A Fast Impulsive Contact Suite for Rigid Body Simulation

Harald Schmidl and Victor J. Milenkovic

Abstract—A suite of algorithms is presented for contact resolution in rigid body simulation under the Coulomb friction model: Given a set of rigid bodies with many contacts among them, resolve dynamic contacts (collisions) and static (persistent) contacts. The suite consists of four algorithms: 1) partial sequential collision resolution, 2) final resolution of collisions through the solution of a single convex QP (positive semidefinite quadratic program), 3) resolution of static contacts through the solution of a single convex QP, 4) freezing of "stationary" bodies. This suite can generate realistic-looking results for simple examples yet, for the first time, can also tractably resolve contacts for a simulation as large as 1,000 cubes in an "hourglass." Freezing speeds up this simulation by more than 25 times. Thanks to excellent commercial QP technology, the contact resolution suite is simple to implement and can be "plugged into" any simulation algorithm to provide fast and realistic-looking animations of rigid bodies.

Index Terms—Quadratic programming, computer graphics, physically-based modeling, simulation, animation.

1 Introduction

Everybody is familiar with computer generated images and movies. Although most people are not aware of the underlying theories and algorithms, computer graphics (CG) techniques have found their way into living rooms through animation and special effects for features and advertisements in video and print. Our work focuses on techniques for computer animation. Sometimes it is not of the utmost importance that the final simulation be 100 percent physically accurate. It is the main concern of CG to generate images and simulations that "look right." Therefore, CG allows liberty regarding implementation. It is permissible to trade off realism for performance as long as physical plausibility is not violated. In some instances, the laws of physics are tweaked deliberately to generate more dramatic results, as with the work which has been done in fluid mechanics to generate water and smoke [13], [15].

Various researchers have developed different techniques that generate pleasing results. Some animations from commercials and film are of truly amazing quality. Yet there is room for improvement and further research. It is known that contact resolution for simulation is a nontrivial problem if there is friction to be modeled [3], [20], [34]. In fact, it has been reported as the bottleneck in many simulation packages if the number of contacts rises greatly [4], [18], [27], [28]. Our work focuses on simulations with a very large number of contacts, such as in stacks, piles, or otherwise "crowded" arrangements.

- *H. Schmidl is with the Department of Mathematics and Computer Science, North Carolina Central University, Robinson Science Bldg., 1801 Fayetteville St., Durham, NC 27707. E-mail: hschmidl@wpo.nccu.edu.*
- *V.J. Milenkovic is with the Department of Computer Science, University of Miami, PO Box 248154, Coral Gables, FL 33124-4245. E-mail: vjm@cs.miami.edu.*

Manuscript received 20 Sept. 2002; revised 3 Apr. 2003; accepted 17 July 2003.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number 117426.

Most simulators use the modular design depicted in Fig. 1. We propose here a new suite of algorithms for the last stage: responding to and resolving contacts. First, a standard sequential algorithm responds to collisions, but only partially. This algorithm is halted after it has provided sufficient realism and information about moving bodies but before it has used up too much running time. Second, the solution to a single convex QP (positive semidefinite quadratic program) finishes resolving the collisions. Third, the solution to another convex QP resolves the static contacts: persistent contacts with pressure but no normal velocity. Fourth, stationary bodies are detected and "frozen." Freezing reduces the number of contacts in Steps two and three of the next iteration's contact response, greatly increasing their efficiency.

This contact resolution suite can make tractable the simulation of 1,000 cubes falling through an hourglass. We have found no evidence of other approaches solving similarly large and computationally demanding simulations. We demonstrate and test the new contact suite by combining it with our previous algorithm for position update [24], [37]. However, any existing software package that aims at modeling a high number of concurrent contacts can use our algorithms for its own benefit by replacing its collision handling module.

The remainder of this paper is structured in the following way: Section 2 gives a brief summary of existing techniques. Section 3 introduces the notation in this paper and physical concepts. Section 4 presents the QP-based impulse approach for both dynamic (collision) and static contact resolution (Steps two and three). Section 5 adds partial sequential collision resolution (Step one) and freezing (Step four) to improve both realism and efficiency. Section 6 describes experiments which have been conducted. We provide performance analysis and some implementation detail. Section 7 concludes this paper.

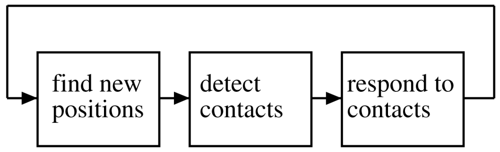


Fig. 1. The simulation loop.

2 Background

This section summarizes existing collision response methods and points out some relevant problems. We adopt the standard terms *collisions* and *static contacts*.

Penalty force methods detect interpenetration and insert a symbolic spring with large spring constant k between the bodies. The spring exerts a repulsive force proportional to the interpenetration depth $\mathbf{d}$, $\mathbf{F} = -k\,\mathbf{d}$, and is deleted immediately as the bodies become receding. Stiff ordinary differential equations resulting from large spring constants can make the problem hard, but recent advances in implicit integration make the problem tractable [5]. Mirtich got good results for static force calculation using stiff integration [27].

Analytical methods are based primarily on a physical *complementarity* principle. Either the normal force or normal acceleration at a contact must be zero: If the bodies are pressing on each other, they are not accelerating away from each other, and vice versa. Lötstedt [20], [21] was the first to formulate analytic contact resolution as an LCP [12], [14]. Considerable work has been done since then [2], [4], [7], [18], [25], [26], [35], [39], [42], much of it addressing infinities and/or ambiguities that arise in some degenerate cases.

The paradigm of impulse-based contact resolution was introduced by Mirtich [25], [26]. It addresses the problems of ambiguous or infinite force solutions by completely avoiding the calculation of forces. An advantage of using the principle of impulse and momentum for collision problems is that the solution is algebraic. Contact forces require integration, which destroys this feature [9]. Mirtich describes a simulator, called *Impulse*, that handles collisions and contacts entirely with the application of impulses. It is shown that both collisions and static contacts with Coulomb friction can be modeled with impulses.

Moreau [29], [30], [31], [32] introduced the concept of single convex QP contact response. This QP is based on Gauss's principle of least deviation. Moreau's work is presented in a theoretical context and assumes single collisions. There is no evidence of an actual implementation and example simulations. In contrast, our approach is inspired by practicality and application.

Also, Baraff formulated QP approaches to contact resolution [2], [3], but stated that they were impractical [3], [4]. In previous work, the authors [24] also applied QP to animation of rigid bodies. However, this work focused on the position update and only used ad hoc methods for resolving collisions and static contacts. In particular, the collision resolution is unrealistic on small simulations such as the "office toy." The new contact suite is both much more realistic and much faster. For our experiments (Section 6), we combine the previous work on position update with the new contact suite. However, the new contact suite could be similarly "plugged into" any simulation system.

3 Physics of Contacts

We will now introduce important rigid body concepts and also the physics of collision resolution with impulses. Along the way, the notation in our paper will be explained.

3.1 Rigid Body Physics

Any text on classical mechanics can give a good introduction on rigid body concepts, e.g., Goldstein [17]. A rigid body has mass m, inertia $\mathbf{I}$, position $\mathbf{x}$, and orientation $\mathbf{R}$. Any point $\mathbf{q}$ in body coordinates can be calculated in world coordinates with $\mathbf{q}_{\text{world}} = \mathbf{x} + \mathbf{R}\mathbf{q}$. The body has velocities $\mathbf{v}$ and ω and corresponding momenta $\mathbf{p} = m\mathbf{v}$ and $\ell = \mathbf{I}\omega$. For two contacting bodies $\mathbf{A}$ and $\mathbf{B}$, we calculate a right-handed collision coordinate frame. The normal $\mathbf{n}$ points from $\mathbf{A}$ to $\mathbf{B}$ and vectors $\mathbf{n}_x$ and $\mathbf{n}_y$ span the tangential collision plane, perpendicular to $\mathbf{n}$. Note: Multiple contacts between bodies can be handled in the obvious manner by using an additional subscript.

3.2 Analytic Impulses

People expect rigid bodies to "bounce" as a result of collisions. The idea of solving collisions with impulses is based on the concept of restitution. It was originally defined by Newton [33] who observed a constant, material dependent ratio between rebound and incidence velocities for two colliding spheres:

$$v^+ = -\epsilon v^-. \tag{1}$$

The coefficient of restitution $\epsilon \in [0, 1]$ determines how elastic the collision is [41].

Newton's model is also called kinematic definition of restitution [22]. It is well-known, it is simple to implement, and will be used in our contact suite. Although it was not intended for multiple point three-dimensional collisions, we find its application yields realistic-looking animations at reasonable cost. Implied physical inconsistencies cannot be detected by the human eye because the exact value for ϵ is not known [38].

For simplicity, we will first present the frictionless case in this section and then modify it to deal with friction in the next section. See the literature [1], [2], [37] for explanations of point velocity, contact separation, and relative contact normal velocity. We follow the literature by defining a collision as a contact with $v^- < 0$ and a static contact with $v^- = 0$. For each collision, we calculate equal but opposite impulses $\mathbf{j}$ using Newton's empirical model [9], [41], which are applied to the bodies at a contact point in order to instantaneously make the bodies receding ($v^+ \geq 0$).

Assume now that we have two bodies $\mathbf{A}$ and $\mathbf{B}$. Subscripts a and b designate properties of bodies $\mathbf{A}$ and $\mathbf{B}$, respectively. The two bodies contact at a point $\mathbf{q}_{ab}$. The contact levers are calculated by subtracting the center of mass coordinates from the contact, i.e., $\mathbf{r}_a = \mathbf{q}_{ab} - \mathbf{x}_a$ and $\mathbf{r}_b = \mathbf{q}_{ab} - \mathbf{x}_b$. Without friction, an impulse acts in the direction of the contact normal $\mathbf{n}$ which points out of body $\mathbf{A}$ and into body $\mathbf{B}$. Therefore, for an impulse with magnitude j, let $-j\mathbf{n}$ be the impulse on body $\mathbf{A}$ and $j\mathbf{n}$ the impulse on body $\mathbf{B}$. The following familiar formula computes the impulse magnitude j [1], [37]:

$$j = \frac{-(1+\epsilon)v^-}{m_b^{-1} + m_a^{-1} + \mathbf{n}\cdot\left(\mathbf{I}_b^{-1}(\mathbf{r}_b \times \mathbf{n})\right)\times\mathbf{r}_b + \mathbf{n}\cdot\left(\mathbf{I}_a^{-1}(\mathbf{r}_a \times \mathbf{n})\right)\times\mathbf{r}_a}. \tag{2}$$

All body properties in (2) are known and we can easily calculate the impulse magnitude j. We know, in any case, $j \geq 0$, i.e., the impulse always pushes the bodies apart and never pulls the bodies together.

3.3 Impulse with Coulomb Friction

Friction is pervasively present in mechanical systems and a usual consequence of it is the slow down of their movement. Coulomb's friction law is the simplest physical model for dry friction. It relates the contact normal force to the tangential, frictional force by a friction coefficient μ. Coulomb friction is widely used in many applications [2], [6], [8], [10], [11], [24], [25], [37]. Coulomb's friction law relates forces but will be applied to impulses throughout the developed suite due to its simplicity. This leads to useful results if visual appearance is the main concern. Arising physical inconsistencies are not detected by the human eye because μ is not known exactly.

With the orthonormal collision coordinate frame $(\mathbf{n}_x, \mathbf{n}_y, \mathbf{n})$ from Section 3.1, we express the total contact impulse:

$$\mathbf{j} = (j_x, j_y, j) = j_x\mathbf{n}_x + j_y\mathbf{n}_y + j\mathbf{n}. \tag{3}$$

Coulomb friction simply relates the tangential, frictional impulse $\mathbf{j}_t = j_x\mathbf{n}_x + j_y\mathbf{n}_y$ to the impulse acting normal to the touching surfaces $\mathbf{j}_n = j\mathbf{n}$ [19]. It states that the magnitude of $\mathbf{j}_t$ can be maximally a friction factor μ times the normal component's magnitude:

$$|\mathbf{j}_t| \leq \mu|\mathbf{j}_n| \qquad \leftrightarrow \qquad j_x^2 + j_y^2 \leq \mu^2 j^2. \tag{4}$$

This inequality geometrically describes the inside of a cone and is commonly called the *friction cone*.

Typically [2], [25], [37], when two bodies **A** and **B** collide with nonzero tangential velocity, one expects a slow down of the tangential motion. Therefore, the direction of the tangential component of the "bounce impulse" is chosen to oppose the tangential velocity. The magnitude is set to μ times the magnitude of the normal impulse, as calculated in the previous section:

$$\mathbf{j}_t = -\mu|\mathbf{j}_n||\mathbf{v}_t|^{-1}\mathbf{v}_t, \tag{5}$$

where $\mathbf{v}_t = (v_x, v_y) = v_x\mathbf{n}_x + v_y\mathbf{n}_y$ is the tangential part of the collision velocity $\mathbf{v}_{ab}$ between **A** and **B**. This method calculates a frictionless normal impulse and adds friction later via (5). Despite this nonphysical simplification, its use for animation can be justified because it generates visually pleasing results.

Alternatively, we can write down an extension of Newton's model that also takes into account tangential, frictional impulse components:

$$j = \frac{-(1+\epsilon)v^-}{m_b^{-1}\mathbf{J} + m_a^{-1}\mathbf{J} + \mathbf{n}\cdot\left(\mathbf{I}_b^{-1}(\mathbf{r}_b \times \mathbf{J})\right)\times\mathbf{r}_b + \mathbf{n}\cdot\left(\mathbf{I}_a^{-1}(\mathbf{r}_a \times \mathbf{J})\right)\times\mathbf{r}_a} \tag{6}$$

with $\mathbf{J} = -\mu|\mathbf{v}_t|^{-1}v_x\mathbf{n}_x - \mu|\mathbf{v}_t|^{-1}v_y\mathbf{n}_y + \mathbf{n}$. Although the two approaches of (5) and (6) will generate slightly different motion, each is equally plausible. Exact simulation would require exact knowledge of ϵ and μ.

4 QP-Based Collision Resolution

This section describes how to resolve collisions with a single convex QP[1] and how to similarly resolve static contacts with a single convex QP. In terms of the entire contact suite, these two algorithms come second and third and will resolve collisions and static contacts even without Steps one, partial sequential collision resolution, and four, freezing of stationary bodies. However, the first and the fourth step only make sense in the context of Steps two and three.

4.1 Step 2: Pure QP-Based Momentum Update

The concepts of Sections 3.2 and 3.3 can be implemented and solved as a QP. Collisions with Coulomb friction have been well researched [8], [9], [19], [36], [41]. The two parameters for our impulses are the friction coefficient μ and the coefficient of restitution ϵ. Both can be understood as material constants for the purpose of simulation. We reformulate the Coulomb model from Section 3.3:

$$(\mathbf{n}_x\cdot\mathbf{j})^2 + (\mathbf{n}_y\cdot\mathbf{j})^2 \leq \mu^2(\mathbf{n}\cdot\mathbf{j})^2 \quad \text{and} \quad \mathbf{n}\cdot\mathbf{j} \geq 0. \tag{7}$$

This constraint is convex but nonlinear. A QP has a nonlinear objective but must have linear constraints. To formulate a QP, we have to linearize the friction cone. Specifically, we approximate the cone with a polygonal pyramid. Currently, we use an eight-sided pyramid (as do Stewart and Trinkle [39]). To do so, we inscribe a polygon with eight equal edges into the base of the pyramid, as seen in Fig. 2b, yielding an eight-sided pyramid as in Fig. 2c. The collision frame has its origin centered at the cone tip and the collision normal $\mathbf{n}$ points along the axis of the cone or pyramid. The original nonlinearized cone (Fig. 2a) has slope $1/\mu$.[2] the linearized cone's sides have slope $(\mu\cos\pi/8)^{-1} > \mu^{-1}$. To confine the impulse $\mathbf{j}$ to the inside of the linearized friction cone, we calculate inward pointing normal vectors to the cone sides

$$\mathbf{u}_h = \mu\cos\frac{\pi}{8}\mathbf{n} + \cos\frac{\pi h}{4}\mathbf{n}_x + \sin\frac{\pi h}{4}\mathbf{n}_y, \tag{8}$$

for $h = 0, 1, 2, \ldots, 7$.[3] The impulse $\mathbf{j}$ lies inside the linearized cone if and only if it lies inside all these sides:

$$\mathbf{u}_h\cdot\mathbf{j} \geq 0, \quad \text{for} \quad h = 0, 1, 2, \ldots, 7. \tag{9}$$

1. This paper only considers QPs with a positive-definite objective, which are in P-time [16] and for which there is excellent commercial software.
2. The right circular cone $z^2 = s^2(x^2 + y^2)$ has slope s.
3. For convenience, $\mathbf{u}_h$ is not unit-length, but its $\mathbf{n}_x, \mathbf{n}_y$ component is.

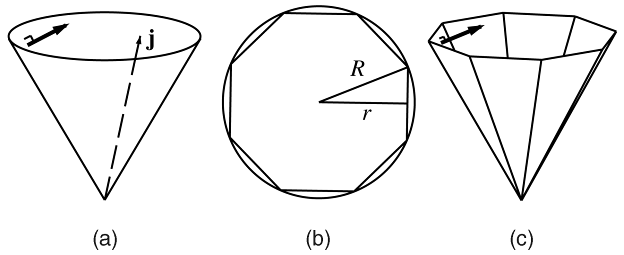


Fig. 2. Friction cone and regular octagonal approximation. Ratio r/R of inner to outer radius is $\cos \pi/8$.

We now have eight linear inequalities per collision, which can be added as constraints to a QP to implement the Coulomb friction model.

We also have to address restitution by bouncing. We follow Newton's impact model from Section 3.2 and calculate the (scalar) relative contact normal velocity $v_\perp$ between bodies $\mathbf{A}$ and $\mathbf{B}$ with contact levers $\mathbf{r}_a$ and $\mathbf{r}_b$ [1], [37]:

$$v_\perp = \mathbf{n} \cdot ((\mathbf{v}_b + \omega_b \times \mathbf{r}_b) - (\mathbf{v}_a + \omega_a \times \mathbf{r}_a)). \tag{10}$$

An impulse $\mathbf{j}$ acts on a body by changing its linear and angular momentum. The impulse $\mathbf{j}$ between bodies $\mathbf{A}$ and $\mathbf{B}$ adds to $\mathbf{v}_b$ and ω_b

$$\mathbf{v}_b \leftarrow \mathbf{v}_b + m_b^{-1}\mathbf{j} \qquad \text{and} \qquad \omega_b \leftarrow \omega_b + \mathbf{I}_b^{-1}(\mathbf{r}_b \times \mathbf{j}), \tag{11}$$

and subtracts from $\mathbf{v}_a$ and ω_a

$$\mathbf{v}_a \leftarrow \mathbf{v}_a - m_a^{-1}\mathbf{j} \qquad \text{and} \qquad \omega_a \leftarrow \omega_a - \mathbf{I}_a^{-1}(\mathbf{r}_a \times \mathbf{j}). \tag{12}$$

The relative contact velocity before impact is a constant and can be easily calculated with (10). The relative contact velocity after impact can be calculated depending on $\mathbf{j}$ with (10), (11), and (12): Just express the velocities after impact with (11) and (12) and plug these into (10).

Let v^- and v^+ denote the relative contact normal velocity before and after any impulses are applied (and $\mathbf{v}_a$, ω_a, $\mathbf{v}_b$, and ω_b have been updated). In the following, we adopt some of Baraff's ideas [2]. If the bodies were not really colliding before the collision ($v^- \geq 0$), then they must still not be colliding after the collision, else they must semi-elastically "bounce." Baraff determined that the following applies generally, even to nonpoint masses:

$$\text{if } v^- \geq 0 \text{ then } v^+ \geq 0 \text{ else } v^+ \geq -\epsilon v^-. \tag{13}$$

Using quadratic programming, we simultaneously solve for each impulse $\mathbf{j}$ at each contact. The QP has six variables per body: the components of $\mathbf{v}$ and ω. It has four variables per contact: the components of $\mathbf{j}$ and the collision normal velocity v^+. It has six equality constraints per body to express how each body's $\mathbf{v}$ and ω change as a result of impulses according to (11) and (12). It has one equality constraint per contact to relate v^+ to the updated body velocities according to (10). It has eight inequalities per contact to constrain $\mathbf{j}$ to the linearized friction cone according to (9). Finally, it has one more linear inequality per contact to implement the bounce according to (13).

The objective of the QP is the total kinetic energy after application of impulses, which is a positive definite quadratic function of the updated $\mathbf{v}$s and ωs. For i, the index over all bodies, we have the objective:

$$\sum_i \frac{1}{2}\, m_i \mathbf{v}_i^2 + \frac{1}{2}\, \omega_i^T \mathbf{I}_i \omega_i. \tag{14}$$

Thus, we ask the system to dissipate a maximum amount of kinetic energy as a result of friction. This produces plausible results, although the maximum energy dissipation principle applies to forces, not impulses, and it is not directly equivalent to minimization of the system's total kinetic energy. Explicit maximum energy dissipation results in a very stable simulation which brings a slowly moving system to rest. The slow down of the system movement will be seen as realistic as the exact friction and restitution coefficients are not known.

4.2 Step 3: Impulsive Static Contact Response

After collisions are resolved with bounces, resulting static contacts ($v^+ = 0$) must have contact forces computed to prevent bodies from sinking into each other. We experimented with several methods to calculate forces and verified that it is a nontrivial problem. Mirtich's work used impulsive forces exclusively [25], [26] and other researchers have stated that impulsive forces can be applied when nonimpulsive contact forces are not easily calculated [3], [40]. We realized that our pure QP momentum update can be modified and used in the vein of an impulse-based approach to resolve static contacts very reliably, efficiently, and extremely stably.

Under true physics, the force of gravity accelerates the bodies. A force arises at each contact that provides a nonnegative relative contact normal acceleration $a_\perp \geq 0$. Since the impulse update ensures nonnegative relative normal velocity $v_\perp \geq 0$ at the contacts, the integration of $a_\perp$ ensures that $v_\perp$ remains nonnegative. Contact forces do no positive work on the bodies, i.e., they do not increase the body energies. In other words, the contact forces are "just strong enough" to prevent negative contact normal acceleration.

We mimic this behavior as follows: First, our algorithm integrates the accelerations without regard to contacts: add Δt times the acceleration of gravity to the linear velocity of each body. At this point, the relative contact normal velocities may have changed from nonnegative to negative. The algorithm applies small impulses at all contacts to make all relative contact normal velocities nonnegative again. It does so by solving a QP that is the momentum update QP with restitution ϵ set to zero, but otherwise with the same objective and constraints. Consider the example of a single body at rest on the ground. Initially, its relative contact normal velocity $v_\perp$ is zero. As a result of gravity, $v_\perp < 0$. The QP generates an impulse at the contact sufficiently strong to set $v_\perp$ equal to zero again. Since $\epsilon = 0$, it will not "bounce up." Since the QP minimizes kinetic energy, it will not give the body a spurious tangential impulse nor will it otherwise violate the laws of physics. The tangential impulse will at the most diminish tangential contact velocity, i.e., until the contact has made its transition from sliding to sticking. This reduces our system to impulse-based physics and works very well in practice. The simulations are stable and efficient.

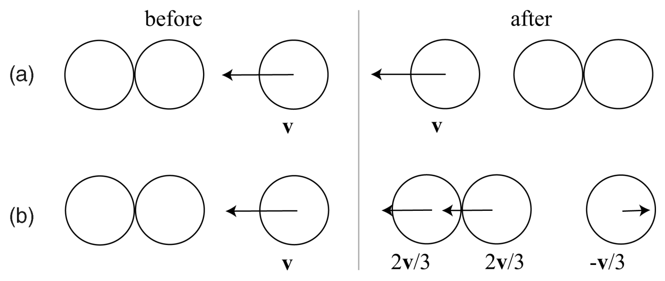


Fig. 3. Explanation of momentum conservation.

5 Increasing Realism and Speed

This section presents Steps one and four of the contact resolution suite. The first step is a partial sequential collision response. This is necessary for realism in small examples and also to determine moving bodies for the fourth step, freezing (Section 5.2). Freezing, in turn, greatly increases the speed of Steps two and three, QP collision resolution and QP static contact resolution, which have been described in the previous section. Obviously, freezing does not help Steps two and three in the *current* time-step: It speeds up Steps two and three of the *next* time-step.

5.1 Step 1: Partial Sequential Collision Resolution

Fig. 3a depicts the expected behavior of three aligned balls with equal mass that collide elastically: The right, incoming ball stops and only the leftmost ball bounces off. In contrast, the QP-based simultaneous impulse algorithm of Section 4.1 generates the output shown in Fig. 3b: Although the total momentum is conserved, all balls are in motion after the collision [2].

The standard solution to this problem is sequential application of impulses [1], [25], [26]. All colliding contacts are placed in a priority queue ordered by collision velocity. The following two steps are repeated until all contacts separate ($v^+ \geq 0$). 1) The fastest collision (most negative v^-) is dequeued and resolved according to the impulse response of Section 3.3. 2) All other contacts involving either of the two colliding bodies have their velocities updated.

Unfortunately, PQ (priority queue-based sequential collision resolution) does not scale well to large problems [2]. We avoid this problem and stop running PQ before it takes too much time. Instead, in Step two of the contact suite, the QP-based algorithm finishes the collision resolution. We can specify a number c_b of bounces to be resolved sequentially that ensures the office toy, or some other specific scene, is modeled properly. For s, the number of spheres in the office toy, $c_b = s - 1$ sequential bounces will produce the desired outcome.

Specifying c_b has another advantage. Solving a QP has a certain overhead even if there are only a few true collisions ($v^- < 0$). Invoking the QP solver, building the constraints, and solving the QP comes at a cost. For only a few collisions, PQ is much faster. If the number of actual collisions is less than c_b, PQ alone will finish the bouncing very cheaply. Therefore, we call the QP momentum update only if necessary. Currently, we use a heuristic to find the value of c_b: It usually works well to set c_b equal to the number of bodies for scenes with fewer than 50 bodies and equal to 10 percent of the number of contacts for scenes with more than 50 bodies. Unless one wants to create the office toy with more than 50 bodies, this heuristic will create plausible animations. For other scenes with a large number of bodies, not all collisions will be in the same direction. Consequently, the scene is too complex for a human to accurately make predictions. Hence, our heuristic provides sufficient realism, yet it maximizes performance.

5.2 Step 4: Freezing

Steps one, two, and three of the contact resolution suite can efficiently generate stable simulations of several hundred bodies with a high number of concurrent contacts. Yet we found them to be too slow when the number of bodies is raised to 1,000. The running time of the algorithm is determined by the solution time for the impulse QPs and, thus, their individual sizes. The latter clearly depends on the number of contacts. This section presents a technique for reducing the number of contacts that we call “freezing.” This fourth step in the contact suite trades away a tiny amount of realism (if any in practice) in exchange for a tremendous increase in speed.

We observed that much computation goes into calculation of static contact response once bodies settle into a stable arrangement. Although the motion has stopped for a settled stack of bodies, contacts have to be resolved at every time step to keep the bodies from sinking into each other. Freezing acts to remove from consideration contacts between stationary bodies. We distinguish *fixed*, *dynamic*, and *frozen* bodies. Walls are examples of fixed bodies. Dynamic, free moving bodies can become frozen and vice versa.

Lack of motion is defined by a heuristic which compares the current kinetic energy ($E_{lin} + E_{rot}$) for a body to the kinetic energy it would pick up in free-fall starting at rest:

$$\frac{1}{2}m\mathbf{v}^2 + \frac{1}{2}\boldsymbol{\omega}^T\mathbf{I}\boldsymbol{\omega} < \frac{\mathbf{p}_g^2}{2m}, \tag{15}$$

where $\mathbf{p}_g = m\mathbf{g}\Delta t$ is the momentum a body with mass m picks up during a time step $\Delta t = 1/30$ sec as a result of gravity.[4] the freezing algorithm allocates a counter to each body, initially zero. If (15) is satisfied and if the body was in contact with a fixed or currently frozen body, increment the counter. Otherwise, reset it to zero. If a counter exceeds c_f, freeze the body. For purposes of collision and contact handling and setting up the QPs in Steps two and three, ignore contacts between frozen bodies or between a frozen body and a fixed body. Also, a frozen body does not pick up momentum through gravity or any other external potential and stays at its current position. The only way for a frozen body to get free again is by picking up momentum through a collision which increases its sum of energies about the threshold of (15), in which case its counter is reset to zero.

Note. 1) If PQ (Step one) is applied, then (15) is also checked for each body in each sequential collision. If it is violated, the respective body’s counter is reset to zero and it is

4. For simplicity, our only external potential is gravity but others can be added.

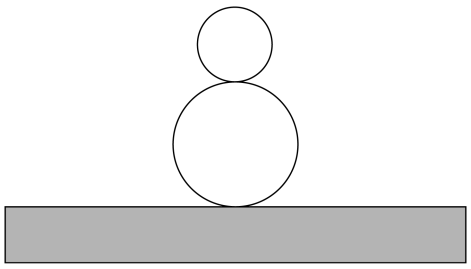


Fig. 4. An unstable configuration: Care must be taken when freezing is used.

unfrozen if applicable. 2) After static contact response (Step three), every body which is designated as frozen has its momentum set to zero. This prevents unnatural "drifting" in the position update.

Consider a single cube sitting on a solid, static surface. Without freezing, the cube picks up momentum from gravity during each time step, interpenetration has to be prevented and static contact has to be resolved just to keep the cube at a visually resting state. If (15) was satisfied while the cube was in contact with the static surface, its counter is incremented. If the counter exceeds c_f, we *freeze* the cube. This now frozen cube will no longer pick up energy through gravity; it stays at its current position and the contacts between the frozen cube and static surface will not be considered in the QP collision response (Step two) and the QP static contact response (Step three). For this simple example, the number of effective contacts has shrunk to zero, yet interpenetration is prevented. The simulation has virtually zero runtime.

Now, consider a more interesting scene with many cubes and a fixed container. We define "cold" volumes in which the cubes can freeze. For example, the cold volume could be the entire lower container portion. Once the cubes settle in the cold volume, we examine their energies according to (15). Each cube maintains a counter, which we set appropriately. If a counter exceeds c_f, we freeze the cube. Again, a frozen cube stays at its current position; it no longer picks up energy from gravity nor will its contacts with fixed or frozen bodies require immediate resolution.

Since freezing freezes bodies which are very likely to move very little, it has little effect on realism. However, consider the example in Fig. 4. A sphere is resting on a table with another sphere on it. This is clearly an unstable configuration and contact between the spheres should not persist. Instead, one would expect the top sphere to roll off after a short time. Care must be taken in selecting a high enough value for c_f if freezing has to model this scene correctly. Selecting c_f too small will allow the upper sphere to be frozen, maintaining the unstable configuration indefinitely. However, if the value of c_f is chosen large enough, the top sphere has enough chance to start its rolling and the contact will not be frozen unrealistically.

Only Steps two and three of the contact suite are truly essential, but, if freezing (Step four) is applied, then it is best that PQ (Step one) also be applied. Consider the situation of Fig. 5 with a frozen pair of contacting bodies **A** and **B**. A third body **C** bounces against **B**. The QP momentum update will unfreeze only body **B** because it resolves collisions simultaneously. The contacts between **A** and **B** are inactive and will not be processed in the QP. Body **A** cannot be unfrozen until the contact resolution after the next position update. In contrast, the PQ will first unfreeze **B** and then unfreeze **A**. For a large cluster of frozen bodies, the number of unfrozen bodies is dependent on the limit of bounces c_b after which we terminate the PQ. However, if c_b is selected such that PQ uses about 1 percent of the running time for a large simulation, PQ appears to unfreeze virtually all the bodies that should be unfrozen. Hence, it is easy to attain a simulation that is both lively and realistic yet efficient. In conclusion, freezing provides an effective means for speeding up the simulation with little or no visible loss of realism.

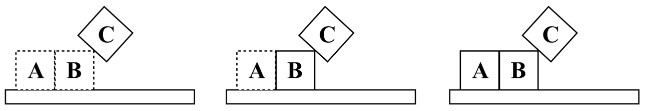


Fig. 5. Frozen bodies (dashed) are revived one at a time.

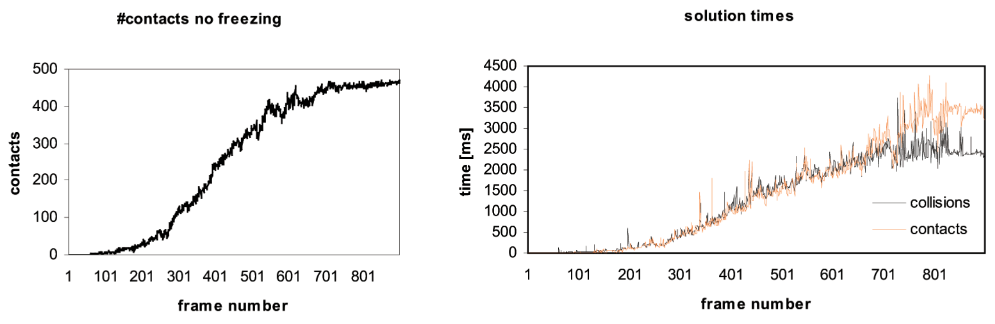
6 Experiments, Analysis, and Implementation

We constructed scenes that simulate stacks or otherwise "crowded" arrangements of bodies. These scenes are nontrivial to simulate and show the advantages of our proposed techniques due to their high number of contacts. We simulate scenes whenever possible with and without freezing of bodies for comparison. Our experiments were conducted by combining the OBA position update [24], [37] and the new contact suite. OBA position update assumes convex object geometry. This is the reason why we used convex spheres and cubes in our simulations. However, the contact suite's applicability is not limited to convex objects only. It can be employed to resolve collisions for general objects. Page limits prevent us from including color plates. Still images and a video are available at http://www.cs.miami.edu/~harald/ics/ics.html.[5]

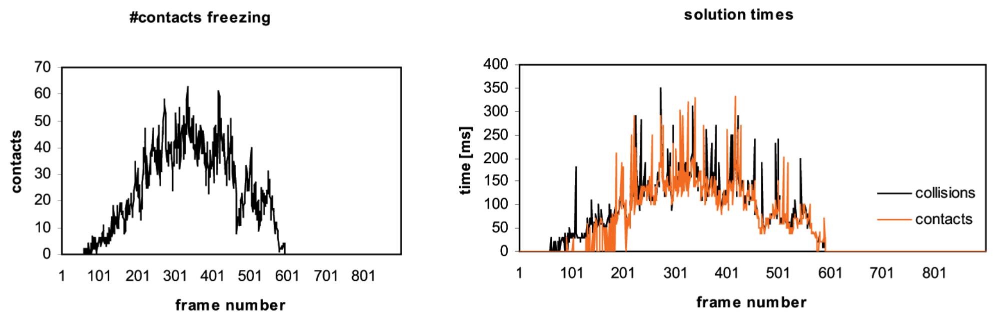
The "stack" simulation simply shows how 10 cubes get stacked on top of each other. The "cubejam" scene has 100 cubes, which fall into a container with obstacles. This scene is somewhat more complicated, but, in essence, also here the final arrangement is a stack. In addition, "c-s stack" simulates a stack that mixes cubes and spheres. Our findings show that freezing compares favorably to simulations that do not make use of it. We also simulated the "office toy" to demonstrate the correct transfer of momentum. This last scene makes no use of freezing because each body would be unfrozen during each iteration anyway.

We provide some implementation detail to illustrate our following performance analysis. The CPLEX 7.0 runtime library was used for solving the QPs in our experiments. All examples were coded in Java on a Pentium III PC with 450MHz running Windows NT. Our initial experiments without freezing limited the number of bodies which we could simulate in acceptable time to just a few hundred.

5. The authors wish to thank Adam McMahon for rendering and generation of the video.



(a)



(b)

Fig. 6. Number of contacts and QP solution times (a) without and (b) with freezing.

Milenkovic has simulated an hourglass with 1,000 spheres using position-based physics [23]. The OBA algorithm of Milenkovic and Schmidl [24], [37] improved Milenkovic's earlier work: It can simulate bounces, Newtonian trajectories (second order physics), and friction. However, even here the 1,000 sphere hourglass was frictionless. With our freezing method, we are able to simulate a *frictional* hourglass with 1,000 *cubes*. Frictionless spheres have only three degrees of freedom, whereas frictional cubes have six. Furthermore, two touching spheres have only one contact point, whereas two cubes can have up to eight if they are stacked and slightly rotated. Also, since the "cubeglass" has friction, the cubes temporarily wedge in the funnel opening, clogging the downward flow of cubes. Hence, contacts persist longer than in Milenkovic's older "sphereglass." Simulating the "cubeglass" is more difficult than the "sphereglass" for these reasons. There is no experimental evidence of similarly complicated simulations by other research.

It is clear that the overall performance of our impulse algorithm is governed by how fast we can solve each individual QP. In general, this depends on the number $m = 6n + 10k$ of constraints in the QP, where n is the number of bodies and k is the number of contacts. The number of contacts depends on the body geometry and is roughly proportional to the number of contacting bodies. For a tight packing of n bodies, the number m of constraints is therefore proportional to n. Fig. 6 shows the development of number of contacts and solution times for the collision and contact QPs, with and without freezing, for the "cubejam" simulation. Without freezing (Fig. 6a), the number of contacts grows slowly up to its maximum at about 500 contacts per frame. The solution time for the two QPs solved develops in correlation with the number of contacts up to a maximum. (Interestingly, although practically the same QP is solved for collision and contact resolution, the solver spends more time on the contact QP.)

With freezing (Fig. 6b), we observe a similar correlation between solution time and number of contacts. As bodies settle, they lose most of their kinetic energy and are frozen. Consequently, the effective number of contacts is diminished and tends toward zero. The number of contacts stays reasonable at all times during the simulation until all bodies are frozen and no more contacts remain to be treated. We observe a strongly diminished time for handling collisions and contacts.

Table 1 makes a comparison between scenes with and without the freezing technique. We examine the solution time per QP for the collision and contact update and find a tremendous speed-up by up to a factor 28. For the "c-s stack," the speed-up is the smallest. Simulation of spheres is a priori at a lesser cost because involved computations are cheaper. Therefore, the gap between the two timings is smaller here. It is clear that freezing reduces the number of static contacts drastically. Therefore, also the number of frames with static contacts and, hence, the number of QPs solved for contact resolution is reduced at a higher rate than for collisions. This is the reason why an overall larger speed-up is achieved for static contact handling.

The final simulations are as plausible as without freezing. Assuming this factor, it should be clear why we cannot provide the same comparison for the "cubeglass." The contacts per frame are averages. The actual number peaks for the "cubeglass" at over 4,000 contacts per frame.

TABLE 1
Compare Collision QPs (coQP) and the Static Contact QPs (scQP) with (f) and without Freezing (nf)

	Stack	Cubejam	C-S stack	Cubeglass
avg. contacts/frame (nf/f)	28/1	249/15	13/2	—/528
avg. time/coQP (nf/f)	0.18/0.012	2.6/0.12	0.1/0.04	—/12
coQPs solved (nf/f)	489/145	822/513	478/388	—/493
speed-up	15	22	2.5	—
avg. time/scQP (nf/f)	0.2/0.008	2.8/0.1	0.1/0.03	—/11
scQPs solved (nf/f)	480/96	749/428	469/344	—/607
speed-up	25	28	3	—

Timings are in seconds.

The contact geometry of the "cubeglass" compares with the "cubejam." Therefore, it is safe to assume similar factors for speed-up. With freezing, the program spent a total of 9.5 hours for updating collisions and contacts. Hence, simulation of collisions and contacts for the "cubeglass" without freezing would have taken almost 10 days.

Note. Although PQ (Step one) takes a small and very predictable amount of time, it can sometimes greatly reduce the running time of Step two or eliminate the need for it entirely. For this reason, the experiments to determine the speed-up from freezing always skip Step one. This reduces the variability in the running times and provides a more accurate estimate of the speed-up factor. For a realistic result, one would always apply PQ (Step one) when using freezing (Step four) (see Section 5.2).

For the "stack," we set $c_f = 1$ for freezing. In contrast, we have to use a higher threshold for the "cubejam." As cubes squeeze around the obstacles, they come to a rest only seemingly due to friction although they are still far from the bottom. We had to set $c_f = 3$ and declare a cube then as frozen.

7 Conclusions and Future Work

We presented a method which is easily implemented and efficient. Despite trade offs made for efficiency, they exhibit no sign of visually lacking realism. Our model makes approximations and uses physically inspired models for efficiency purposes when possible. Contacts and collisions are resolved with impulses only. This follows Mirtich's *impulse* simulations and addresses the findings of other researchers who noted that impulsive forces are necessary whenever true contact forces cannot be calculated. The simulations are stable. Stability is an important trait of any simulation algorithm. The QP-based, impulsive contact suite can be used in any existing simulator that follows a modular structure as in Fig. 1. Partial sequential collision response (Step one) ensures realistic "liveliness" at a modest cost. Freezing (Step four) allows tremendous speed-up of rigid body simulation, making the simulation of 1,000 cubes in an hourglass tractable. Collisions and static contacts are each solved with a single convex QP. The entire contact suite is implemented by solving only two successive convex QPs.

Traditional PQ bouncing can increase kinetic energy, which is not realistic. On the one hand, the energy minimization QP always ensures that the laws of thermodynamics are not violated. On the other hand, its simultaneous resolution of collisions is not realistic for examples like the office toy. Ideally, one might want to apply the QP technique to *individual* bounces in the PQ algorithm, but, in practice, it would take too much time to set up a QP for each individual bounce. Instead, we are working on methods that allow detection of contacts which actually need to be treated with care in order to avoid increase of energy. We are also considering development of specialized code that solves small QPs very efficiently and without invocation of CPLEX.

References

[1] D. Baraff and A. Witkin, "Physically Based Modeling," *SIGGRAPH '98 Course Notes,* July 1998.
[2] D. Baraff, "Analytical Methods for Dynamic Simulation of Non-Penetrating Rigid Bodies," *SIGGRAPH '89 Conf. Proc.,* pp. 223-232, 1989.
[3] D. Baraff, "Coping with Friction for Non-Penetrating Rigid Body Simulation," *SIGGRAPH '91 Conf. Proc.,* vol. 25, no. 4, pp. 31-40, July 1991.
[4] D. Baraff, "Fast Contact Force Computation for Nonpenetrating Rigid Bodies," *SIGGRAPH '94 Conf. Proc.,* pp. 23-34, 1994.
[5] D. Baraff and A. Witkin, "Large Steps in Cloth Simulation," *SIGGRAPH '98 Conf. Proc.,* pp. 43-52, 1998.
[6] J.A. Batlle, "The Sliding Velocity Flow of Rough Collisions in Multibody Systems," *J. Applied Mechanics,* vol. 63, pp. 804-809, Sept. 1996.
[7] J.A. Batlle and S. Cardona, "The Jamb (Self-Locking) Process in Three-Dimensional Collisions," *J. Applied Mechanics,* vol. 65, pp. 417-423, June 1998.
[8] V. Bhatt and J. Koechling, "Three-Dimensional Frictional Rigid-Body Impact," *J. Applied Mechanics,* vol. 62, pp. 893-898, Dec. 1995.
[9] R.M. Brach, "Rigid Body Collisions," *J. Applied Mechanics,* vol. 56, pp. 133-138, Mar. 1989.
[10] R. Bridson, R. Fedkiw, and J. Anderson, "Robust Treatment of Collisions, Contact and Friction for Cloth Animation," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 594-603, July 2002.
[11] A. Chatterjee and A. Ruina, "A New Algebraic Rigid-Body Collision Law Based on Impulse Space Considerations," *J. Applied Mechanics,* vol. 65, pp. 939-950, Dec. 1998.
[12] R.K. Cottle, J.-S. Pang, and R.E. Stone, *The Linear Complementarity Problem.* Academic Press, 1992.
[13] R. Fedkiw, J. Stam, and H.W. Jensen, "Visual Simulation of Smoke," *SIGGRAPH '01 Conf. Proc.,* pp. 15-22, Aug. 2001.
[14] M.C. Ferris and F. Tin-Loi, "Limit Analysis of Frictional Block Assemblies as a Mathematical Program with Complementarity Constraints," Technical Report 99-01, Computer Sciences Dept., Univ. of Wisconsin, Feb. 1999.
[15] N. Foster and R. Fedkiw, "Practical Animation of Liquids," *SIGGRAPH '01 Conf. Proc.,* pp. 23-30, Aug. 2001.

[16] M. Fu, Z.Q. Luo, and Y. Ye, "Approximation Algorithms for Quadratic Programming," *J. Combinatorial Optimization,* vol. 2, pp. 29-50, 1998.
[17] H. Goldstein, *Klassische Mechanik,* 11th ed. Wiesbaden: AULA Verlag, 1991.
[18] J.K. Hahn, "Realistic Animation of Rigid Bodies," *SIGGRAPH '88 Conf. Proc.,* vol. 22, no. 4, pp. 299-308, Aug. 1988.
[19] J.B. Keller, "Impact with Friction," *J. Applied Mechanics,* vol. 53, pp. 1-4, Mar. 1986.
[20] P. Lötstedt, "Coulomb Friction in Two-Dimensional Rigid Body Systems," *Zeitschrift für angewandte Mathematik und Mechanik,* vol. 61, pp. 605-615, 1981.
[21] P. Lötstedt, "Numerical Simulation of Time-Dependent Contact and Friction Problems in Rigid Body Mechanics," *SIAM J. Scientific and Statistical Computing,* vol. 5, no. 2, pp. 370-393, June 1984.
[22] D.B. Marghitu and Y. Hurmuzlu, "Three-Dimensional Rigid-Body Collisions with Multiple Contact Points," *J. Applied Mechanics,* vol. 62, pp. 725-732, Sept. 1995.
[23] V.J. Milenkovic, "Position-Based Physics: Simulating the Motion of Many Highly Interacting Spheres and Polyhedra," *SIGGRAPH '96 Conf. Proc.,* pp. 129-136, 1996.
[24] V.J. Milenkovic and H. Schmidl, "Optimization-Based Animation," *SIGGRAPH '01 Conf. Proc.,* pp. 37-46, 2001.
[25] B. Mirtich, "Impulse-Based Simulation of Rigid Bodies," *Proc. 1995 Symp. Interactive 3D Graphics,* pp. 181-189, 1995.
[26] B. Mirtich, "Impulse-Based Dynamic Simulation of Rigid Body Systems," PhD thesis, Univ. of California, Berkeley, Dec. 1996.
[27] B. Mirtich, "Timewarp Rigid Body Simulation," *SIGGRAPH '00 Conf. Proc.,* pp. 193-200, 2000.
[28] M. Moore and J. Wilhelms, "Collision Detection and Response for Computer Animation," *SIGGRAPH '88 Conf. Proc.,* pp. 289-297, 1988.
[29] J.J. Moreau, "Quadratic Programming in Mechanics: Dynamics of One-Sided Constraints," *SIAM Control,* vol. 4, no. 1, pp. 153-158, 1966.
[30] J.J. Moreau, "Standard Inelastic Shocks and the Dynamics of Unilateral Constraints," *Unilateral Problems in Structural Analysis,* pp. 171-221, 1985.
[31] J.J. Moreau, "Unilateral Contact and Dry Friction in Finite Freedom Dynamics," *Nonsmooth Mechanics and Applications,* pp. 1-82, 1988.
[32] J.J. Moreau, "Numerical Aspects of the Sweeping Process," *Computer Methods in Applied Mechanics and Eng.,* pp. 329-349, 1999.
[33] I. Newton, *Philosophiae Naturalis Principia Mathematica.* London Reg. Soc. Praeses, 1686.
[34] P. Painlevé, "Sur le Lois du Frottement de Glissemment," *C. R. Acad. Sci. Paris,* vol. 121, pp. 112-115, 1895.
[35] J.S. Pang and J.C. Trinkle, "Complementarity Formulations and Existence of Solutions of Dynamic Multi-Rigid-Body Contact Problems with Coulomb Friction," *Math. Programming,* vol. 73, pp. 199-226, 1996.
[36] M.B. Rubin, "Physical Restrictions on the Impulse Acting during Three-Dimensional Impact of Two 'Rigid' Bodies," *J. Applied Mechanics,* vol. 65, pp. 464-469, June 1998.
[37] H. Schmidl, "Optimization-Based Animation," PhD thesis, Dept. of Computer Science, Univ. of Miami, May 2002.
[38] C.E. Smith and P.-P. Liu, "Coefficients of Restitution," *J. Applied Mechanics,* vol. 59, pp. 963-969, Dec. 1992.
[39] D. Stewart and J.C. Trinkle, "An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Inelastic Collisions and Coulomb Friction," *Zeitschrift für Angewandte Mathematik und Mechanik,* vol. 77, no. 4, pp. 267-279, 1997.
[40] D.E. Stewart, "Rigid-Body Dynamics with Friction and Impact," *SIAM Rev.,* vol. 42, no. 1, pp. 3-39, 2000.
[41] W.J. Stronge, "Rigid Body Collisions with Friction," *Proc. Royal Soc. London,* 1990.
[42] J.C. Trinkle, J.A. Tzitzouris, and J.S. Pang, "Dynamic Multi-Rigid-Body Systems with Concurrent Distributed Contacts: Theory and Examples," *Philosophical Trans. Math., Physical, and Eng. Sciences,* vol. 359, no. 1789, pp. 2575-2593, Dec. 2001.

Harald Schmidl studied physics at the Universität Regensburg and the Universität Tübingen in Germany. He received the MS degree in computer science from the University of Miami, after which he worked as a multimedia software developer. He returned to the University of Miami for his PhD work in an interdisciplinary program of computer science and computer engineering under Victor Milenkovic and graduated in May 2002. He was at the University of North Carolina at Chapel Hill as a postdoctoral researcher and is currently a visiting assistant professor at North Carolina Central University. He is interested in computer graphics, physically-based simulation, and application of computers to foreign areas, such as music or psychology.

Victor J. Milenkovic graduated from Harvard University summa cum laude in mathematics and received the PhD degree in computer science from Carnegie Mellon University under the direction of Takeo Kanade and Daniel Sleator. While on the faculty at Harvard, he received the US National Science Foundation Presidential Young Investigator Award. He founded the Department of Computer Science at the University of Miami where he is currently chair and an associate professor. His areas of research are computational geometry, particularly layout and numerical issues, and computer graphics, with a focus on applications of mathematical programming.