact for the 16 ms and 64 ms maximum step size simulations in Figures 6 and 7. Although dependent on the given simulation, these histograms show the effect that boundary crossing events have on the variation of step sizes. Note that boundary crossings occur only when a minimum step (in this case, 1 ms) evolves the contact out of one of the currently contacting patches.

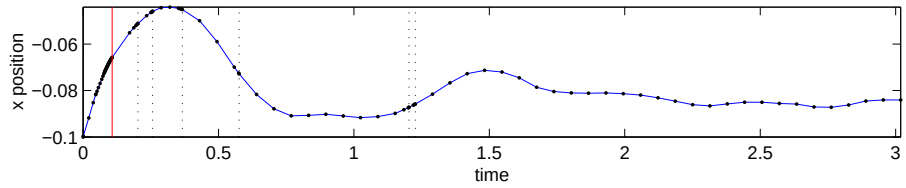

Fig. 9. Center of mass x position during the first second of the bowl on table simulation. Patch transitions are denoted with a vertical dotted line while conversions to and from continuous contact are shown with a solid red vertical line. The data points of the computed trajectory are shown with dots. Between data points, either linear or Hermite interpolation is used to compute a system state for visualization.

timings as the halving of computation times with the doubling of the maximum step size, since the simulations are not dominated by the patch boundary crossing events. Figure 8 also shows this with histograms of successful step sizes during the contact evolution portion of two simulations.

Last, Figure 9 shows the x position of the bowl's center of mass over time for

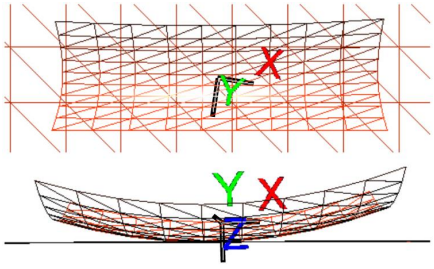

Fig. 10. Top and front views of our rattleback model.

the maximum step size case of 64 ms. The dashed vertical lines denote boundary crossings, while the distances between dots correspond to step sizes. Because the rendered simulation has a constant frame rate, not synchronized with the adaptive time steps, we compute the system state at each frame with either linear or Hermite interpolation. During periods of continuous contact this consists of interpolating the reduced coordinates.

4.2.1 *Rattleback.* Rattleback tops, also known as celts or wobblestones, are interesting because they can reverse their spin. Some rattlebacks reverse their spin in both directions while others have a spin bias and only reverse their direction if spun in the direction opposite to their preferred direction. The nonintuitive behaviour of rattleback tops makes them interesting test objects for contact simulations (see, for example, [Mirtich 1996]).

We tested our simulation with the very simple rattleback model shown in Figure 10 consisting of a single bi-cubic Bézier patch. It is longest in a direction 10 degrees off of the x axis and its width is one third of its length. Its curved shape comes from elevating the outer control points. We spin the model around its z axis.

Although the top is symmetric, we set the inertia tensor such that it is not in alignment with the planes of symmetry. A nonuniform mass density can cause this in a physical model. The reversal effect is due to the misalignment of the principal axes of inertia with the principal axes of curvature at the point of contact [Garcia and Hubbard 1988]. With this simple model we can simulate single or multiple spin reversals, depending on viscous damping and the coefficient of friction.

Figure 11 shows a sequence of frames where the top reverses its spin after about eight seconds. Several previous positions of the top are drawn in grey to indicate the direction of motion. We start the spin slightly off the center of the top because a perfect spin around the z axis results in a stable rotation. Just before the rattleback model reverses its spin, it rocks with a period of about 0.5 seconds. We find that this simulation gives best results for step sizes smaller than 50 ms. Simulations with steps of about 50 ms result in the top losing most of its energy during the reversal; the top does not end up spinning convincingly in the opposite direction. Although the top spins stably with larger step sizes such as 100 ms, it does not reverse. Other motions, such as tapping the rattleback at one end, are obviously not stable for these exaggerated step sizes.

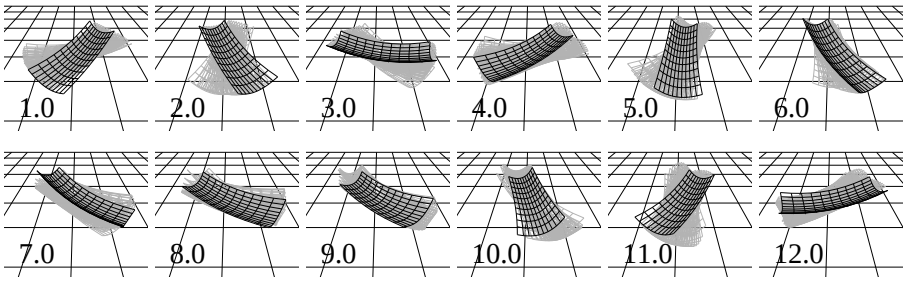

Fig. 11. Spin start of a rattleback. The lower left corner of each image shows the time in seconds and the previous positions of the rattleback are shown in grey. The direction of the spin is initially counter clockwise and quickly changes to clockwise. The maximum step size for this simulation was 10 ms.

5. CONCLUSIONS

We have described a fast contact evolution algorithm for single contact between piece-wise parametric surfaces. We first derived contact kinematics equations for arbitrary regular parametric surfaces. Next, we showed that using these equations as acceleration constraints we can formulate the contact dynamics as an ODE in the contact coordinates. The resulting ODE can be easily integrated using explicit integrators, without the need for constraint stabilization. We easily incorporate a Coulomb friction model into our formulation. A method of evolving contacts across patch boundaries is presented.

Our contact evolution technique, although useful for single contact, is not suitable for all configurations of single contact due to its sensitivity to the parameterization of the contacting surfaces. Although many parametric surfaces can have non-regular points, such as Loop subdivision surfaces with their natural parameterization, these points are limited in number and can be avoided entirely by repatching [Peters 2001], or by evolving the system with an alternate method when the system state approaches these configurations.

As in all simulation algorithms, the truncation error can be quite large if we take large step sizes while in continuous contact. These errors, however, can be quite acceptable for visual realism as they do not violate the noninterpenetration constraint.

Having measured the time used to perform various calculations for our simulation, we notice that a large proportion of the time can be spent computing partial derivatives of the parametric surfaces, which suggests that surfaces that are less expensive to evaluate are more desirable with our technique (for example, Catmull-Clark subdivision surfaces, which are bi-cubic almost everywhere).

Our method extends easily to multi-body systems with chain or tree structures. The benefits of our method decrease, however, for systems with kinematic loops, such as a pair of objects with multiple contacts, since these systems do not, in general, reduce to an ODE and require stabilization of the extra constraints. Nevertheless, we feel that the contact evolution method we have presented here is both interesting and useful for rigid body simulation.

APPENDIX

This derivation of the contact kinematics equations uses the same setup as described by Equations 5 through 8 in Section 3.1.

As the contact point moves, so do the contact frames. The spatial velocity of the contact frame *1c* relative to the body frame *1* in terms of contact parameter velocities, has a particularly simple form in the contact frame. If ${}^{1c}\phi(1,1c) = (\omega^T, v^T)^T$, then the spatial velocity of the contact frame *1c* relative to the body frame *1* in coordinates of *1c* is

$$\begin{pmatrix} [\omega] & v \\ 0 & 0 \end{pmatrix} = {}^{1c}_{1}\mathsf{E}\ {}^{1}_{1c}\mathsf{E}_{,s}\ \dot{s} + {}^{1c}_{1}\mathsf{E}\ {}^{1}_{1c}\mathsf{E}_{,t}\ \dot{t}. \tag{44}$$

Let Θ be the leading 3×3 sub-matrix of ${}^{1}_{1c}\mathsf{E}$ (see Equation 6) and recall that the rightmost column of ${}^{1}_{1c}\mathsf{E}$ is the location. Examining the matrix multiplications in Equation 44 for a closer look at the contributions of $\dot{s}$ and $\dot{t}$ to ω and v yields

$$[\omega] = \Theta^T\Theta_{,s}\ \dot{s} + \Theta^T\Theta_{,t}\ \dot{t}, \quad v = \Theta^T\mathsf{c}_{,s}\ \dot{s} + \Theta^T\mathsf{c}_{,t}\ \dot{t}. \tag{45}$$

First, consider ω by examining the skew symmetric matrix $\Theta^T\Theta_{,s}$ [5] ($\Theta^T\Theta_{,t}$ will be very similar).

$$\Theta^T\Theta_{,s} = \begin{pmatrix} \mathsf{x}\cdot\mathsf{x}_{,s} & \mathsf{x}\cdot\mathsf{y}_{,s} & \mathsf{x}\cdot\mathsf{z}_{,s} \\ \boxed{\mathsf{y}\cdot\mathsf{x}_{,s}} & \mathsf{y}\cdot\mathsf{y}_{,s} & \boxed{\mathsf{y}\cdot\mathsf{z}_{,s}} \\ \boxed{\mathsf{z}\cdot\mathsf{x}_{,s}} & \mathsf{z}\cdot\mathsf{y}_{,s} & \mathsf{z}\cdot\mathsf{z}_{,s} \end{pmatrix} \tag{46}$$

The least complicated expressions relating ω to $\dot{s}$ are those which are boxed in Equation 46 ($\mathsf{x}_{,s}$ is less complicated than $\mathsf{z}_{,s}$ which is less complicated than $\mathsf{y}_{,s}$). Thus we can write ω as,

$$\omega = \begin{pmatrix} -\mathsf{y}\cdot\mathsf{z}_{,s} \\ -\mathsf{z}\cdot\mathsf{x}_{,s} \\ \mathsf{y}\cdot\mathsf{x}_{,s} \end{pmatrix} \dot{s} + \begin{pmatrix} -\mathsf{y}\cdot\mathsf{z}_{,t} \\ -\mathsf{z}\cdot\mathsf{x}_{,t} \\ \mathsf{y}\cdot\mathsf{x}_{,t} \end{pmatrix} \dot{t}. \tag{47}$$

Now looking at the $\Theta^T c_{,s}$ and $\Theta^T c_{,t}$ components we can write,

$$\Theta^T\mathsf{c}_{,s} = \begin{pmatrix} \mathsf{x}\cdot\mathsf{c}_{,s} \\ \mathsf{y}\cdot\mathsf{c}_{,s} \\ \mathsf{z}\cdot\mathsf{c}_{,s} \end{pmatrix} = \begin{pmatrix} \mathsf{x}\cdot\mathsf{c}_{,s} \\ 0 \\ 0 \end{pmatrix}, \tag{48}$$

$$\Theta^T\mathsf{c}_{,t} = \begin{pmatrix} \mathsf{x}\cdot\mathsf{c}_{,t} \\ \mathsf{y}\cdot\mathsf{c}_{,t} \\ \mathsf{z}\cdot\mathsf{c}_{,t} \end{pmatrix} = \begin{pmatrix} \mathsf{x}\cdot\mathsf{c}_{,t} \\ \mathsf{y}\cdot\mathsf{c}_{,t} \\ 0 \end{pmatrix}. \tag{49}$$

Equations 47, 48 and 49 combine to give ${}^{1c}H_1$.

$$ {}^{1c}\phi(1,1c) = {}^{1c}H_1 \begin{pmatrix} \dot{s} \\ \dot{t} \end{pmatrix} \tag{50}$$

[5] The skew symmetric property of this matrix is easily seen by differentiation of the identity $\Theta^T\Theta = I$.

$$\text{where } {}^{1c}H_1 = \begin{pmatrix} -\mathsf{y}\cdot\mathsf{z}_{,s} & -\mathsf{y}\cdot\mathsf{z}_{,t} \\ -\mathsf{z}\cdot\mathsf{x}_{,s} & -\mathsf{z}\cdot\mathsf{x}_{,t} \\ \mathsf{y}\cdot\mathsf{x}_{,s} & \mathsf{y}\cdot\mathsf{x}_{,t} \\ \mathsf{x}\cdot\mathsf{c}_{,s} & \mathsf{x}\cdot\mathsf{c}_{,t} \\ 0 & \mathsf{y}\cdot\mathsf{c}_{,t} \\ 0 & 0 \end{pmatrix} \tag{51}$$

Note that here the subscript on H denotes body 1 as opposed Equation 13 where we denote a relationship to an individual component of the contact coordinates. We can analogously define ${}^{2c}H_2$. This transforms to frame *1c* as

$${}^{1c}H_2 \;=\; {}^{1c}_{2c}\mathsf{Ad}\;\; {}^{2c}H_2 \;=\; \begin{pmatrix} R_\psi & 0 \\ 0 & R_\psi \end{pmatrix}\; {}^{2c}H_2. \tag{52}$$

Finally, relative spatial velocity of the two contact frames is a pure rotation about the surface normal, that is,

$${}^{1c}\phi(1c, 2c) = {}^{1c}H_\psi \dot{\psi}, \tag{53}$$

$$\text{where, } {}^{1c}H_\psi = \begin{pmatrix} 0 & 0 & -1 & 0 & 0 & 0 \end{pmatrix}^T. \tag{54}$$

We can now compute the contact matrices H_k as follows. The relative spatial velocity of the two bodies is given by

$$\begin{aligned} \phi(1, 2) \;&=\; \phi(1, 1c) + \phi(1c, 2c) + \phi(2c, 2), \\ &=\; \phi(1, 1c) + \phi(1c, 2c) - \phi(2, 2c). \end{aligned}$$

Substituting Equations 50, 52 and 54 yields

$${}^{1c}\phi(1, 2) = {}^{1c}H\dot{q}, \tag{55}$$

$$\text{where } {}^{1c}H = \begin{pmatrix} {}^{1c}H_1 & -{}^{1c}H_2 & {}^{1c}H_\psi \end{pmatrix}. \tag{56}$$

The contact matrix H can now be transformed to any convenient frame, for instance, frame *1*: ${}^{1}H = {}^{1}_{1c}\mathsf{Ad}\;\; {}^{1c}H$.

In Section 3.3 we need ${}^{1c}\dot{H}$ as we will take the time derivative of Equation 55. Once ${}^{1c}\dot{H}_1 = {}^{1c}H_{1,s}\dot{s} + {}^{1c}H_{1,t}\dot{t}$ and ${}^{2c}\dot{H}_2 = {}^{2c}H_{2,u}\dot{u} + {}^{2c}H_{2,v}\dot{v}$ are computed with the current state (q and $\dot{q}$) using the chain and product rules, we can write

$${}^{1c}\dot{H} = \begin{pmatrix} {}^{1c}\dot{H}_1 & -({}^{1c}_{2c}\dot{\mathsf{Ad}}\;\; {}^{2c}H_2 + {}^{1}_{1c}\mathsf{Ad}\;\; {}^{2c}\dot{H}_2) & 0 \end{pmatrix}. \tag{57}$$

REFERENCES

Anitescu, M., Cremer, J., and Potra, F. 1996. Formulating 3D contact dynamics problems. *Mechanics of Structures and Machines 24,* 4, 405–437.

Anitescu, M. and Potra, F. 1997. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics 14*, 231–247.

Ascher, U., Pai, D. K., and Cloutier, B. 1997. Forward dynamics, elimination methods, and formulation stiffness in robot simulation. *International Journal of Robotics Research 16,* 6, 749–758.

Baraff, D. 1990. Curved surfaces and coherence for non-penetrating rigid body simulation. *Computer Graphics (SIGGRAPH 90 Conference Proceedings) 24,* 4, 19–28.

Baraff, D. 1996. Linear-time dynamics using Lagrange multipliers. In *SIGGRAPH 96 Conference Proceedings.* Annual Conference Series. ACM SIGGRAPH, Addison Wesley, 137–146.

Chatterjee, A. and Ruina, A. 1998. A new algebraic rigid body collision law based on impulse space considerations. *Journal of Applied Mechanics 65,* 4, 939–951.

Farin, G., Piper, B., and Worsey, A. 1988. The octant of a sphere as a non-degenerate triangular Bézier patch. *Computer Aided Geometric Design 4,* 4, 329–332.

Featherstone, R. 1987. *Robot dynamics algorithms*. Kluwer.

Garcia, A. and Hubbard, M. 1988. Spin reversal of the rattleback: theory and experiment. *Proceedings of the Royal Society. London. Series A. Mathematical, Physical and Engineering Sciences 418,* 1854, 165–197.

Gilbert, E. G., Johnson, D. W., and Keerthi, S. S. 1988. A fast procedure for computing the distance between complex objects in three dimensional space. *IEEE Journal of Robotics and Automation 4,* 2, 193–203.

Goyal, S. 1989. Second order kinematic constraint between two bodies rolling, twisting and slipping against each other while maintaining point contact. Technical Report TR89-1043, Cornell University, Computer Science Department. October.

Jain, A. 1991. Unified formulation of dynamics for serial rigid multibody systems. *Journal of Guidance, Control, and Dynamics 14,* 3, 531–542.

Kry, P. G. 2000. Fast contact evolution for piecewise smooth surfaces. M.S. thesis, University of British Columbia.

Lin, M. and Canny, J. F. 1991. Efficient algorithms for incremental distance computation. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Loop, C. T. and DeRose, T. D. 1989. A multisided generalization of Bézier surfaces. *ACM Transactions on Graphics 8,* 3, 204–234.

Mirtich, B. V. 1996. Impulse-based dynamic simulation of rigid body systems. Ph.D. thesis, University of California at Berkeley.

Mirtich, B. V. and Canny, J. F. 1995. Impulse-based dynamic simulation of rigid bodies. In *Symposium on Interactive 3D Graphics*.

Montana, D. J. 1988. The kinematics of contact and grasp. *International Journal of Robotics Research 7,* 3, 17–32.

Murray, R., Li, Z., and Sastry, S. S. 1994. *A mathematical introduction to robotic manipulation*. CRC Press.

Nelson, D. D. and Cohen, E. 1998. User interaction with CAD models with nonholonomic parametric surface constraints. International Mechanical Engineering Congress and Exposition. Haptic Symposium.

Nelson, D. D., Johnson, D. E., and Cohen, E. 1999. Haptic rendering of surface-to-surface sculpted model interaction. In *Proceedings of the ASME Dynamic Systems and Control Division*. Vol. DSC 67. 101–108.

Pai, D. K., Ascher, U. M., and Kry, P. G. 2000. Forward dynamics algorithms for multibody chains and contact. In *IEEE International Conference on Robotics and Automation*. 857–863.

Peters, J. 2001. Smooth patching of refined triangulations. *ACM Transactions on Graphics 20,* 1, 1–9.

Quinlan, S. 1994. Efficient distance computation between non-convex objects. In *IEEE International Conference on Robotics and Automation*. 3324–3330.

Rabier, P. J. and Rheinboldt, W. C. 1993. On the numerical solution of the Euler-Lagrange equations. Technical Report ICMA-93-177, University of Pittsburgh. February.

Stam, J. 1998. Evaluation of loop subdivision surfaces. In *SIGGRAPH 98 Conference Proceedings*. Supplemental paper provided on conference proceedings CD-ROM.

van den Doel, K., Kry, P. G., and Pai, D. K. 2001. FoleyAutomatic: Physically-based sound effects for interactive simulation and animation. In *SIGGRAPH 2001 Conference Proceedings*. Annual Conference Series. ACM SIGGRAPH, Addison Wesley, 537–544.

Zorin, D. and Schroder, P. 2000. Subdivision for modeling and animation. SIGGRAPH 2000 Course Notes. Course number 23.

...