A Fast and Stable Penalty Method for Rigid Body Simulation

Evan Drumwright
University of Southern California
drumwrig@robotics.usc.edu

***Abstract*— Two methods have been used extensively to model resting contact for rigid body simulation. The first approach, the *penalty method*, applies virtual springs to surfaces in contact to minimize interpenetration. This method, as typically implemented, results in oscillatory behavior and considerable penetration. The second approach, based on formulating resting contact as a *linear complementarity problem*, determines the resting contact forces analytically to prevent interpenetration. The analytical method exhibits expected-case polynomial complexity in the number of contact points, and may fail to find a solution in polynomial time when friction is modeled. We present a fast penalty method that minimizes oscillatory behavior and leads to little penetration during resting contact; our method compares favorably to the analytical method with regard to these two measures, while exhibiting much faster performance both asymptotically and empirically.**

I. Introduction

The problem of contact modeling is one of the greatest obstacles to simulating rigid bodies. Tradeoffs must be made between speed, numerical stability, and accuracy. Recent work has approached contact modeling in different ways: solving for contact forces analytically [1], treating collision and resting contact using impulses [2], [3], and formulating the time-stepping equations to respect non-penetration constraints [4]. Nevertheless, these solutions may be too slow for or otherwise inappropriate for some simulation domains. For example, both robotic and haptic simulation require high frequency integration, and the simplest method, the *penalty method* [5] often remains the best solution in such cases.

The penalty method applies a restorative force at the point of deepest penetration between two bodies. This point can be generally be found very quickly, particularly if the geometries of intersection are primitive types. However, the penalty method suffers from two particular drawbacks that often make dynamic simulation difficult. First, applying a separating force can result in oscillatory behavior when two bodies are in resting contact (see Figure 1). Second, the forces generated by the penalty method can be immense if the bodies collide with relatively high velocity, often leading to numerical instability [6]. As a result, very small step sizes and excessively damped implicit integrators are often required.

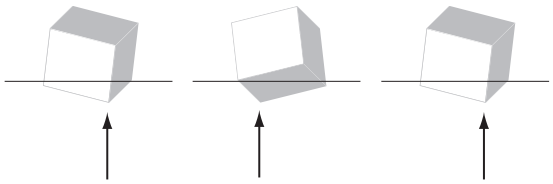


Fig. 1. The result of the application of restoring forces to the point of deepest penetration on successive simulation steps. The cube oscillates on the planar surface as a result.

We present a method for contact modeling that treats impacts with a standard impulse-based method while resting contact is managed using a penalty method that runs in time $O(N \lg N)$ in the number of contacts. Our approach uses multiple contact points and an integral term to achieve greater stability and less penetration than using the deepest-point penalty method. Additionally, our penalty method is competitive in terms of stability and penetration with the analytical approach for resting contact.

We use the word *stability* within this article to refer to resistance to perturbation, as in the dynamical systems sense, rather than numerical stability. However, we are unable to prove stability using convential means (e.g., root-locus method, Lyapunov functions, etc.), because arbitrary forces may be applied to simulated bodies. We do provide an indicator of stability, however, as seen in Section IV.

II. Background

This section provides background on contact and collision modeling in rigid body simulation. It covers several classes of methods and discusses where the particular approaches have been successful and where they are lacking.

A. Separation of methods for contact and collision

Moore and Wilhelms [5] provided some of the earliest work toward combining rigid-body simulation and computer graphics. They introduced a popular paradigm in rigid-body simulation: contacts and collisions are separated using a threshold velocity, and handled using different means. For example, Moore and Wilhelms determine the impulse-velocities[1] for collision handling by algebraically solving a system of 15 equations with 15 variables using conservation of momentum and Newton's law of restitution.[2] For low relative velocity projected along the contact normal, Moore and Wilhelms applied a single virtual spring to mitigate interpenetration.

Moore and Wilhelms' virtual spring method is discontinuous. They apply a restorative force to the point of deepest penetration (or nearest approach), and do not apply force once the objects begin separating (even if still penetrating). This discontinuous method can be readily extended to perform analogously to a proportional-derivative (PD) controller (using zero desired penetration depth and zero desired relative normal velocity) or even a proportional-integrative-derivative (PID) controller, as is done in Section III-B.

B. Analytical solutions to constrained contact

Resting contact can be modeled by determining the appropriate constraining forces analytically. For bilateral (i.e., joint) constraints, a Lagrange-multiplier approach can be used to solve for the constraining forces in $O(n)$ time [8] (n is the number of links in the loop-free, articulated body). Unfortunately, unilateral constraints cannot be resolved in this manner. The principal way of determining contact forces is by formulating the system as a *linear complementarity problem* (LCP) [9]. A linear complementarity problem takes the form:

$$\begin{aligned} \mathbf{A}\mathbf{x}+\mathbf{b} &= \mathbf{w} \\ \mathbf{x} &\geq 0 \\ \mathbf{w} &\geq 0 \\ \mathbf{x}^{\mathbf{T}}\mathbf{w} &= 0 \end{aligned}$$

where $\mathbf{A}$ is a $n \times n$ matrix, and $\mathbf{x}$, $\mathbf{w}$, and $\mathbf{b}$ are $n \times 1$ vectors. Given $\mathbf{A}$ and $\mathbf{b}$, the goal is to solve for vectors $\mathbf{w}$ and $\mathbf{x}$ that satisfy the constraints. In the case of contact constraints, $\mathbf{x}$ and $\mathbf{w}$ are the vectors of normal forces and relative accelerations, respectively; the n^{th} element of each denotes the signed magnitude of the vector when projected along the n^{th} contact normal at the n^{th} contact point. $\mathbf{A}$ and $\mathbf{b}$ can be computed in time $O(n^2)$, where n is the number of contact points, if the bodies in contact are unarticulated [10]. If the bodies in contact are articulated and possess m links, $\mathbf{A}$ and $\mathbf{b}$ can be computed in time $O(mn)$ [11] or time (m^3n) [12]. Given $\mathbf{b}$ and the symmetric, positive-definite matrix $\mathbf{A}$, the desired normal force magnitudes can be computed in worst-case exponential time but expected polynomial time in the number of contact points [13].

When friction is added to the system, the problem becomes much harder computationally. For dynamic (sliding) friction, the number of constraints per contact point increases to two. Far more troubling, however, is that the matrix $\mathbf{A}$ generally becomes non-symmetric and non-positive definite. As Cottle notes [13], solving a LCP under such conditions is effectively as difficult as quadratic programming (i.e., NP-hard in general). Managing static (sticking) friction in this model is no easier. Static friction constrains the magnitude of the applied friction force at any contact point to be no greater than the coefficient of friction times the magnitude of the normal force. The magnitudes of the two tangent directions at the contact point are thus constrained nonlinearly (i.e., $\sqrt{f_{t_1}^2 + f_{t_2}^2} \leq \mu f_N$), and the problem becomes a nonlinear complementarity problem (NCP); as might be expected, solving a NCP is also as difficult as solving a quadratic programming problem. Indeed, Löstedt [14] used quadratic programming to compute the normal and frictional forces analytically. Baraff [1] presented a pivoting algorithm for computing the normal and frictional forces in $O(n^4)$ time; this algorithm is not guaranteed to terminate for systems with friction. As an aside, the problem of determining contact forces necessary to ensure that normal and frictional constraints are satisfied is NP-hard [15]. It is also possible that such a problem is *inconsistent*, meaning that no non-impulsive forces can be determined to satisfy all of the constraints [1].

Finally, approaches that compute static friction forces analytically, such as the method described above, exhibit a significant limitation with regard to control of articulated bodies. The equations used to compute inverse dynamics and the equations for determining sticking frictional forces are coupled, so standard inverse dynamics algorithms (i.e.,

[1]The term *impulse-velocity* is used because true (i.e., infinite force over an infinitesimally small time interval) impulses cannot be used in discrete time rigid body simulation. However, the velocities of colliding objects are updated as if an impulse had been applied.

[2]Hahn [7] also solved for collision impulses analytically, but did not handle resting contact separately, thereby leading to drift.

Recursive Newton-Euler [16]) cannot be employed.[3]

C. Impulse-based simulation

Mirtich's seminal thesis on *impulse-based* simulation [2]) modeled all contacts using trains of impulse-velocities. Mirtich not only provided an algorithm for applying impulses to articulated bodies simulated using generalized (rather than maximal) coordinates, but also for computing impulses using the Stronge hypothesis [17], which ensures that energy is conserved during collisions with Coulomb friction.

Mirtich's work suffers from a few weaknesses and limitations. For resting contact, he applies restitution coefficients above one to combat drift, which is somewhat unappealing. Additionally, this mechanism experiences significant slowdown when objects are resting upon another (i.e., stacked), as a result of the numerous propagated impulses. Mirtich's method for determining the proper impulse to apply to an articulated body runs in time $O(n)$ (n being the number of links in the body), but the constant factor is somewhat high.

D. Formulating the ODEs with constraints

A number of approaches toward rigid body simulation have investigated explicit time-stepping formulations [18], [19], [20], [21], [22], [23], [24], [4], [25], [26], [27]. The most recent of these approaches have combined the Newton-Euler equations for motion with joint and non-penetration constraints to produce methods that do not require separation into contact and collision. Non-penetration and frictional constraints are incorporated into the integration formula, and positions and velocities that satisfy the model for the next time step are determined using a LCP solver. These methods can typically prove the existence of solutions that satisfy the constraints when the integration step size is sufficiently low (i.e., inconsistent configurations do not occur).

Explicit time-stepping formulations are subject to three particular limitations. First, the computational burden can be significant: the mixed-LCP (unilateral and bilateral constraint) pivoting algorithm used by the more popular methods runs in time $O(n^4)$ in the number of constraints in the expected case. Second, interpenetration generally occurs due to drift, and must be corrected via an *ad hoc* post-stabilization method. Finally, these approaches have yet to be extended to articulated bodies formulated in generalized coordinates.

[3]Such inverse dynamics algorithms can be used, but the outputs will be incorrect. It remains to be seen whether standard inverse dynamics algorithms could still function in a useful feedforward capacity while not accounting for sticking frictional forces

E. Effective treatment of contacts and impacts using impulses

Guendelman et al. [3] introduced a novel idea to rigid body simulation: by interleaving collision and contact between integration of the velocity and position equations of motion, both contact and collision can be modeled with impulses without resorting to Mirtich's microimpulse method for resting contact. Guendelman et al.'s simulation paradigm consists of the following four phases:

1) modeling collisions (velocity update)
2) integration of acceleration ($\dot{v} = F/m$ and $\dot{\omega} = J^{-1}\tau$)
3) modeling contact (velocity update)
4) integration of velocity (position update)

Forces are incorporated in step (2). Forces that lead to interpenetration are effectively cancelled in step (3). In addition to this particular innovation, [3] is also the most notable work on simulation of rigid bodies with non-convex geometries. Guendelman et al.'s work was later extended to articulated bodies (using maximal coordinates) by Weinstein et al. [28], [29].

The work of Guendelman et al. is subject to some limitations. First, the use of Newton's law of restitution for handling frictional collision (or contact) is subject to energy gain, as Stronge noted in his thesis [17]; the authors could have used Mirtich's method to address this problem, but it is computationally expensive. Second, the method of Guendelman et al. suffers from drift for objects in resting contact, resulting in increasing interpenetration over time. Third, Guendelman et al. do not regress the simulation to the time of impact; skipping this step avoids significant computation, but often results in noticeable visual artifacts unless the step size is small or velocities are low. Finally, Guendelman et al. requires considerable overhead per articulated body formulated in generalized coordinates to determine the collision matrix, K, as is done in [2].

The broader issue with impulse-based methods, including the work of both Mirtich and Guendelman et al., is that they have traditionally been local methods: impulses are applied sequentially at points of intersection. This paradigm generally works quite well for unarticulated bodies. For articulated bodies under contact constraints that induce kinematic loops (e.g., a biped standing with both feet touching the ground), the constraints will likely not be satisfied after sequential application of impulses. This situation arises frequently under Coulomb friction, particularly sticking friction. Guendelmann et al. attempt to alleviate

this problem by performing several iterations of contact and collision processing. However, there is no guarantee of convergence to a stable solution, nor do there exist any heuristics for choosing the number of iterations. Global impulse-based methods, as suggested by Baraff [30] and Schmidl and Milenkovic [31], can potentially avoid these issues, though the computation required is essentially identical to that described in Section II-B.

F. Analytical computation of penalty forces

Hasegawa and Sato [32] developed a method for analytical computation of penalty forces; their particular application domain is haptic simulation, which requires high-frequency updates to drive force displays [33], and for which solving LCPs is too slow. To compute the penalty forces, the authors determine the intersection of the two convex polyhedra in contact using the algorithm of Preparata and Shamos [34] (running time of $O(N \lg N)$, where $N = m + n$ and m and n are the features in each polyhedron), and then integrate over the triangular volume of intersection.

The method of Hasegawa and Sato is quite fast compared to LCP methods, but suffers from two key limitations. First, determining the intersection of convex polyhedra suffers from degeneracies unless exact (and slow) arithmetic is utilized. If the volume of integration is quite small, then the intersection algorithm will likely fail, and penalty forces will not be computed; greater penetration will be the result. Second, the constant factor for this algorithm is quite high. The Muller-Preparata algorithm requires an intricate operation to determine a point interior to both polyhedra and two 3D convex hull operations. Additionally, the integration formulae for computing the penalty forces requires several thousand floating-point operations per triangle.

III. METHOD

Determining exact contact forces necessary to satisfy nonpenetration constraints is NP-hard, as noted in the previous section. Most of the methods discussed in the previous section are approximation methods; exact solutions are not computed in order to maintain acceptable simulation frequency. These methods make tradeoffs between accuracy, speed, and numerical stability. The method introduced in this section is not an exception. It was developed toward robotic simulation, which requires generalized coordinate formulations, maximal speed, and minimal oscillations arising from contact modeling.

Our method uses the normal velocity threshold paradigm introduced by Moore and Wilhelms [5] to classify contacts as either impacting or resting. The sequential impulse-based method of Baraff [35], modified to handle frictional collisions as in [5], [7], [3] and applied at the centroid of the contact surface, treats impacts; a penalty method handles resting contacts. Unlike traditional velocity threshold approaches, we apply our resting contact method during continuous contact regardless of the relative normal velocity; once continuous contact has begun, impacting contact can only be triggered if one of the bodies is impacted by a third body. This approach allows us to drive the velocity threshold to very near zero, thereby addressing the criticism that the threshold is set in an *ad hoc* manner. Figure 2 depicts the transitions between contact phases for a pair of bodies.

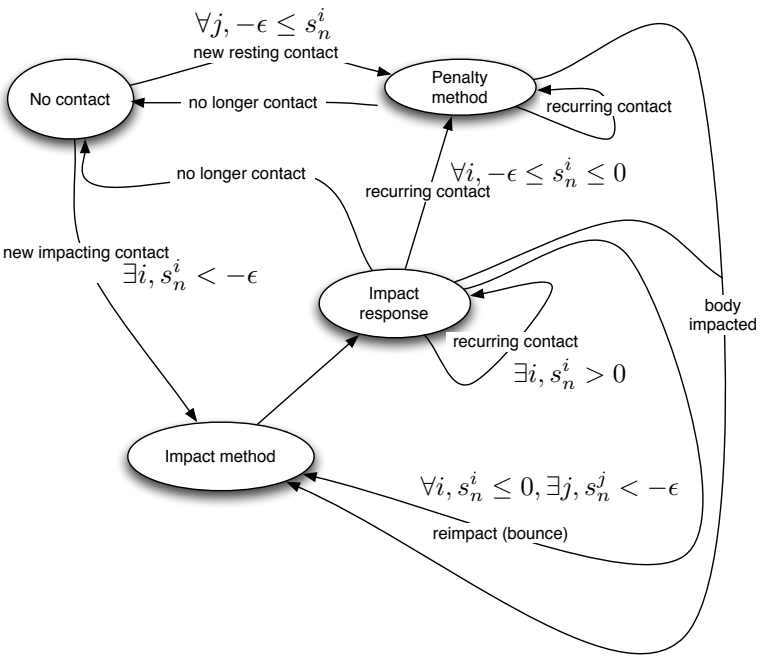


Fig. 2. The state transition diagram tracks the contact phase between two bodies. Note that after two bodies are in resting contact (i.e., the penalty method state), the impact method can only be triggered by a third body impacting one of the two. ϵ is a floating-point value very near to zero, and s_n^i is the relative normal velocity at the i^{th} contact point (the bodies are separating when $s_n^i > 0$).

We extend the traditional (i.e., deepest-point) penalty method in two ways. First, multiple points are used, rather than only the point of deepest penetration. These points are taken from the "bottom" surface of the contact geometry (see Figure 3). This approach yields stability to perturbation, as will be seen in Section IV. The second extension is the use of an integrative term, thus moving from the traditional proportional-derivative (PD) control paradigm to a proportional-integrative-derivative (PID) control model. The integrative term corrects steady-state error that over-

whelms the proportional term; practically, this means that stacks of objects can be simulated without extensively tweaking the PD gains.

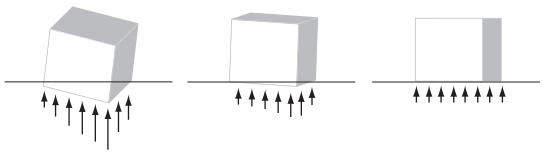


Fig. 3. The result of the application of restoring forces to multiple points of penetration (in the manner of the algorithm described in this article) on successive simulation steps. After a small period of time, the cube lies flat on the planar surface and experiences little oscillation.

Given points of contact and a normal (a robust method to compute these is given in Appendix I), computing penalty forces requires two steps: determining the contact points on the bottom of the volume of penetration and calculating restorative forces. These steps are described Sections III-A and III-B. This section concludes with a discussion on tuning gains.

A. Determining the points on the bottom of the volume of penetration

The bottom of the volume of penetration (see Figure 4) is defined as the subset of the intersection surface for which the inner product between the surface normal and the contact normal is non-positive (or non-negative, depending on the direction of the contact normal).

We apply restorative forces only to the points on the bottom of the volume of penetration to simulate the effect of virtual springs and dampers existing between the two objects. This effect is not only physically cogent, but also prevents the restorative forces from being "wasted" on surfaces that do not need correction (the applied forces are normalized by the number of contact points, as described in Section III-B.)

We compute points on the bottom of the volume of penetration in the following manner. First, points are sorted by their distance to the contact plane (i.e., the plane containing the facet from which the contact normal was obtained). An algorithm similar to Graham's Scan [36] is then employed. Our algorithm selects the farthest point to the contact plane as a comparator; if multiple such points exist, the points are projected onto the two-dimensional contact plane, and the bottom, right-most point is used. All points are then projected onto the contact plane, and the signed angles between the vertical axis and the lines passing through the comparator and every other point are computed.

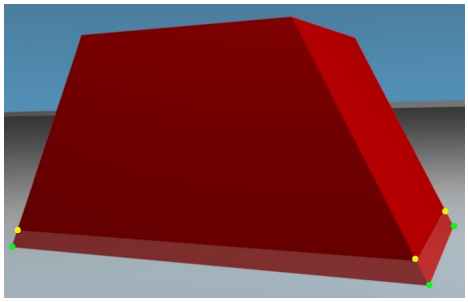


Fig. 4. We apply restorative forces only to the bottom of the volume of intersection. In this figure, an extruded isosceles trapezoid is penetrating the ground, represented by a translucent box. The volume of intersection is itself an extruded isosceles trapezoid, shown in a lighter shade of red. Six points of intersection are marked; the bottom points are depicted in green, while non-bottom points are depicted in yellow. The method described in this article would not apply restorative forces at the points marked in yellow.

The points are then processed in descending order keyed on the distance to the contact plane, and a polygon is constructed incrementally: the place in the sequence in which a candidate point would be inserted is determined by performing binary search using the signed angles, and the point is inserted if it lies outside the current polygon. Otherwise, the point is discarded. At the end of this process, seen in detail in Figure 5, all vertices of the polygon are on the bottom of the intersection volume. This algorithm runs in time $O(N \lg N)$ in the number of contact points.

B. Computing restorative and frictional forces

The restorative force magnitude applied to a given point ρ using a virtual spring and damper is given by:

$$||f(\rho)|| = \mathbf{max}\{\frac{k_p d(\rho) - k_v v(\rho)}{r}, 0\} \tag{1}$$

where $d(.)$ is the penetration depth determined using the contact normal, $v(.)$ is the relative velocity of the two bodies at the contact point projected along the contact normal, and k_p and k_v are proportional and derivative gains, respectively. We arbitrarily set the contact normal to point toward body p (i.e., away from body q); therefore, $v(\rho) \triangleq \dot{x}_p + \omega_p \times (\rho - \bar{x}_p) - \dot{x}_q - \omega_q \times (\rho - \bar{x}_q)$, where $\bar{x}_A$, $\dot{x}_A$, and ω_A are respectively the center-of-mass, linear velocity, and angular velocity for a body A. The $\mathbf{max}$ operator is used in Equation 1, because it is desired that contact forces only act to push objects apart (i.e., they cannot be "glued" together). The applied contact force is normalized using the number of contact points (r) at which non-zero force is applied, so that increasing the number of contact points does not result in greater applied force.

In addition to the proportional-derivative approach described above, we also utilize an integrative term to reduce

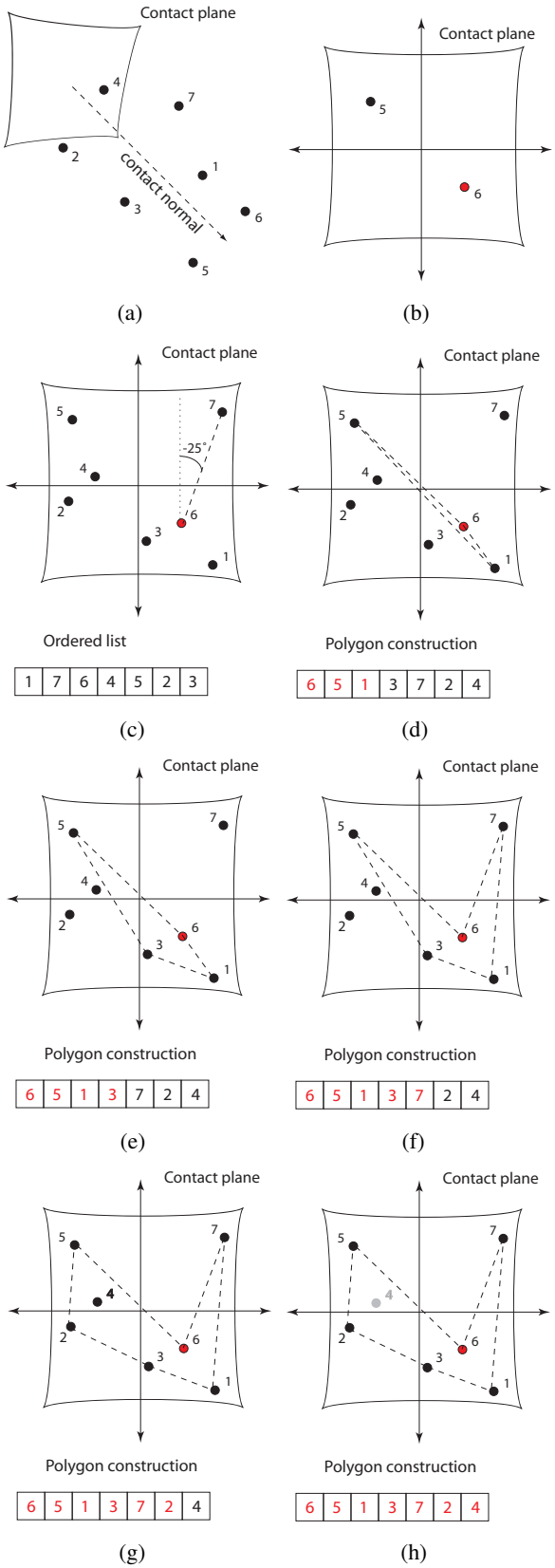


Fig. 5. Depiction of the algorithm for determining contact points on the bottom surface of the contact geometry. In (a), points are first sorted in decreasing distance from the contact plane. In (b) the farthest, bottom, right-most point– the comparator– is selected. In (c), the signed angles between the vertical axis and the line segment between the comparator and every other point is computed. (d)–(g) show a polygon constructed incrementally; all of the points are outside the polygon, and are thus on the bottom surface. In (h), point 4 is found to be inside the polygon (i.e., it is not on the bottom surface), and is discarded.

steady-state error. Incorporation of an integrative term is not as straightforward as the proportional and derivative terms. Contact points must be tracked over time, which is generally quite difficult, and the integrative term accumulates until the objects separate. This results in visible "popping", as objects resting on others briefly pop upwards.

Rather than accumulate the penetration error per contact point, we sum the maximum penetration error for the pair of bodies over the past time of the contact; this precludes the need to track the penetration error over individual contact points. To prevent popping, we utilize a "forgetting factor" (i.e., exponential decay), $0 < \lambda < 1$; λ works quite well for $0.8 < \lambda < 0.9$ empirically. The decay reduces the contributions of previous penetration errors, thus minimizing the popping phenomenon. As Figure 6 indicates, the force from the integrative term accumulates as time increases for higher values of penetration (e.g., greater than 0.1). In contrast, the applied force from the integrative term becomes essentially asymptotic for smaller amounts of penetration. This behavior is key, because it allows the integrative term to be used for cases when proportional-derivative terms are insufficient (e.g., stacks of objects) without generating excessive forces when the penetration is low. The resulting formulae for determining the magnitude of the restorative force is given in Equations 2 and 3.

$$I \triangleq \sum_{i=0}^{t-1} \lambda^{t-i-1} \mathbf{max}_y d(y,i) \tag{2}$$

$$||f(\rho)|| = \mathbf{max}\{\frac{k_p d(\rho,t) - k_v v(\rho) + k_i I}{r}, 0\} \tag{3}$$

In the equations above, $d(\rho, s)$ refers to the penetration depth of point ρ at step s of the simulation, and t is the current simulation step.

Once normal forces have been computed as above, frictional forces may be computed. For slipping friction, the relative velocity between the two bodies in the tangent plane is computed, and a force of magnitude $\mu||f(\rho)||$ is applied in direction opposite to the relative velocity. Viscous friction and Stribeck effects can be handled in a similar fashion. Static friction is considerably more difficult to simulate. Hwang et al. [37] and Yamane and Nakamura [38] have enjoyed some success by applying virtual springs and dampers in the tangent plane. In addition to that method, we have implemented static friction by solving the linear system $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ and $\mathbf{b}$ are determined in the manner used for LCP methods. Determining static friction forces in this manner is expensive, and thus warranted for modeling only certain types of contacts (e.g., wheels rolling on a surface).

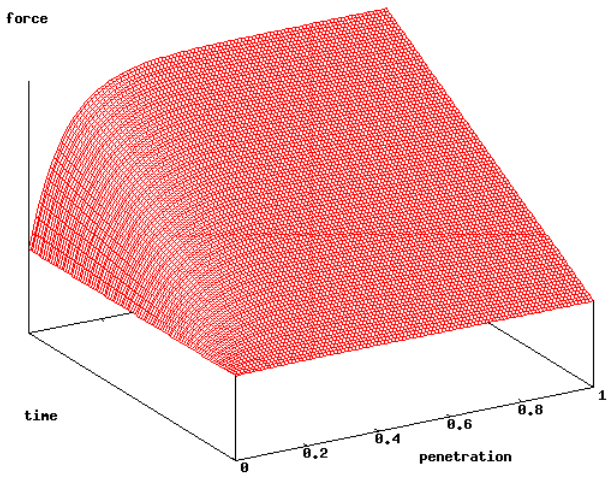


Fig. 6. Plot depicting the efficacy of the forgetting factor, λ, used in the integrative term. Because it is infeasible to plot the high-dimensional function $I(.)$ from Equation 2, each plane intersecting the x-axis (penetration) depicts constant penetration; for example, penetration of 0.1 at time 10 would indicate that the penetration depth remained at 0.1 throughout the time interval [0,10].

C. Tuning gains

The method described in this section introduces a new gain, thereby increasing the difficulty of the tuning process over that required by the standard penalty method. There are principled methods to select the gains when the plant dynamics (i.e., dynamics of the two bodies in contact) are known; we assume no such knowledge, given that arbitrary forces may be applied to to these bodies.

Fortunately, the addition of the integral term tends to decrease sensitivity of gain selection: the proportional gain can be decreased, because it no longer needs to correct steady-state error. The examples described in the next section all used the following set of gains: $k_p = 100.0(m_1 + m_2)$, $k_v = 50.0(m_1 + m_2)$, and $k_i = 2.0(m_1 + m_2)$, where m_1 and m_2 are the masses of the two bodies in contact. There also exist numerous heuristics for setting gains for PID controllers, such as the Ziegler-Nichols method [39]. It is also possible to tune gains offline to maximize performance criteria.

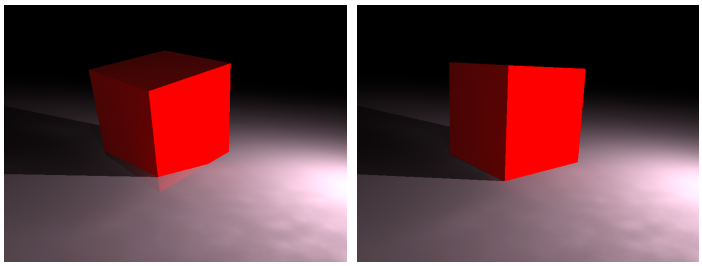
IV. Validation

It would be ideal to validate our contact model against real-world physical phenomena. However, physicists have yet to describe models for contact and collision that accurately reflect rigid-body collisions, in part because the rigid-body assumption (i.e., Young's modulus going to infinity) is often a poor approximation. When this omission is combined with the incomplete understanding of frictional effects and the difficulty of modeling surface textures, it becomes challenging to empirically validate the fidelity of rigid body simulation with contacts. That is why some in the computer graphics community favor instead the subjective standard of physically plausible motion (e.g., Barzel et al. [40], Guendelman et al. [3]).

Rather than attempt to evaluate the physical accuracy of our method, we instead present several examples that stress simulation speed, physical plausibility, and stability to perturbations during resting contact. Specifically, our examples consist of a block penetrating a plane, a block resting on a plane undergoing perturbations, a pile of objects of different shapes and masses, and two robots operating in planar environments. Further explanation and analysis of the examples follow in the remainder of this section.

A. Block penetrating a plane

A block is set to penetrate a plane before beginning simulation (see Figure 7) to illustrate the effectiveness of our method in correcting interpenetration with minimum oscillation. Though this example is simple, it has important ramifications for robotic simulation in particular. Simulation of bipeds using the deepest-point penalty method is quite difficult due to the oscillations produced. Figures 8 and 9 illustrate the stability and penetration error of our method compared to the deepest-point penalty method, both of which utilized the integrative term.



(a) (b)

Fig. 7. A block penetrating a plane (a) before the application of our method (block is set to penetrate plane prior to simulating) and after (b) 1,000 time steps have elapsed. Note that the proportional gain could be increased to correct interpenetration faster (i.e., improve the rise time), though popping might result.

We used the same initial configuration of the block for both methods. A step-size of $1e^{-3}$ was used with fourth-order Runge-Kutta integration. We used the gains for the multiple-point method as the starting point for the deepest point method, and sought to achieve a good compromise between penetration and popping; we ultimately selected $k_p = 6000$, $k_v = 3000$, $k_i = 200$ for the deepest-point method, versus $k_p = 5000$, $k_v = 2500$, $k_i = 100$ for the multiple-point method. We expended considerable effort

to determine the gains for the deepest-point method in an attempt to meet the performance of the multiple-point method; we were unable to lower either the kinetic energy or the penetration for the former method significantly using any combination of gains.

It is meaningless to compare our method to a LCP-based approach for resting contact on this example, because LCP approaches determine forces necessary to yield zero acceleration at contact points: they are unable to correct for existing penetration.

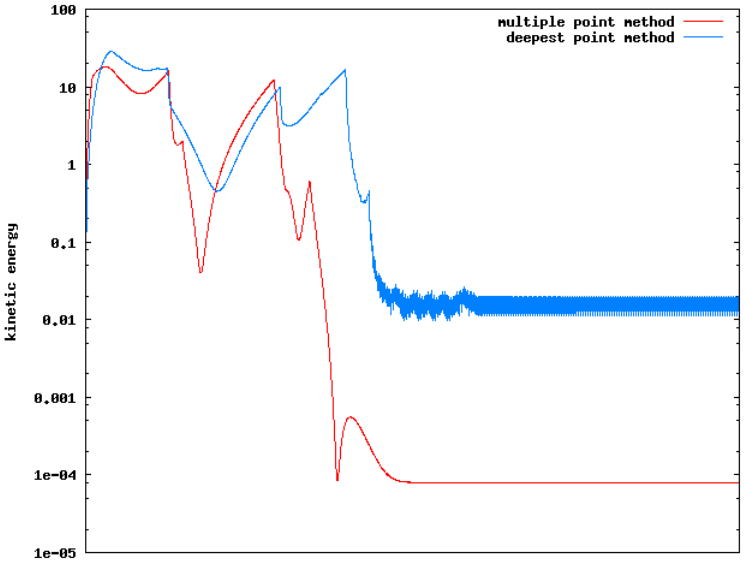


Fig. 8. Plots of kinetic energy on the problem of a randomly-oriented block penetrating the plane under the influence of gravity over 3,000 simulation steps. Note that the y-axis is log-scale.

B. Block with applied forces

We tested a block experiencing downward forces of random magnitude (sampled uniformly from the interval [10,100]) applied at random points on the top surface of the block (see Figure 12). Each force was applied continuously for 500 time steps to a given point, and then no force was applied for another 500 steps to give the method time to return the block to stasis. The process was then repeated, for a total of ten iterations. Fourth-order Runge-Kutta integration was used with a step-size of $1e^{-3}$.

Figures 10 and 11 illustrate the performance of three methods: the deepest-point penalty method, the multiple-point penalty method, and a LCP-based method that uses the deepest-point penalty method to combat drift. The LCP solver used is a generic iterative method described in [41]. Figure 10 indicates that the stability to perturbations of our method is generally several orders of magnitude better than the deepest-point method; our method performs similarly to

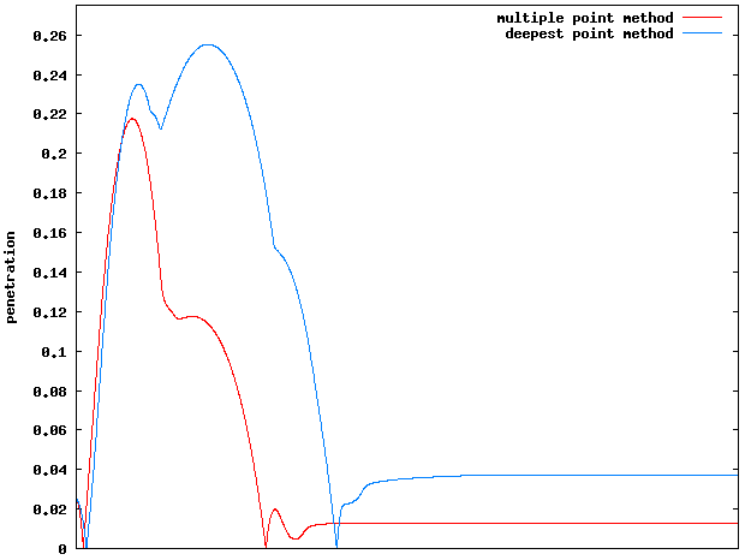


Fig. 9. Plots of absolute penetration on the problem of a randomly-oriented block penetrating the plane under the influence of gravity over 3,000 simulation steps. The block measures one unit per side, so a penetration of 0.02 represents a penetration of 2%.

the LCP-based method. Figure 11 demonstrates the clear advantage of the LCP-based method over the others with respect to penetration; the mean penetration of the LCP-based method is $1.5e^{-4}$, which is almost two orders of magnitude better than the mean penetration of our method at $5.9e^{-3}$. However, our method performs an order of magnitude better than the deepest point method, which exhibits a mean penetration of $9.0e^{-2}$.

C. Pile of objects

Stacks and piles of objects are often problematic for contact methods and for local methods in particular. The constant propagation of forces will cause divergence in simulations of stacks of objects modeled with a penalty if the step size is above $1/n$, as Kass [42] notes. In practice, the step size must be several orders of magnitude smaller than $1/n$.

Figure 13 shows a pile of objects for which contacts are modeled using our penalty method. The objects consist of boxes, cylinders, and spheres of various sizes and masses; the masses are determined by selecting from a uniform distribution over the interval $[0, 100]$. The objects are resting inside a translucent box and press against each other and the box.

D. Simulated robots

We illustrate the advantage of our method over a LCP-based approach in the domain of robotic simulation with

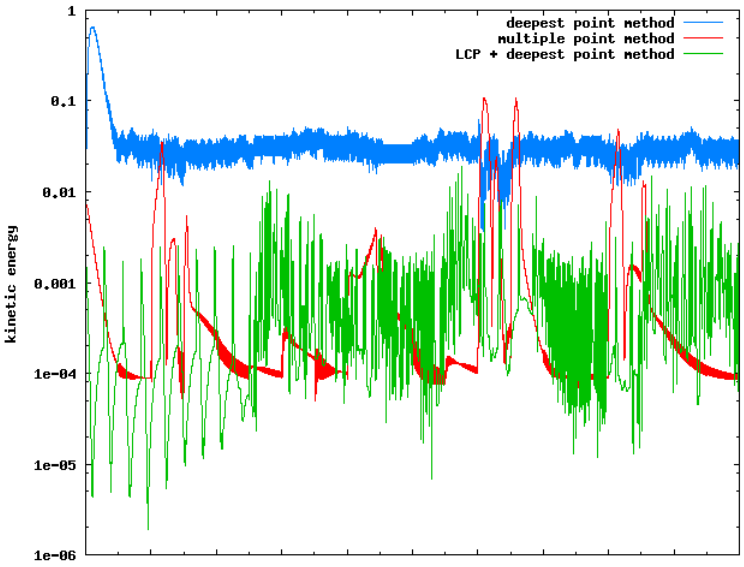


Fig. 10. Plots of kinetic energy on the example of forces being applied to the top of a block resting on a plane; zero kinetic energy is ideal. The forces are applied from every major tick mark to every minor tick mark on the x-axis. No force is applied in the intervals from the minor tick marks to the major tick marks to illustrate the stability of the methods. Note that the y-axis is log-scale.

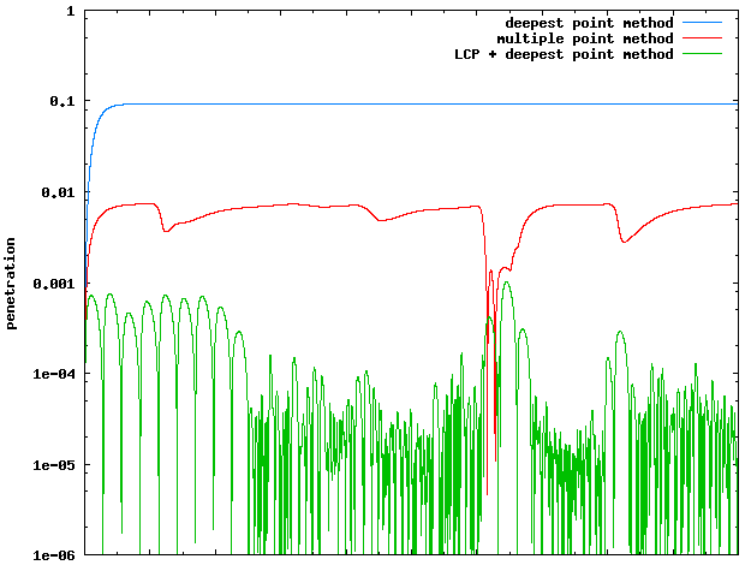


Fig. 11. Plots of penetration on the example of forces being applied to the top of a block resting on a plane; zero penetration is ideal. The block measures one unit per side, so a penetration of 0.02 represents a penetration of 2%. Note that the y-axis is log-scale.

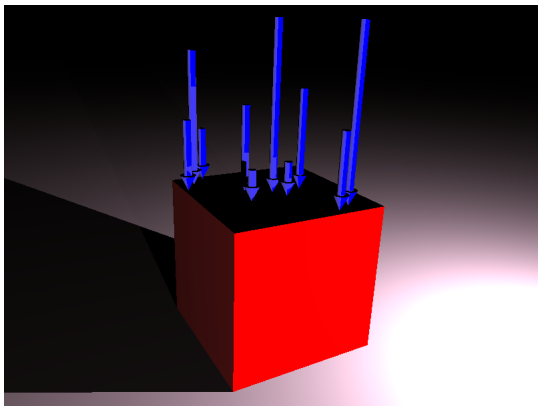


Fig. 12. Positions and magnitudes of forces applied to perturb a cube resting on the plane. Each arrow points to the position that the force was applied, and the arrow's length denotes its magnitude. An arrow of unit length (i.e., an arrow with a length of a side of the cube) would denote a force of 100N (the mass of the cube is 10kg, and gravity is 9.8m/s^2, so the weight of the cube alone exerts a force of 98.1N on the plane). The forces were not applied simultaneously; rather, one force was applied for the first 500 out of every 1000 time steps.

two examples. The first example is the two-wheel, differentially driven robot shown in Figure 15, which is similar to various commercially available mobile robots. The second example is of a humanoid robot (Figure 14), which has a differentially driven base, but possesses 26 degrees-of-freedom. Both robots are in contact with only the ground plane, and the number of contact points per wheel is limited to ten.

We used the method proposed by Kokkevis [11] to determine the LCP matrix $\mathbf{A}$ and vector $\mathbf{b}$. Ruspini and Khatib [12] provide an alternate method that runs in time $O(m^3)$; their method is able to reuse results from forward dynamics (assuming that the Composite Rigid Body algorithm [16] is used) and may be faster, particularly for bodies with relatively few degrees-of-freedom. Further research is necessary to determine where the approaches by Kokkevis and Ruspini and Khatib are most applicable.

Both examples were run for 2500 iterations. The LCP-based method required 76.7 seconds to simulate the mobile robot (32.6 steps/second), while the penalty method completed in 31.0 seconds (80.6 steps/second). Simulating the humanoid robot required 566.9 seconds with the LCP method (4.4 steps/second), while the penalty method completed in 81.4 seconds (30.7 steps/second). These examples show that the $O(mn)$ factor (m is the degrees-of-freedom of the robot, n is the number of contact points) in LCP-based methods is considerable, and also that the LCP-based method is far slower, even for relatively few contact points and degrees-of-freedom in the contacting bodies.

V. Discussion

It is somewhat unfair to compare the deepest-point penalty method, multiple-point penalty method, and LCP-based method for resting contact using the number of contact points alone because the process of finding the contact points is a factor as well; the three methods present different requirements in this regard. The deepest-point penalty method requires tracking only the closest

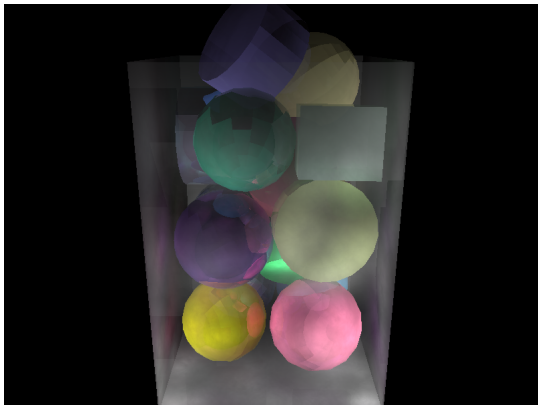


Fig. 13. A pile of objects of different shapes and masses. The pile is in a steady configuration.

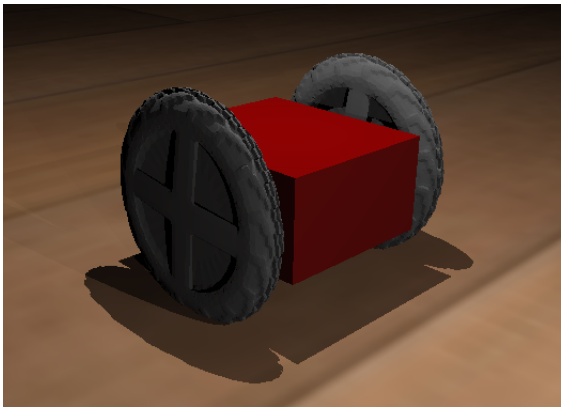


Fig. 15. A differential-drive mobile robot (two joint, "floating" base) in a planar environment.

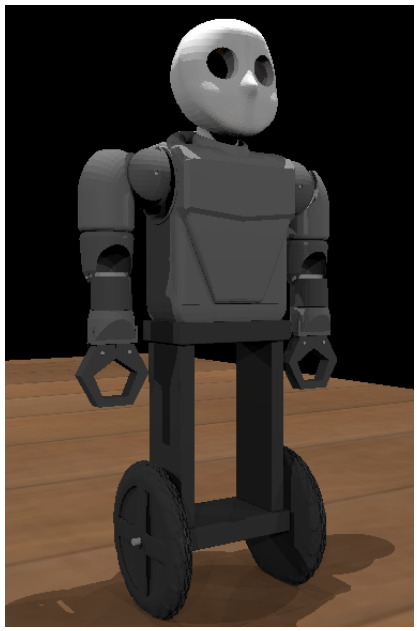


Fig. 14. A 26 degree-of-freedom humanoid robot operating in a planar environment. LCP-based methods for resting contact are too slow for this articulated body, even when the contact points are relatively few (less than ten per wheel in the example).

features of two bodies; updating the closest features can be done in $O(1)$ time [43], [44], and the contact normal can be computed in constant time as well. The multiple-point method uses the method described in Appendix I to determine contact points and the contact normal; its worst case complexity is $O(mn)$, where m and n are the numbers of features of two convex polyhedra. LCP-based approaches can potentially use the $O(1)$ methods noted above, though oscillation will still be a problem. Baraff [35] described a $O(mn)$ algorithm for collision detection and contact finding for use with a LCP-based method. A near $O(m+n)$ method might be possible by tracking nearest features and using signed-distance functions; thus, it may be possible to utilize the information that the polyhedra are only touching or penetrating slightly to reduce the computational complexity. Further research is necessary to investigate whether the disparities in requirements for finding contact points and the contact normal might offset the complexity requirements of the individual methods.

Determining contact forces can be viewed as a control theoretic problem complicated by unknown plant dynamics: in general, arbitrary forces can be applied to the bodies in contact. This article indicates that not only can resting contact be achieved effectively using more powerful models than the proportional-derivative controller, but that other techniques from control theory might lead to even better performance. In the future, we plan to investigate using adaptive control to tune penalty gains online, which would likely improve performance while eliminating the most significant drawback of penalty methods, the need to tune gains.

Appendix I
Determining Points of Contact and Contact Normals

Determining points of contact and calculating contact normals have been discussed at length in the context of rigid body simulation. Nevertheless, we have found that these processes are typically targeted toward nonpenetrating contact, as in Baraff [35]. When these processes are conducted in the context of penetrating contact, such as in Hasegawa and Sato [32], estimations are often employed– particularly when computing contact normals– which can result in highly undesirable behavior. As a result, this section details the exact processes used in this article to find points of contact, determine the point of deepest penetration, and compute the contact normal. The processes described in

this section have proven to be robust over a wide range of simulation models.

A. Finding points of contact and determining the point of deepest penetration

In order to rapidly find points of contact, we represent collision geometries using both triangle meshes and signed distance functions (computed offline), as in Guendelman et al. [3]. We use the *adaptively sampled distance field* (ADF) of Frisken et al. [45], which provides very fast inside/outside queries using an octree, though other approaches are viable as well, such as that of Jones et al. [46].

We first determine the set of pairs of intersecting triangles. We query individual vertices of these triangles against the ADFs; points found to be inside the other geometry have their adjacent triangles added for processing, thus handling the case where one or more triangles is completely inside the mesh of the other. Naturally, we also store the point of deepest penetration, which is simply the vertex that results in the minimum signed distance, as reported by the ADFs. We are not the first to use an approximate method for computing the point of deepest penetration; previous work includes that of Fisher and Lin [47].

Given that the number of vertices of a given polyhedron s is v_s, the expected asymptotic complexity for this operation is $O(v_p \lg N_q + v_q \lg N_p)$, where N_p and N_q are the number of cells of the respective ADFs. Though it is theoretically possible for the octrees underlying the ADF representations to be extremely poorly balanced, resulting in $O(v_p N_q + v_q N_p)$ complexity, in practice. In comparison, the worst-case complexity of 3D penetration depth computation for exact approaches is $O(mn)$ [48], where m and n are the number of features in each polyhedron. Frisken et al. [45] have indicated that the number of cells required to accurately represent an arbitrary polyhedron with an ADF may be an order of magnitude lower than the number of features in the polyhedron, thus implying that $N_s << v_s$.

Some geometries may utilize fewer vertices per surface area than others (e.g., long cylinders, boxes, etc.), resulting in relatively few contact points. For these shapes, we subdivide the triangle meshes offline using Steiner points until all triangles are below a minimum area. We have observed our method to be robust to non-uniform distributions of vertices over the intersecting polyhedra.

B. Determining the contact normal

Once the point of deepest penetration has been determined, the normal is computed to be in the direction that will separate the two objects with minimum translation. There are two vectors that satisfy this requirement, pointing in opposite directions.

To determine the contact normal, each point of penetration is projected along the facet normals of the alternate geometry to the individual planes containing the facets. The contact normal will lie in the direction of the facet normal that yields the minimum of the maximum distances of contact points to facets (see Equation 4). Only facets that are intersecting or inside the alternate geometry are considered.

$$\hat{n} = \min_{\hat{n}_i}(\max|\hat{n}_i \cdot p_j - d_i|) \tag{4}$$

The complexity of this operation is $O(v_p v_q)$, where v_p is the number of features of a polyhedron. This operation often can be speeded considerably by first determining the contact normal using the process above with only the point of deepest penetration, exhibiting $O(v_s)$ complexity (v_s is the number of features of the second geometry). The distance of all contact points to the facet plane is then computed; if a distance is greater than the distance from the deepest point of penetration to the facet, then the full process described in the paragraph above will need to be performed.

REFERENCES

[1] D. Baraff, "Fast contact force computation for nonpenetrating rigid bodies," in *Proc. of SIGGRAPH*, Orlando, FL, July 1994.

[2] B. Mirtich, "Impulse-based dynamic simulation of rigid body systems," Ph.D. dissertation, University of California, Berkeley, 1996.

[3] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking," *ACM Trans. on Graphics*, vol. 22, no. 3, pp. 871–878, 2003.

[4] D. Stewart and J. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with coulomb friction," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, San Francisco, CA, April 2000.

[5] M. Moore and J. Wilhelms, "Collision detection and response for computer animation," in *Proc. of Intl. Conf. on Computer Graphics and Interactive Techniques*, 1988, pp. 289–298.

[6] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Engelwood Cliffs, NJ: Prentice Hall, 1971.

[7] J. K. Hahn, "Realistic animation of rigid bodies," *Computer Graphics*, vol. 22, no. 4, 1988.

[8] D. Baraff, "Linear-time dynamics using lagrange multipliers," in *Proc. of Computer Graphics*, New Orleans, LA, Aug 1996.

[9] P. Löstedt, "Mechanical systems of rigid bodies subject to unilateral constraints," *SIAM Journal on Applied Mathematics*, 1982.

[10] D. Baraff, "An introduction to physically based modeling: Rigid body simulation II– nonpenetration constraints," Robotics Institute, Carnegie Mellon University, Tech. Rep., 1997.

[11] E. Kokkevis, "Practical physics for articulated characters," in *Proc. of Game Developers Conf.*, 2004.

[12] D. Ruspini and O. Khatib, "A framework for multi-contact multi-body dynamic simulation and haptic display," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2000.

[13] R. W. Cottle, J.-S. Pang, and R. Stone, *The Linear Complementarity Problem*. Boston: Academic Press, 1992.

[14] P. Löstedt, "Numerical simulation of time-dependent contact friction problems in rigid body mechanics," *SIAM J. of Scientific Statistical Computing*, vol. 5, no. 2, pp. 370–393, 1984.

[15] D. Baraff, "Coping with friction for non-penetrating rigid body simulation," *Computer Graphics*, vol. 25, no. 4, pp. 31–40, 1991.

[16] R. Featherstone, *Robot Dynamics Algorithms*. Kluwer, 1987.

[17] W. J. Stronge, "Rigid body collisions with friction," *Proc. of the Royal Society of London A*, vol. 431, no. 169–181, 1990.

[18] J. J. Moreau, "Standard inelastic shocks and the dynamics of unilateral constraints," in *C.I.S.M. Courses and Lectures*, G. del Piero and F. Maceri, Eds. Vienna: Springer-Verlag, 1985, vol. 288, pp. 173–221.

[19] J. J. Moreau, "Unilateral contact and dry friction in finite freedom dynamics," in *Nonsmooth Mechanics and Applications*, J. J. Moreau and P. D. Panagiotopoulos, Eds. Vienna: Springer-Verlag, 1988, pp. 1–82.

[20] M. D. P. M. Marques, "Differential inclusions in nonsmooth mechanical problems: Shocks and dry friction," in *Progress in Nonlinear Differential Equations and Their Applications*. Basel: Birkhäuser Verlag, 1993, vol. 9.

[21] J. J. Moreau, "Numerical experiments in granular dynamics: Vibration-induced sized segregation," in *Contact Mechanics*, M. Raous, M. Jean, and J. J. Moreau, Eds. New York: Plenum Press, 1995, pp. 347–358.

[22] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction," *Intl. Journal for Numerical Methods in Engineering*, vol. 39, no. 15, pp. 2673–2691, 1996.

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

[24] M. Anitescu, F. Potra, and D. Stewart, "Time-stepping for three dimensional rigid body dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 177, pp. 183–197, 1999.

[25] M. Anitescu and F. A. Potra, "A time-stepping method for stiff multi-rigid-body dynamics with contact and friction," *Intl. Journal for Numerical Methods in Engineering*, vol. 55, pp. 753–784, 2002.

[26] M. Anitescu and G. Hart, "A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contacts, and friction," *Intl. Journal for Numerical Methods in Engineering*, vol. 60, no. 14, pp. 2335–2371, 2004.

[27] F. A. Potra, M. Anitescu, B. Gavrea, and J. Trinkle, "A linearly implicit trapezoidal method for stiff multibody dynamics with contact, joints, and friction," *Intl. Journal for Numerical Methods in Engineering*, vol. 66, no. 7, pp. 1079–1124, 2006.

[28] R. Weinstein, J. Teran, and R. Fedkiw, "Pre-stabilization for rigid body articulation with contact and collision," in *Proc. of ACM SIGGRAPH*, 2005.

[29] R. Weinstein, J. Teran, and R. Fedkiw, "Dynamic simulation of articulated rigid bodies with contact and collision," *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 3, pp. 365–374, May/Jun 2006.

[30] D. Baraff, "Analytical methods for dynamic simulation of non-penetrating rigid bodies," *Computer Graphics*, vol. 23, no. 3, July 1989.

[31] H. Schmidl and V. J. Milenkovic, "A fast impulsive contact suite for rigid body simulation," *IEEE Trans. on Visualization and Computer Graphics*, vol. 10, no. 2, pp. 189–197, March/April 2004.

[32] S. Hasegawa and M. Sato, "Real-time rigid body simulation for haptic interactions based on contact volume of polygonal objects," in *Proc. of Eurographics*, 2004.

[33] L. Love and W. Book, "Contact stability analysis of virtual walls," in *Proc. of Dynamic Systems and Control Divsion ASME*, 1995.

[34] D. Muller and F. Preparata, "Finding the intersection of two convex polyhedra," *Theoretical Computer Science*, vol. 7, pp. 217–236, 1978.

[35] D. Baraff, "An introduction to physically based modeling: Rigid body simulation ii – constrained rigid body dynamics," Robotics Institute, Carnegie Mellon University, Tech. Rep., 1997.

[36] J. O'Rourke, *Computational Geometry in C*, 2nd ed. Cambridge University Press, 2001.

[37] Y. Hwang, E. Inohira, A. Konno, and M. Uchiyama, "An order n dynamic simulator for a humanoid robot with a virtual spring-damper contact model," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Taipet, Taiwan, September 2003.

[38] K. Yamane and Y. Nakamura, "Stable penalty-based model of frictional contacts," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Orlando, FL, USA, May 2006.

[39] J. G. Ziegler and N. B. Nichols, "Optimal settings for automatic controllers," *Trans. American Society Mech. Engineers*, vol. 64, pp. 759–762.

[40] R. Barzel, J. F. Hughes, and D. N. Wood, "Plausible motion simulation for computer graphics animation," in *Computer Animation and Simulation (Proc. Eurographics Workshop)*, R. Boulic and G. Hégron, Eds., 1996, pp. 183–197.

[41] K. G. Murty, *Linear Complementarity, Linear and Nonlinear Programming*. Berlin: Heldermann Verlag, 1988.

[42] M. Kass, "An introduction to continuum dynamics for computer graphics," Pixar, Tech. Rep., 1997.

[43] M. C. Lin and J. F. Canny, "A fast algorithm for incremental distance calculation," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1991, pp. 1008–1014.

[44] B. Mirtich, "V-Clip: fast and robust polyhedral collision detection," *ACM Trans. on Graphics*, vol. 17, no. 3, pp. 177–208, 1998.

[45] S. F. Frisken, R. N. Perry, A. P. Rockwood, and T. R. Jones, "Adaptively sampled distance fields: A general representation of shape for computer graphics," *Computer Graphics (Proc. of ACM SIGGRAPH)*, 2000.

[46] M. W. Jones, J. A. Baerentzen, and M. Sramek, "3D distance fields: A survey of techniques and applications," *IEEE Trans. on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 581–599, July/August 2006.

[47] S. Fisher and M. C. Lin, "Deformed distance fields for simulation of non-penetrating flexible bodies," in *Proc. of the Eurographic Workshop on Computer Animation and Simulation*, Manchester, UK, 2001, pp. 99–111.

[48] Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha, "Fast penetration depth computation for physically-based animation," in *Proc. of Symposium on Computer Animation (SCA)*, 2002.